hURL 1.2

The Multi-Protocol Data Transfer Plugin for Hollywood

Andreas Falkenhahn

Inhaltsverzeichnis

1 Allgemeine Informationen

1.1 Einführung

hURL ist ein Plugin für Hollywood, mit dem Sie Daten mit vielen verschiedenen Protokollen übertragen können. Basierend auf Curl unterstützt hURL eine unglaublich breite Palette von Übertragungsprotokollen, z.B. DICT, FILE, FTP, FTPS, Gopher, HTTP, HTTPS, IMAP, IMAPS, LDAP, LDAPS, POP3, POP3S, RTMP, RTSP, SCP, SFTP, SMB, SMBS, SMTP, SMTPS, Telnet und TFTP. Darüber hinaus unterstützt hURL SSL-Zertifikate, HTTP POST, HTTP PUT, FTP-Upload, HTTP-Formular-basierter Upload, Proxies, HTTP/2, Cookies, Benutzer+Passwort-Authentifizierung (Basic, Plain, Digest, CRAM-MD5, NTLM, Negotiate und Kerberos), File-Transfer-Resume, Proxy-Tunnel und mehr. Es ist wirklich das ultimative Datentransfer-System für Hollywood, welches keine Wünsche offen lässt.

Es gibt zwei Möglichkeiten, hURL zu verwenden: Es gibt eine High-Level-Schnittstelle, die sich direkt in die Hollywood-Netzwerkbibliothek einbinden kann und diese um hURL-Funktionen wie SSL/TLS-Unterstützung erweitert. Dies ermöglicht die Verwendung von Hollywood-Befehlen wie `DownloadFile()`, um Dateien mit benutzerdefinierten Protokolle herunterzuladen, die Hollywood selbst nicht unterstützt, z.B. SSL/TLS.

Eine weitere Möglichkeit, hURL zu verwenden, ist die Low-Level-Schnittstelle: Diese Schnittstelle ermöglicht es Ihnen, direkt aus Hollywood-Skripten auf die Curl-API zuzugreifen. Dies ist extrem leistungsstark, da es Ihnen den Zugriff auf Hunderte von verschiedenen Curl-Optionen ermöglicht und Sie können hURL auf Ihre Bedürfnisse abstimmen. hURL enthält über 500 Befehle, um alle Ihre Anforderungen an den Datentransfer zu erfüllen!

Schließlich enthält hURL eine umfangreiche Dokumentation in verschiedenen Formaten wie PDF, HTML, AmigaGuide und CHM, die detaillierte Beschreibungen aller vom Plugin angebotenen Funktionen und Methoden enthält.

All dies macht hURL zum ultimativen Datentransfer-Werkzeug für Hollywood, das alles enthält, was Sie zum Senden und Empfangen von Daten über fast jedes Übertragungsprotokoll auf der Welt benötigen.

1.2 Lizenzbestimmungen

3. Wenn Sie das Programm auf einer Coverdisk veröffentlichen möchten, müssen Sie erst um Erlaubnis fragen.

Diese Software verwendet Curl von Daniel Stenberg. Siehe Abschnitt A.1 [curl license], Seite 297, für Details.

Diese Software verwendet Lua-cURL von Alexey Melnichuk. Siehe Abschnitt A.2 [Lua-cURL], Seite 297, für Details.

Diese Software verwendet libssh2. Siehe Abschnitt A.4 [libssh2 license], Seite 299, für Details.

Dieses Produkt enthält Software, die vom OpenSSL-Projekt zur Verwendung im OpenSSL-Toolkit entwickelt wurde (`http://www.openssl.org/`) Siehe Abschnitt A.3 [OpenSSL license], Seite 297, für Details.

Das Symbol im hURL-Logo ist von Vectors Market von www.flaticon.com und ist von CC 3.0 BY lizenziert.

Alle Warenzeichen sind Eigentum ihrer jeweiligen Firmen.

FÜR DIESES PROGRAMM GIBT ES KEINE GARANTIE, SOWEIT ES DIE ANZUWENDENDEN GESETZE ZULASSEN. SOFERN ANDERSWO NICHTS GEGENTEILIGES GESCHRIEBEN STEHT STELLEN DER AUTOR UND/ODER DRITTE DAS PROGRAMM "SO WIE ES IST" ZUR VERFÜGUNG, OHNE IRGENDEINE GARANTIE, WEDER DIREKT NOCH INDIREKT. DIES BEINHALTET, IST ABER NICHT DARAUF BESCHRÄNKT, VERKÄUFLICHKEIT UND EIGNUNG FÜR EINEN BESTIMMTEN VERWENDUNGSZWECK. DAS VOLLSTÄNDIGE RISIKO DER QUALITÄT UND AUSFÜHRBARKEIT DES PROGRAMMS LIEGT BEIM ANWENDER. SOLLTE SICH DAS PROGRAMM ALS DEFEKT HERAUSSTELLEN, LIEGEN ALLE KOSTEN FÜR SERVICE, INSTANDSETZUNG ODER NACHBESSERUNG BEIM ANWENDER.

KEIN COPYRIGHT-INHABER ODER DRITTER, DER DAS PROGRAMM WIE OBEN ERLAUBT WEITERVERKAUFT, KANN FÜR SCHÄDEN IRGENDWELCHER ART HAFTBAR GEMACHT WERDEN (DIES BEINHALTET, IST ABER NICHT BESCHRÄNKT AUF, DATENVERLUST INFOLGE UNFÄHIGKEIT DES PROGRAMMS, MIT ANDEREN PROGRAMMEN ZUSAMMENZUARBEITEN), SELBST WENN EIN SOLCHER INHABER ODER DRITTER AUF DIE MÖGLICHKEIT EINES SOLCHEN SCHADENS HINGEWIESEN WURDE, AUSSER ES BESTEHT EINE SCHRIFTLICHE EINWILLIGUNG ODER WIRD VOM GESETZ VERLANGT.

1.3 Anforderungen

- Hollywood 8.0 oder besser; beachten Sie, dass empfohlen wird, mindestens Hollywood 9.0 mit hURL zu verwenden, da Sie nur mit Hollywood 9.0 (oder besser) die High-Level-Schnittstelle von hURL voll ausnutzen können.
- unter macOS benötigt hURL mindestens 10.8 auf x86 sowie x64 Systemen und 10.4 auf PowerPC Systemen
- auf Android ist mindestens Version 5.0 erforderlich
- auf AmigaOS und Kompatiblen ist AmiSSL v4 erforderlich

1.4 Installation

Die Installation von hURL ist unkompliziert und einfach: Kopieren Sie einfach die Datei `hurl.hwp` für die Plattform Ihrer Wahl in Hollywoods Plugin-Verzeichnis. Auf allen Systemen außer auf AmigaOS und kompatiblen müssen Plugins in einem Verzeichnis mit dem Namen `Plugins` gespeichert werden, das sich im selben Verzeichnis wie das Hauptprogramm von Hollywood befindet. Auf AmigaOS und kompatiblen Systemen müssen Plugins stattdessen in `LIBS:Hollywood` installiert werden. Unter macOS X muss sich das Verzeichnis `Plugins` im Verzeichnis `Resources` des Programmpakets befinden, d.h. im Verzeichnis `HollywoodInterpreter.app/Contents/Resources`. Beachten Sie, dass `HollywoodInterpreter.app` im Programmpaket `Hollywood.app` selbst gespeichert ist, nämlich in `Hollywood.app/Contents/Resources`.

Anschliessend kopieren Sie den Inhalt des Ordners `Examples` in den Ordner `Examples` Ihrer Hollywood-Installation. Alle hURL-Beispiele erscheinen dann in Hollywoods GUI und Sie können sie bequem von der Hollywood-GUI oder IDE aus starten und anzeigen.

Unter Windows sollten Sie auch die Datei `hURL.chm` in das Verzeichnis `Docs` Ihrer Hollywood-Installation kopieren. Wenn sich dann der Cursor über einem hURL-Befehl in der Hollywood-IDE befindet, können Sie die Online-Hilfe aufrufen, indem Sie F1 drücken.

Unter Linux und macOS kopieren Sie das Verzeichnis `hURL`, das sich im Verzeichnis `Docs` des hURL-Distributionsarchivs befindet, in das Verzeichnis `Docs` Ihrer Hollywood-Installation. Beachten Sie, dass sich unter macOS das Verzeichnis `Docs` innerhalb des Programmpakets `Hollywood.app` befindet, d.h. in `Hollywood.app/Contents/Resources/Docs`.

2 Über hURL

2.1 Danksagungen

hURL wurde von Andreas Falkenhahn,geschrieben, basierend auf der Arbeit von Alexey Melnichuk und Daniel Stenberg.

Ein besonderer Dank geht an Helmut Haake und Dominic Widmer für die Übersetzung des Handbuchs ins Deutsche. Fehler oder Verbesserungsvorschläge bzgl. des deutschen Handbuchs bitte an das Übersetzungsteam richten, welches unter handbuch@gmx.ch oder https://amiga-resistance.info erreicht werden kann.

Wenn Sie mich kontaktieren möchten, können Sie entweder eine E-Mail an andreas@airsoftsoftwair.de schicken oder nutzen Sie das Kontaktformular unter http://www.hollywood-mal.com.

2.2 Häufig gestellte Fragen

In diesem Abschnitt werden einige häufig gestellte Fragen behandelt. Bitte lesen Sie diese zuerst, bevor Sie im Forum nachfragen, da Ihr Problem möglicherweise hier aufgeführt wurde.

F: Warum gibt es keine AROS-Version von hURL?

A: AmiSSL v4, das von hURL benötigt wird, ist für AROS noch nicht verfügbar, weshalb hURL derzeit nicht auf AROS portiert werden kann.

Q: Warum ist hURL unter MorphOS langsam?

A: AmiSSL v4 ist derzeit nicht in einer nativen PPC-Version für MorphOS verfügbar, was bedeutet, dass Sie die 68k-Version von AmiSSL v4 verwenden müssen, wenn Sie hURL auf MorphOS verwenden möchten. Dies erklärt, warum hURL auf MorphOS langsamer sein kann als auf AmigaOS 4, da AmiSSL V4 in einer PPC-nativen Version für AmigaOS 4 verfügbar ist, aber nicht für MorphOS.

F: Gibt es ein Hollywood-Forum, in dem ich mit anderen Nutzern in Kontakt treten kann?

A: Ja, bitte besuchen Sie den Bereich "Community" des offiziellen Hollywood-Portals online unter http://www.hollywood-mal.com.

F: Wo kann ich um Hilfe bitten?

A: Es gibt ein lebhaftes englischsprachiges Forum auf http://forums.hollywood-mal.com. Sie können gerne mitmachen und dort Ihre Frage stellen. Ausserdem ist ein deutschsprachiges Forum vorhanden, welches Sie unter https://www.amiga-resistance.info/ erreichen können.

F: Ich habe einen Fehler gefunden.

A: Bitte schreiben Sie darüber im "Bugs" Abschnitt des englischsprachigen Forums.

2.3 Zukunft

Hier sind einige Dinge, die auf meiner To-Do-Liste stehen:

– weitere Beispiele hinzufügen

Zögern Sie nicht, mich zu kontaktieren, wenn hURL eine bestimmte Funktion fehlt, die für Ihr Projekt wichtig ist.

2.4 Geschichte

Bitte beachten Sie die auf englisch verfasste Datei `history.txt` für ein vollständiges Änderungsprotokoll von hURL.

3 Verwendung von hURL

3.1 Übersicht

Es gibt zwei verschiedene Möglichkeiten, hURL zu verwenden: Sie können entweder direkt über eine Low-Level-Schnittstelle auf die Curl-API zugreifen oder die High-Level-Schnittstelle von hURL verwenden, die einige der Funktionen von Curl auf Standard Hollywood-Funktionen abbildet.

Die Verwendung der High-Level-Schnittstelle ist wirklich einfach und erweitert Hollywood-Funktionen wie `DownloadFile()` oder `UploadFile()`, um mit Curl zu arbeiten, so dass sie z.B. SSL/TLS verwenden können. Wenn Sie nur Dateien von/zu HTTP(S) herunterladen oder hochladen möchten und keine fein abgestimmte Kontrolle darüber benötigen, wie die Übertragung durchgeführt wird, ist die High-Level-Schnittstelle der richtige Weg für Sie.

Die Low-Level-Schnittstelle, d.h. der direkte Zugriff auf die API von curl, ist nützlich, wenn Sie eine feinere Kontrolle über den Transfer benötigen. Die Low-Level-Schnittstelle ermöglicht es Ihnen, alle Arten von Optionen in hURL zu konfigurieren und gewährt den Zugriff auf alle erweiterten Funktionen von curl, so dass Sie die Kontrolle über die Verwaltung von Übertragungen sorgfältig übernehmen können.

3.2 Verwendung der High-Level-Schnittstelle

Die Verwendung der High-Level-Schnittstelle von hURL ist denkbar einfach. Sie wird hauptsächlich verwendet, um Hollywoods Befehle `DownloadFile()` und `UploadFile()` zu erweitern, um SSL/TLS-Verbindungen zu unterstützen, die Hollywood selbst nicht unterstützt. Um eine Datei über eine SSL/TLS-Verbindung mit hURL über die High-Level-Schnittstelle herunterzuladen, gehen Sie einfach wie folgt vor:

```
@REQUIRE "hurl"
url$ = "https://www.paypal.com/"
DownloadFile(url$, {File = "index.html", Adapter = "hurl"})
```

Der obige Code lädt die Hauptseite von `https://www.paypal.com/` herunter und speichert sie als `index.html`.

Indem Sie `hurl` im Tag `Adapter` übergeben, sagen Sie `DownloadFile()`, dass hURL den Download durchführen soll. Das gleiche ist mit `UploadFile()` und `OpenConnection()` möglich. Wenn Sie für diese Befehle den Tag `Adapter` auf `hurl` setzen, wird die Verbindung automatisch von hURL verwaltet, so dass Sie beispielsweise SSL/TLS-Verschlüsselung verwenden können.

Hollywoods Befehle `DownloadFile()`, `UploadFile()` und `OpenConnection()` haben auch einen Tag `SSL`, den Sie auf `True` setzen können, um hURL anzuweisen, eine Verbindung über SSL/TLS zu erzwingen. Dies ist normalerweise nicht notwendig, wenn Schemata wie `https://` oder `ftps://` übergeben werden, kann aber für benutzerdefinierte Verbindungen nützlich sein.

3.3 Verwendung der Low-Level-Schnittstelle

Die Verwendung der Low-Level-Schnittstelle von hURL ist schwieriger als die Verwendung der High-Level-Schnittstelle, da Sie damit direkt auf die APIs von curl zugreifen können.

Das bedeutet, dass Sie sich zuerst mit der API von curl vertraut machen sollten, damit Sie wissen, wie sie gestaltet ist und wie sie ihren Zwecken dienen kann.

Grundsätzlich beinhaltet die direkte Verwendung einer Curl-API die folgenden drei Schritte:

1. Erstellen Sie einen objektbasierten curl-Handle, z.B. einen Curl-Easy-Handle.
2. Erledigen Sie etwas mit dem Handle, z.B. eine Übertragung starten.
3. Schliessen Sie den Handle.

Um beispielsweise eine Datei mit der Easy-Schnittstelle von curl zu übertragen, können Sie den folgenden Code verwenden:

```
@REQUIRE "hurl"

; diese Funktion wird aufgerufen, wenn es neue Daten gibt
Function p_WriteData(data$)
   WriteBytes(1, data$)
EndFunction

OpenFile(1, "test.html", #MODE_WRITE)

; Easy-Objekt erstellen und konfigurieren
e = hurl.Easy({URL = "https://www.paypal.com/", WriteFunction =
        p_WriteData, FollowLocation = True})

; Daten übertragen
e:Perform()

; Easy-Objekt schliessen
e:Close()
CloseFile(1)
```

Der obige Code lädt die Seite unter `https://www.paypal.com/` herunter und speichert sie mit Hilfe der Easy-Schnittstelle von curl in der Datei `test.html`. Dies geschieht, indem zuerst ein Easy-Objekt mit `hurl.Easy()` erstellt und dann die Optionen `#CURLOPT_URL`, `#CURLOPT_WRITEFUNCTION` und `#CURLOPT_FOLLOWLOCATION` auf diesem Easy-Objekt gesetzt werden.

Wie oben gezeigt, können die curl-Optionen direkt beim Erstellen von curl-Objekten eingestellt werden. Alternativ können Sie auch ein leeres curl-Objekt erstellen und die Optionen anschließend so einstellen:

```
e = hurl.Easy()
e:SetOpt_URL("https://www.paypal.com/")
e:SetOpt_WriteFunction(p_WriteData)
e:SetOpt_FollowLocation(True)
```

Dieser Code hat die gleiche Funktion wie der Code im entsprechenden Abschnitt oben. Der einzige Unterschied besteht darin, dass die Optionen nicht zur Erstellungszeit, sondern nach der Erstellung festgelegt werden. Darüber hinaus können Sie nach der Objekterstellung auch mehrere Optionen auf einmal einstellen. Hier ist eine weitere Alternative für die beiden obigen Codeausschnitte:

```
e = hurl.Easy()
```

```
e:SetOpt({URL = "https://www.paypal.com/", WriteFunction = p_WriteData,
            FollowLocation = True})
```

Schließlich können Sie auch `easy:SetOpt()` verwenden, um Curl-Optionen für Easy-Curl-Zugriffe festzulegen. Es gibt also sogar eine vierte Möglichkeit, das zu tun, was die obigen Code-Ausschnitte tun. Hier ist sie:

```
e = hurl.Easy()
e:SetOpt(#CURLOPT_URL, "https://www.paypal.com/")
e:SetOpt(#CURLOPT_WRITEFUNCTION, p_WriteData)
e:SetOpt(#CURLOPT_FOLLOWLOCATION, True)
```

Weitere Informationen über die Funktion der verschiedenen Optionen von curl finden Sie in den nun folgenden Kapiteln.

4 Allgemeine Befehle

4.1 hurl.Easy

BEZEICHNUNG

hurl.Easy – startet eine libcurl-Easy-Sitzung

ÜBERSICHT

```
handle = hurl.Easy([table])
```

BESCHREIBUNG

Dieser Befehl muss der erste Befehl sein, der aufgerufen wird und der ihnen einen curl-Easy-Handle zurück gibt, den Sie als Eingabe für andere Befehle in der Easy-Schnittstelle verwenden müssen. Wenn die Operation abgeschlossen ist, muss ein entsprechender Aufruf von `easy:Close()` folgen.

Das optionale Argument `table` ermöglicht es Ihnen, zusätzliche Optionen für das Easy-Objekt festzulegen. Es ist möglich, hier alle Optionen zu verwenden, die auch separat mit dem Befehl `easy:SetOpt()` eingestellt werden können. Um beispielsweise `#CURLOPT_URL`, `#CURLOPT_VERBOSE` und `#CURLOPT_FOLLOWLOCATION` zum Zeitpunkt der Erstellung festzulegen, gehen Sie wie folgt vor:

```
e = hurl.Easy({URL = "http://www.hollywood-mal.com",
        Verbose = True, FollowLocation = True})
```

Dieser Code funktioniert genauso wie:

```
e = hurl.Easy()
e:SetOpt_URL("http://www.hollywood-mal.com")
e:SetOpt_Verbose(True)
e:SetOpt_FollowLocation(True)
```

Alternativ können Sie auch `easy:SetOpt()` verwenden, um diese Optionen einzustellen:

```
e = hurl.Easy()
e:SetOpt(#CURLOPT_URL, "http://www.hollywood-mal.com")
e:SetOpt(#CURLOPT_VERBOSE, True)
e:SetOpt(#CURLOPT_FOLLOWLOCATION, True)
```

Alle oben genannten Codeausschnitte machen genau das Gleiche.

EINGABEN

`table` optional: Tabellenargument mit weiteren Optionen (siehe oben)

RÜCKGABEWERTE

`handle` curl-Easy-Handler

4.2 hurl.Form

BEZEICHNUNG

hurl.Form – erstellt ein HTTP-Multipart/Formdata-Objekt

ÜBERSICHT

```
handle = hurl.Form()
```

BESCHREIBUNG

Dieser Befehl erstellt ein HTTP-Multipart/Formdata-Objekt und gibt es zurück. Sie können dann Befehle wie `form:AddFile()`, `form:AddBuffer()` oder `form:AddContent()` verwenden, um es mit Inhalt zu füllen. Wenn die Operation abgeschlossen ist, muss ein entsprechender Aufruf von `form:Free()` folgen.

EINGABEN

keine

RÜCKGABEWERTE

`handle` HTTP-Multipart/Formdata-Objekt

4.3 hurl.Multi

BEZEICHNUNG

hurl.Multi – erstellt einen Multi-Handle

ÜBERSICHT

```
handle = hurl.Multi([table])
```

BESCHREIBUNG

Dieser Befehl gibt einen Curl-Multi-Handle zurück, der als Eingabe für alle anderen Multibefehle verwendet wird, die manchmal an einigen Stellen in der Dokumentation als Multi-Handle bezeichnet werden. Wenn die Operation abgeschlossen ist, muss ein entsprechender Aufruf von `multi:Close()` folgen.

Das optionale Argument `table` ermöglicht es Ihnen, zusätzliche Optionen für das Multi-Objekt festzulegen. Es ist möglich, hier alle Optionen zu verwenden, die auch separat mit dem Befehl `multi:SetOpt()` eingestellt werden können. Siehe Abschnitt 4.1 [hurl:Easy], Seite 11, für ein Beispiel.

EINGABEN

`table` optional: Tabellenargument mit weiteren Optionen

RÜCKGABEWERTE

`handle` curl-Multi-Handle

4.4 hurl.Share

BEZEICHNUNG

hurl.Share – erstellt ein Share-Objekt

ÜBERSICHT

```
handle = hurl.Share([table])
```

BESCHREIBUNG

Dieser Befehl gibt einen curl-Share-Handle zurück, der als Eingabe für alle anderen Share-Befehle verwendet wird, die an einigen Stellen in der Dokumentation manchmal als Share-Handle bezeichnet werden. Wenn die Operation abgeschlossen ist, muss ein entsprechender Aufruf von `share:Close()` folgen.

Dieser Share-Handle ist das, was Sie mit der Option `#CURLOPT_SHARE` mit dem Befehl `easy:SetOpt()` an curl übergeben, damit dieser spezielle Curl-Handle die Daten in dieser Share verwendet.

Das optionale Argument `table` ermöglicht es Ihnen, zusätzliche Optionen für das Share-Objekt festzulegen. Es ist möglich, hier alle Optionen zu verwenden, die auch separat mit dem Befehl `share:SetOpt()` eingestellt werden können. Siehe Abschnitt 4.1 [hurl:Easy], Seite 11, für ein Beispiel.

EINGABEN

`table` optional: Tabellenargument mit weiteren Optionen

RÜCKGABEWERTE

`handle` curl-Share-Handle

4.5 hurl.Version

BEZEICHNUNG

hurl.Version – gibt die libcurl-Version als Zeichenkette zurück

ÜBERSICHT

```
v$ = hurl.Version()
```

BESCHREIBUNG

Liefert eine vom Menschen lesbare Zeichenkette mit der Versionsnummer von libcurl und einigen seiner wichtigen Komponenten (wie OpenSSL-Version).

Wir empfehlen die Verwendung des Befehls `hurl.VersionInfo()`!

EINGABEN

keine

RÜCKGABEWERTE

`v$` libcurl-Versions-Zeichenkette

4.6 hurl.VersionInfo

BEZEICHNUNG

hurl.VersionInfo – gibt zur Laufzeit Informationen der libcurl-Version zurück

ÜBERSICHT

```
t = hurl.VersionInfo()
```

BESCHREIBUNG

Dieser Befehl gibt detaillierte Informationen über die libcurl-Version zur Laufzeit zurück.

Das Tabellenargument `t` enthält die folgenden Felder:

`Version:` Eine ASCII-Zeichenkette für die libcurl-Version.

`VersionNum:`
Eine 24-Bit-Zahl der Versionsnummer, die so erstellt wurde: <8 Bits Hauptnummer> | <8 bits Nebennummer> | <8 bits Revisionsnummer>. Die Version 7.9.8 wird daher als 0x070908 zurückgegeben.

`Host:` Eine ASCII-Zeichenkette, die anzeigt, für welche Hostinformationen diese libcurl erstellt, welche von einem Konfigurationsskript erkannt oder von der Buildumgebung festgelegt wurde.

`Features:`
Dies ist eine Tabelle, die die folgenden booleschen Felder enthält, die alle entweder auf `True` oder `False` gesetzt sind. Dies hängt davon ab, ob die jeweilige Funktion verfügbar ist oder nicht.

`IPV6:` Unterstützt IPv6

`Kerberos4:`
Unterstützt Kerberos V4 (bei der Verwendung von FTP)

`Kerberos5:`
Unterstützt Kerberos V5 Authentifizierung für FTP, IMAP, POP3, SMTP und SOCKSv5 proxy (Hinzugefügt in 7.40.0)

`SSL:` Unterstützt SSL (HTTPS/FTPS) (Hinzugefügt in 7.10)

`Libz:` Unterstützt HTTP-Deflate mit libz (Hinzugefügt in 7.10)

`NTLM:` Unterstützt HTTP NTLM (Hinzugefügt in 7.10.6)

`GSSNegotiate:`
Unterstützt HTTP GSS-Negotiation (Hinzugefügt in 7.10.6)

`Debug:` libcurl wurde mit Debug-Fähigkeiten entwickelt (Hinzugefügt in 7.10.6)

`CurlDebug:`
libcurl wurde mit Speicher-Tracking-Debug-Funktionen entwickelt. Dies ist vor allem für Libcurl-Hacker von Interesse. (Hinzugefügt in 7.19.6)

`AsynchDNS:`
libcurl wurde mit Unterstützung für asynchrone Namensauflösung (asynchronous name lookups) entwickelt, was genauere Timeouts (auch unter Windows) und weniger Blockaden bei der Verwendung der Multi-Schnittstelle ermöglicht. (Hinzugefügt in 7.10.7)

`SPNEGO:` libcurl wurde mit Unterstützung für die SPNEGO-Authentifizierung (Simple and Protected GSS-API Negotiation Mechanism, definiert inRFC 2478.) entwickelt. (Hinzugefügt in 7.10.8)

`LargeFile:`
libcurl wurde mit Unterstützung für große Dateien entwickelt. (Hinzugefügt in 7.11.1)

`IDN:` libcurl wurde mit Unterstützung von IDNA, Domainnamen mit internationalen Buchstaben, entwickelt. (Hinzugefügt in 7.12.0)

`SSPI:` libcurl wurde mit Unterstützung von SSPI erstellt. Dies ist nur unter Windows verfügbar und ermöglicht es libcurl, von Windows bereitgestellte Funktionen für Kerberos, NTLM, SPNEGO und Digest-Authentifizierung zu verwenden. Es erlaubt libcurl auch, die aktuellen Benutzer-Anmeldeinformationen zu verwenden, ohne dass die App sie weitergeben muss. (Hinzugefügt in 7.13.2)

`GSSAPI:` libcurl wurde mit Unterstützung von GSS-API erstellt. Dadurch nutzt libcurl die bereitgestellten Funktionen für die Kerberos- und SPNEGO-Authentifizierung. Es erlaubt libcurl auch, die aktuellen Benutzer-Anmeldeinformationen zu verwenden, ohne dass die App sie weitergeben muss. (Hinzugefügt in 7.38.0)

`CONV:` libcurl wurde mit Unterstützung für Zeichenkonvertierungen erstellt, wie sie von den Callbacks #CURLOPT_CONV_* bereitgestellt werden. (Hinzugefügt in 7.15.4)

`TLSAuthSRP:`
libcurl wurde mit Unterstützung von TLS-SRP erstellt. (Hinzugefügt in 7.21.4)

`NTLM_WB:` libcurl wurde mit Unterstützung der NTLM-Delegation an einen Winbind Helper erstellt. (Hinzugefügt in 7.22.0)

`HTTP2:` libcurl wurde mit Unterstützung für HTTP2 erstellt. (Hinzugefügt in 7.33.0)

`HTTPSProxy:`
libcurl wurde mit Unterstützung für HTTPS-Proxy entwickelt.(Hinzugefügt in 7.52.0)

`SSLVersion:`
Eine ASCII-Zeichenkette für den Namen der TLS-Bibliothek + verwendete Version. Zum Beispiel "Schannel", "SecureTransport" oder "OpenSSL/1.1.0g".

`SSLVersionNum:`
Immer 0.

`LibzVersion:`
Eine ASCII-Zeichenkette (es gibt keine numerische Version).

`Protocols:`
Dies wird auf eine Tabelle von Zeichenketten gesetzt, die die Namensprotokolle enthält, die libcurl unterstützt (mit Kleinbuchstaben). Die Protokollnamen sind gleich, wie die in URLs verwendet werden.

EINGABEN

keine

RÜCKGABEWERTE

`t` Tabelle mit Informationen über die libcurl-Version

5 Easy-Methoden

5.1 easy:Close

BEZEICHNUNG

easy:Close – beendet einen libcurl-Easy-Handle

ÜBERSICHT

```
easy:Close()
```

BESCHREIBUNG

Dieser Befehl muss der letzte Befehl sein, die eine Easy-Sitzung aufruft. Er ist das Gegenteil von dem Befehl `hurl.Easy()` und muss mit dem gleichen Handle wie die Eingabe aufgerufen werden, der der Befehl `hurl.Easy()` zurück gibt.

Dies könnte alle Verbindungen schließen, die dieser handle verwendet hat und möglicherweise bis jetzt offen gehalten hat - es sei denn, er wurde während der Transfers an einem Multi-Handle gehängt. Rufen Sie diesen Befehl nicht auf, wenn Sie beabsichtigen, mehr Dateien zu übertragen. Die Wiederverwendung von Handles ist ein Schlüssel zu einer guten Leistung mit libcurl.

Gelegentlich können Sie Ihren Fortschritts- oder Header-Callback aus `easy:Close()` heraus aufrufen lassen (wenn Sie das zuvor mit `easy:SetOpt()` für den Handle eingestellt haben). So wie wenn libcurl beschließt, die Verbindung zu beenden und das Protokoll von einer Art ist, die eine Befehls-/Antwort-Sequenz erfordert, bevor die Verbindung getrennt wird. Beispiele für solche Protokolle sind FTP, POP3 und IMAP.

Jede Verwendung des Handles, nachdem dieser Befehl aufgerufen und zurückgegeben wurde, ist illegal. `easy:Close()` beendet den Handle und den gesamten damit verbundenen Speicher!

EINGABEN

keine

5.2 easy:Escape

BEZEICHNUNG

easy:Escape – kodiert die Zeichenkette in die URL-Prozentdarstellung

ÜBERSICHT

```
e$ = easy:Escape(s$)
```

BESCHREIBUNG

Dieser Befehl konvertiert die angegebene Eingabezeichenkette `s$` in eine URL-Kodierung (auch Prozentkodierung genannt) und gibt diese als Zeichenkette zurück. Alle Eingabezeichen, die nicht a-z, A-Z, 0-9, '-', '.', '.', '_' oder '~' sind, werden in ihre Prozentdarstellung (%NN, wobei NN eine zweistellige hexadezimale Zahl ist) umgewandelt.

libcurl ist sich typischerweise der Zeichenkodierungen nicht bewusst und kümmert sich auch nicht um sie. `easy:Escape()` verschlüsselt die Daten byteweise in die URL-kodierte Version, ohne zu wissen oder sich darum zu kümmern, welches bestimmte Zeichen die

Anwendung oder der empfangende Server verschlüsselt und geht davon aus, dass die Daten verwendet werden.

Der Aufrufer von `easy:Escape()` muss sicherstellen, dass die an den Befehl übergebenen Daten korrekt verschlüsselt sind.

EINGABEN

`s$` Zeichenkette, die URL-kodiert wird

RÜCKGABEWERTE

`e$` Zeichenkette in Prozentdarstellung

5.3 easy:GetInfo

BEZEICHNUNG

easy:GetInfo – extrahiert Informationen aus einem Curl-Handle

ÜBERSICHT

```
info = easy:GetInfo(type)
```

BESCHREIBUNG

Fordern Sie mit diesem Befehl interne Informationen aus der Curl-Sitzung an. Das Argument `type` gibt an, welche Informationen abgerufen werden sollen. Verwenden Sie diesen Befehl NACH einer durchgeführten Übertragung, wenn Sie transferbezogene Daten erhalten möchten.

Die folgenden Typen werden derzeit für `type` unterstützt:

`#CURLINFO_APPCONNECT_TIME`
: Liefert die Zeit, bis die SSL/SSH-Verbindung abgeschlossen ist. Siehe Abschnitt 5.4 [easy:GetInfo_AppConnect_Time], Seite 22, für Details.

`#CURLINFO_CERTINFO`
: Gibt die TLS-Zertifikatskette zurück. Siehe Abschnitt 5.5 [easy:GetInfo_CertInfo], Seite 22, für Details.

`#CURLINFO_CONDITION_UNMET`
: Gibt Informationen über unerfüllte Zeitbedingung zurück. Siehe Abschnitt 5.6 [easy:GetInfo_Condition_Unmet], Seite 23, für Details.

`#CURLINFO_CONNECT_TIME`
: Liefert die Zeit bis zur Verbindung. Siehe Abschnitt 5.7 [easy:GetInfo_Connect_Time], Seite 23, für Details.

`#CURLINFO_CONTENT_LENGTH_DOWNLOAD`
: (Veraltet) Gibt die inhaltliche Länge des Downloads zurück. Siehe Abschnitt 5.8 [easy:GetInfo_Content_Length_Download], Seite 24, für Details.

`#CURLINFO_CONTENT_LENGTH_DOWNLOAD_T`
: Gibt die inhaltliche Länge des Downloads zurück. Siehe Abschnitt 5.9 [easy:GetInfo_Content_Length_Download_t], Seite 24, für Details.

`#CURLINFO_CONTENT_LENGTH_UPLOAD`
: (Veraltet) Liefert die angegebene Größe des Uploads. Siehe Abschnitt 5.10 [`easy:GetInfo_Content_Length_Upload`], Seite 25, für Details.

`#CURLINFO_CONTENT_LENGTH_UPLOAD_T`
: Liefert die angegebene Größe des Uploads. Siehe Abschnitt 5.11 [`easy:GetInfo_Content_Length_Upload_t`], Seite 25, für Details.

`#CURLINFO_CONTENT_TYPE`
: Gibt den Inhaltstyp zurück. Siehe Abschnitt 5.12 [`easy:GetInfo_Content_Type`], Seite 25, für Details.

`#CURLINFO_COOKIELIST`
: Gibt alle bekannten Cookies zurück. Siehe Abschnitt 5.13 [`easy:GetInfo_CookieList`], Seite 26, für Details.

`#CURLINFO_EFFECTIVE_URL`
: Liefert die zuletzt verwendete URL. Siehe Abschnitt 5.14 [`easy:GetInfo_Effective_URL`], Seite 26, für Details.

`#CURLINFO_FILETIME`
: Liefert die Remote-Zeit des abgerufenen Dokuments. Siehe Abschnitt 5.15 [`easy:GetInfo_FileTime`], Seite 27, für Details.

`#CURLINFO_FTP_ENTRY_PATH`
: Gibt den Eingabepfad im FTP-Server zurück. Siehe Abschnitt 5.16 [`easy:GetInfo_FTP_Entry_Path`], Seite 27, für Details.

`#CURLINFO_HEADER_SIZE`
: Liefert die Größe der abgerufenen Header. Siehe Abschnitt 5.17 [`easy:GetInfo_Header_Size`], Seite 28, für Details.

`#CURLINFO_HTTP_CONNECTCODE`
: Liefert den CONNECT-Antwortcode (Response Code). Siehe Abschnitt 5.18 [`easy:GetInfo_HTTP_ConnectCode`], Seite 28, für Details.

`#CURLINFO_HTTP_VERSION`
: Liefert die http-Version, die in der Verbindung verwendet wird. Siehe Abschnitt 5.19 [`easy:GetInfo_HTTP_Version`], Seite 28, für Details.

`#CURLINFO_HTTPAUTH_AVAIL`
: Gibt die verfügbaren HTTP-Authentifizierungsmethoden zurück. Siehe Abschnitt 5.20 [`easy:GetInfo_HTTPAuth_Avail`], Seite 29, für Details.

`#CURLINFO_LASTSOCKET`
: (Veraltet) Liefert den letzten verwendeten Socket. Siehe Abschnitt 5.21 [`easy:GetInfo_LastSocket`], Seite 29, für Details.

`#CURLINFO_LOCAL_IP`
: Ermittelt die lokale IP-Adresse der letzten Verbindung. Siehe Abschnitt 5.22 [`easy:GetInfo_Local_IP`], Seite 30, für Details.

`#CURLINFO_LOCAL_PORT`
: Ruft die neueste lokale Portnummer ab. Siehe Abschnitt 5.23 [`easy:GetInfo_Local_Port`], Seite 30, für Details.

`#CURLINFO_NAMELOOKUP_TIME`
: Liefert die Zeit der Namensauflösung. Siehe Abschnitt 5.24 [`easy:GetInfo_NameLookup_Time`], Seite 30, für Details.

`#CURLINFO_NUM_CONNECTS`
: Liefert die Anzahl der erstellten Verbindungen. Siehe Abschnitt 5.25 [`easy:GetInfo_Num_Connects`], Seite 31, für Details.

`#CURLINFO_OS_ERRNO`
: Liefert die errno-Nummer vom letzten Verbindungsabbruch. Siehe Abschnitt 5.26 [`easy:GetInfo_OS_ErrNo`], Seite 31, für Details.

`#CURLINFO_PRETRANSFER_TIME`
: Liefert die Zeit bis zum Start der Dateiübertragung. Siehe Abschnitt 5.27 [`easy:GetInfo_PreTransfer_Time`], Seite 32, für Details.

`#CURLINFO_PRIMARY_IP`
: Liefert die IP-Adresse der letzten Verbindung. Siehe Abschnitt 5.28 [`easy:GetInfo_Primary_IP`], Seite 32, für Details.

`#CURLINFO_PRIMARY_PORT`
: Ermittelt die neueste Ziel-Portnummer. Siehe Abschnitt 5.29 [`easy:GetInfo_Primary_Port`], Seite 32, für Details.

`#CURLINFO_PROTOCOL`
: Ermittelt das Protokoll, das in der Verbindung verwendet wird. Siehe Abschnitt 5.30 [`easy:GetInfo_Protocol`], Seite 33, für Details.

`#CURLINFO_PROXY_SSL_VERIFYRESULT`
: Ermittelt das Ergebnis der Proxy-Zertifikatsverifizierung. Siehe Abschnitt 5.31 [`easy:GetInfo_Proxy_SSL_VerifyResult`], Seite 34, für Details.

`#CURLINFO_PROXYAUTH_AVAIL`
: Gibt die verfügbaren HTTP-Proxy-Authentifizierungsmethoden zurück. Siehe Abschnitt 5.32 [`easy:GetInfo_ProxyAuth_Avail`], Seite 34, für Details.

`#CURLINFO_REDIRECT_COUNT`
: Ermittelt die Anzahl der Umleitungungen. Siehe Abschnitt 5.33 [`easy:GetInfo_Redirect_Count`], Seite 34, für Details.

`#CURLINFO_REDIRECT_TIME`
: Liefert die Zeit für alle Umleitungungsschritte. Siehe Abschnitt 5.34 [`easy:GetInfo_Redirect_Time`], Seite 35, für Details.

`#CURLINFO_REDIRECT_URL`
: Gibt die URL zurück, zu der eine Umleitungung gehen würde. Siehe Abschnitt 5.35 [`easy:GetInfo_Redirect_URL`], Seite 35, für Details.

`#CURLINFO_REQUEST_SIZE`
: Ermittelt die Größe der gesendeten Abfrage. Siehe Abschnitt 5.36 [`easy:GetInfo_Request_Size`], Seite 36, für Details.

`#CURLINFO_RESPONSE_CODE`
: Liefert den letzten Antwortcode. Siehe Abschnitt 5.37 [easy:GetInfo_Response_Code], Seite 36, für Details.

`#CURLINFO_RTSP_CLIENT_CSEQ`
: Gibt den nächsten RTSP-Client CSeq zurück. Siehe Abschnitt 5.38 [easy:GetInfo_RTSP_Client_CSeq], Seite 37, für Details.

`#CURLINFO_RTSP_CSEQ_RECV`
: Ermittelt das zuletzt empfangene CSeq. Siehe Abschnitt 5.39 [easy:GetInfo_RTSP_CSeq_Recv], Seite 37, für Details.

`#CURLINFO_RTSP_SERVER_CSEQ`
: Ermittelt den nächsten RTSP-Server CSeq. Siehe Abschnitt 5.40 [easy:GetInfo_RTSP_Server_CSeq], Seite 37, für Details.

`#CURLINFO_RTSP_SESSION_ID`
: Ermittelt die RTSP-Sitzungs-ID. Siehe Abschnitt 5.41 [easy:GetInfo_RTSP_Session_ID], Seite 38, für Details.

`#CURLINFO_SCHEME`
: Ermittelt das URL-Schema, das in der Verbindung verwendet wird. Siehe Abschnitt 5.42 [easy:GetInfo_Scheme], Seite 38, für Details.

`#CURLINFO_SIZE_DOWNLOAD`
: (Veraltet) Gibt die Anzahl der heruntergeladenen Bytes zurück. Siehe Abschnitt 5.43 [easy:GetInfo_Size_Download], Seite 39, für Details.

`#CURLINFO_SIZE_DOWNLOAD_T`
: Gibt die Anzahl der heruntergeladenen Bytes zurück. Siehe Abschnitt 5.44 [easy:GetInfo_Size_Download_t], Seite 39, für Details.

`#CURLINFO_SIZE_UPLOAD`
: (Veraltet) Gibt die Anzahl der hochgeladenen Bytes zurück. Siehe Abschnitt 5.45 [easy:GetInfo_Size_Upload], Seite 39, für Details.

`#CURLINFO_SIZE_UPLOAD_T`
: Gibt die Anzahl der hochgeladenen Bytes zurück. Siehe Abschnitt 5.46 [easy:GetInfo_Size_Upload_t], Seite 40, für Details.

`#CURLINFO_SPEED_DOWNLOAD`
: (Veraltet) Gibt die Download-Geschwindigkeit zurück. Siehe Abschnitt 5.47 [easy:GetInfo_Speed_Download], Seite 40, für Details.

`#CURLINFO_SPEED_DOWNLOAD_T`
: Gibt die Download-Geschwindigkeit zurück. Siehe Abschnitt 5.48 [easy:GetInfo_Speed_Download_t], Seite 41, für Details.

`#CURLINFO_SPEED_UPLOAD`
: (Veraltet) Gibt die Upload-Geschwindigkeit zurück. Siehe Abschnitt 5.49 [easy:GetInfo_Speed_Upload], Seite 41, für Details.

`#CURLINFO_SPEED_UPLOAD_T`
: Gibt die Upload-Geschwindigkeit zurück. Siehe Abschnitt 5.50 [easy:GetInfo_Speed_Upload_t], Seite 41, für Details.

`#CURLINFO_SSL_ENGINES`
: Gibt eine Liste der OpenSSL-Crypto-Engines zurück. Siehe Abschnitt 5.51 [easy:GetInfo_SSL_Engines], Seite 42, für Details.

`#CURLINFO_SSL_VERIFYRESULT`
: Gibt das Ergebnis der Zertifikatsprüfung zurück. Siehe Abschnitt 5.52 [easy:GetInfo_SSL_VerifyResult], Seite 42, für Details.

`#CURLINFO_STARTTRANSFER_TIME`
: Ermittelt die Zeit, bis das erste Byte empfangen wird. Siehe Abschnitt 5.53 [easy:GetInfo_StartTransfer_Time], Seite 43, für Details.

`#CURLINFO_TOTAL_TIME`
: Gibt die Gesamtzeit der vorherigen Übertragung zurück. Siehe Abschnitt 5.54 [easy:GetInfo_Total_Time], Seite 43, für Details.

EINGABEN

`type` Typ der abzurufenden Informationen

RÜCKGABEWERTE

`info` Ausgabewert

5.4 easy:GetInfo_AppConnect_Time

BEZEICHNUNG

easy:GetInfo_AppConnect_Time – liefert die Zeit, bis die SSL/SSH-Verbindung abgeschlossen ist

ÜBERSICHT

```
timep = easy:GetInfo_AppConnect_Time()
```

BESCHREIBUNG

Die Zeit, in Sekunden, die vom Start bis zum Abschluss der SSL/SSH-Verbindung (Handshake) zum Remote-Host benötigt wurde, wird zurückgegeben. Diese Zeit liegt meist sehr nahe an der Zeit `#CURLINFO_PRETRANSFER_TIME`, mit Ausnahme von Fällen wie HTTP-Pipelining, in denen die Vorübertragungszeit aufgrund von Wartezeiten für die Pipeline verzögert werden kann.

Siehe auch die TIMES-Übersicht auf der Handbuchseite `easy:GetInfo()`.

EINGABEN

keine

RÜCKGABEWERTE

`timep` Ausgabewert

5.5 easy:GetInfo_CertInfo

BEZEICHNUNG

easy:GetInfo_CertInfo – gibt die TLS-Zertifikatskette zurück

ÜBERSICHT

```
chainp = easy:GetInfo_CertInfo()
```

BESCHREIBUNG

Gibt eine Tabelle zurück, die eine Reihe von Zeichenkettenlisten mit Informationen über die Zertifikatskette enthält, vorausgesetzt, Sie hatten bei der Anfrage `#CURLOPT_CERTINFO` aktiviert. Die Tabelle zeigt, wie viele Zertifikate sie gefunden hat und dann können Sie Informationen für jedes dieser Zertifikate extrahieren, indem Sie den Zeichenkettenlisten folgen. Die Infokette wird in einer Reihe von Daten im Format "name:content" bereitgestellt, wobei der Inhalt für die spezifischen benannten Daten bestimmt ist.

EINGABEN

keine

RÜCKGABEWERTE

`chainp` Ausgabewert

5.6 easy:GetInfo_Condition_Unmet

BEZEICHNUNG

easy:GetInfo_Condition_Unmet – gibt Informationen über unerfüllte Zeitbedingung zurück

ÜBERSICHT

```
unmet = easy:GetInfo_Condition_Unmet()
```

BESCHREIBUNG

Liefert die Nummer 1, wenn die in der vorherigen Anforderung angegebene Bedingung nicht übereinstimmt (siehe `#CURLOPT_TIMECONDITION`). Wenn Sie keine Daten im Gegenzug erhalten und dieser Befehl eine 1 zurückgibt, wissen Sie leider, dass die Bedingung nicht erfüllt wurde. Wird hingegen eine 0 geliefert, ist die Bedingung erfüllt.

EINGABEN

keine

RÜCKGABEWERTE

`unmet` Ausgabewert

5.7 easy:GetInfo_Connect_Time

BEZEICHNUNG

easy:GetInfo_Connect_Time – liefert die Zeit bis zur Verbindung

ÜBERSICHT

```
timep = easy:GetInfo_Connect_Time()
```

BESCHREIBUNG

Liefert die Gesamtzeit in Sekunden vom Start bis zum Abschluss der Verbindung zum Remote-Host (oder Proxy).

Siehe auch die TIMES-Übersicht auf der Handbuchseite `easy:GetInfo()`.

EINGABEN

keine

RÜCKGABEWERTE

`timep` Ausgabewert

5.8 easy:GetInfo_Content_Length_Download

BEZEICHNUNG

easy:GetInfo_Content_Length_Download – gibt die inhaltliche Länge des Downloads zurück (veraltet)

ÜBERSICHT

```
content_length = easy:GetInfo_Content_Length_Download()
```

BESCHREIBUNG

Liefert die Inhaltslänge des Downloads. Dies ist der Wert, der aus dem Feld Content-Length: gelesen wird. Seit 7.19.4 gibt dies -1 zurück, wenn die Größe nicht bekannt ist.

`#CURLINFO_CONTENT_LENGTH_DOWNLOAD_T` ist ein neuerer Ersatz, der einen sinnvolleren Variablentyp zurückgibt. Siehe Abschnitt 5.8 [easy:GetInfo_Content_Length_Download], Seite 24, für Details.

EINGABEN

keine

RÜCKGABEWERTE

`content_length`
Ausgabewert

5.9 easy:GetInfo_Content_Length_Download_t

BEZEICHNUNG

easy:GetInfo_Content_Length_Download_t – gibt die inhaltliche Länge des Downloads zurück

ÜBERSICHT

```
content_length = easy:GetInfo_Content_Length_Download_t()
```

BESCHREIBUNG

Liefert die Inhaltslänge des Downloads. Dies ist der Wert, der aus dem Feld Content-Length: gelesen wird. Gibt -1 zurück, wenn die Größe nicht bekannt ist.

EINGABEN

keine

RÜCKGABEWERTE

`content_length`
Ausgabewert

5.10 easy:GetInfo_Content_Length_Upload

BEZEICHNUNG

easy:GetInfo_Content_Length_Upload – liefert die angegebene Größe des Uploads (veraltet)

ÜBERSICHT

```
content_length = easy:GetInfo_Content_Length_Upload()
```

BESCHREIBUNG

Gibt die angegebene Größe des Uploads zurück. Seit 7.19.4 gibt dies -1 zurück, wenn die Größe nicht bekannt ist.

`#CURLINFO_CONTENT_LENGTH_UPLOAD_T` ist ein neuerer Ersatz, der einen sinnvolleren Variablentyp zurückgibt. Siehe Abschnitt 5.11 [easy:GetInfo_Content_Length_Upload_t], Seite 25, für Details.

EINGABEN

keine

RÜCKGABEWERTE

`content_length`
: Ausgabewert

5.11 easy:GetInfo_Content_Length_Upload_t

BEZEICHNUNG

easy:GetInfo_Content_Length_Upload_t – liefert die angegebene Größe des Uploads

ÜBERSICHT

```
content_length = easy:GetInfo_Content_Length_Upload_t()
```

BESCHREIBUNG

Gibt die angegebene Größe des Uploads zurück. Liefert -1, wenn die Größe nicht bekannt ist.

EINGABEN

keine

RÜCKGABEWERTE

`content_length`
: Ausgabewert

5.12 easy:GetInfo_Content_Type

BEZEICHNUNG

easy:GetInfo_Content_Type – gibt den Inhaltstyp zurück

ÜBERSICHT

```
ct = easy:GetInfo_Content_Type()
```

BESCHREIBUNG

Liefert den Inhaltstyp des heruntergeladenen Objekts. Dies ist der Wert, der aus dem Feld Content-Type: gelesen wird. Wenn Sie `Nil` erhalten, bedeutet dies, dass der Server keine gültige Content-Type-Header gesendet hat oder dass das verwendete Protokoll dies nicht unterstützt.

EINGABEN

keine

RÜCKGABEWERTE

`ct` Ausgabewert

5.13 easy:GetInfo_CookieList

BEZEICHNUNG

easy:GetInfo_CookieList – gibt alle bekannten Cookies zurück

ÜBERSICHT

```
cookies = easy:GetInfo_CookieList()
```

BESCHREIBUNG

Gibt eine Liste aller Cookies zurück, die Curl kennt (auch abgelaufene). Wenn es keine Cookies gibt (Cookies für den Handle wurden nicht aktiviert oder einfach keine empfangen), wird `Nil` zurückgegeben.

Seit 7.43.0 werden Cookies, die im Set-Cookie-Format ohne Domain-Namen importiert wurden, von dieser Option nicht mehr exportiert.

EINGABEN

keine

RÜCKGABEWERTE

`cookies` Ausgabewert

5.14 easy:GetInfo_Effective_URL

BEZEICHNUNG

easy:GetInfo_Effective_URL – liefert die zuletzt verwendete URL

ÜBERSICHT

```
urlp = easy:GetInfo_Effective_URL()
```

BESCHREIBUNG

Liefert die zuletzt verwendete effektive URL.

In Fällen, in denen Sie libcurl gebeten haben, Umleitungungen zu folgen, kann es sehr wohl sein, dass es nicht derselbe Wert ist, den Sie mit `#CURLOPT_URL` gesetzt haben.

EINGABEN

keine

RÜCKGABEWERTE

`urlp` Ausgabewert

5.15 easy:GetInfo_FileTime

BEZEICHNUNG

easy:GetInfo_FileTime – liefert die Remote-Zeit des abgerufenen Dokuments

ÜBERSICHT

```
timep = easy:GetInfo_FileTime()
```

BESCHREIBUNG

Liefert die Remote-Zeit des abgerufenen Dokuments (in Sekunden seit dem 1. Januar 1970 in der GMT/UTC-Zeitzone). Wenn Sie -1 erhalten, kann es aus vielen Gründen sein (es könnte unbekannt sein, der Server könnte es verstecken oder der Server unterstützt den Befehl, der die Dokumentzeit angibt, nicht usw.) und die Zeit des Dokuments ist unbekannt.

Sie müssen libcurl anweisen, diese Informationen zu sammeln, bevor die Übertragung durchgeführt wird, indem Sie die Option `#CURLOPT_FILETIME` für `easy:SetOpt()` verwenden, sonst erhalten Sie bedingungslos eine -1 zurück.

Erwägen Sie die Verwendung von `#CURLINFO_FILETIME_T`, um auf Systemen mit 32-Bit-Langen Daten über das Jahr 2038 hinaus extrahieren zu können.

EINGABEN

keine

RÜCKGABEWERTE

`timep` Ausgabewert

5.16 easy:GetInfo_FTP_Entry_Path

BEZEICHNUNG

easy:GetInfo_FTP_Entry_Path – gibt den Eingabepfad im FTP-Server zurück

ÜBERSICHT

```
path = easy:GetInfo_FTP_Entry_Path()
```

BESCHREIBUNG

Gibt eine Zeichenkette zurück, die den Pfad des Eingabepfades enthält. Das ist der anfängliche Pfad, in dem libcurl bei der Anmeldung am Remote-FTP-Server landete. Dies gibt Null zurück, wenn etwas nicht stimmt.

EINGABEN

keine

RÜCKGABEWERTE

`path` Ausgabewert

5.17 easy:GetInfo_Header_Size

BEZEICHNUNG

easy:GetInfo_Header_Size – liefert die Größe der abgerufenen Header

ÜBERSICHT

```
sizep = easy:GetInfo_Header_Size()
```

BESCHREIBUNG

Liefert die Gesamtgröße aller empfangenen Header. Gemessen in Anzahl der Bytes.

Die Summe enthält die Größe aller empfangenen Header, die durch `#CURLOPT_SUPPRESS_CONNECT_HEADERS` unterdrückt werden.

EINGABEN

keine

RÜCKGABEWERTE

`sizep` Ausgabewert

5.18 easy:GetInfo_HTTP_ConnectCode

BEZEICHNUNG

easy:GetInfo_HTTP_ConnectCode – liefert den CONNECT-Antwortcode

ÜBERSICHT

```
p = easy:GetInfo_HTTP_ConnectCode()
```

BESCHREIBUNG

Liefert den zuletzt empfangenen HTTP-Proxy-Antwortcode (Response Code) an eine `CONNECT`-Anfrage. Der zurückgegebene Wert ist Null, wenn kein solcher Antwortcode verfügbar war.

EINGABEN

keine

RÜCKGABEWERTE

`p` Ausgabewert

5.19 easy:GetInfo_HTTP_Version

BEZEICHNUNG

easy:GetInfo_HTTP_Version – liefert die http-Version, die in der Verbindung verwendet wird

ÜBERSICHT

```
p = easy:GetInfo_HTTP_Version()
```

BESCHREIBUNG

Liefert die Version, die in der letzten http-Verbindung verwendet wurde. Der zurückgegebene Wert ist `#CURL_HTTP_VERSION_1_0`, `#CURL_HTTP_VERSION_1_1`, `#CURL_HTTP_VERSION_2_0` oder 0, wenn die Version nicht ermittelt werden kann.

EINGABEN
keine

RÜCKGABEWERTE
`p` Ausgabewert

5.20 easy:GetInfo_HTTPAuth_Avail

BEZEICHNUNG
easy:GetInfo_HTTPAuth_Avail – gibt die verfügbaren HTTP-Authentifizierungsmethoden zurück

ÜBERSICHT
```
authp = easy:GetInfo_HTTPAuth_Avail()
```

BESCHREIBUNG
Gibt eine Bitmaske zurück, die die Authentifizierungsmethode(n) angibt, die gemäß der vorherigen Antwort verfügbar sind. Die Bedeutung der Bits wird in der Option `#CURLOPT_HTTPAUTH` für `easy:SetOpt()` erläutert.

EINGABEN
keine

RÜCKGABEWERTE
`authp` Ausgabewert

5.21 easy:GetInfo_LastSocket

BEZEICHNUNG
easy:GetInfo_LastSocket – liefert den letzten verwendeten Socket (veraltet)

ÜBERSICHT
```
socket = easy:GetInfo_LastSocket()
```

BESCHREIBUNG
Veraltet seit 7.45.0. (Verwenden Sie stattdessen `#CURLINFO_ACTIVESOCKET`, was allerdings in hURL 1.0 nicht unterstützt wird.)

Liefert den letzten Socket, der von dieser Curl-Sitzung verwendet wurde. Wenn der Socket nicht mehr gültig ist, wird -1 zurückgegeben. Wenn Sie mit dem Socket fertig sind, müssen Sie wie gewohnt `easy:Close()` aufrufen und libcurl wird den Socket schließen und andere Ressourcen, die mit dem Handle verbunden sind, bereinigen lassen. Dies wird typischerweise in Kombination mit `#CURLOPT_CONNECT_ONLY` verwendet.

HINWEIS: Diese API ist veraltet, da sie nicht auf win64 funktioniert, wo der SOCKET-Typ 64 Bit groß ist, während sein "Long" 32 Bit beträgt. (Verwenden Sie stattdessen, wenn möglich, `#CURLINFO_ACTIVESOCKET`. Allerdings wird `#CURLINFO_ACTIVESOCKET` in hURL 1.0 nicht unterstützt.)

EINGABEN
keine

RÜCKGABEWERTE

`socket` Ausgabewert

5.22 easy:GetInfo_Local_IP

BEZEICHNUNG

easy:GetInfo_Local_IP – ermittelt die lokale IP-Adresse der letzten Verbindung

ÜBERSICHT

```
ip = easy:GetInfo_Local_IP()
```

BESCHREIBUNG

Gibt eine Zeichenkette zurück, die die IP-Adresse der letzten Verbindung enthält, die mit diesem Curl-Handle durchgeführt wurde. Diese Zeichenkette kann IPv6 sein, wenn diese aktiviert ist.

EINGABEN

keine

RÜCKGABEWERTE

`ip` Ausgabewert

5.23 easy:GetInfo_Local_Port

BEZEICHNUNG

easy:GetInfo_Local_Port – ruft die neueste lokale Portnummer ab

ÜBERSICHT

```
portp = easy:GetInfo_Local_Port()
```

BESCHREIBUNG

Liefert die lokale (Quell-) Portnummer der letzten Verbindung, die mit diesem Curl-Handle durchgeführt wurde.

EINGABEN

keine

RÜCKGABEWERTE

`portp` Ausgabewert

5.24 easy:GetInfo_NameLookup_Time

BEZEICHNUNG

easy:GetInfo_NameLookup_Time – liefert die Zeit der Namensauflösung

ÜBERSICHT

```
timep = easy:GetInfo_NameLookup_Time()
```

BESCHREIBUNG

Liefert die Gesamtzeit in Sekunden vom Start bis zum Abschluss der Namensauflösung. Siehe auch die TIMES-Übersicht auf der Handbuchseite `easy:GetInfo()`.

EINGABEN

keine

RÜCKGABEWERTE

`timep` Ausgabewert

5.25 easy:GetInfo_Num_Connects

BEZEICHNUNG

easy:GetInfo_Num_Connects – liefert die Anzahl der erstellten Verbindungen

ÜBERSICHT

```
nump = easy:GetInfo_Num_Connects()
```

BESCHREIBUNG

Gibt zurück, wie viele neue Verbindungen libcurl erstellen musste, um den vorherigen Transfer zu erreichen (nur die erfolgreichen Verbindungen werden gezählt). In Kombination mit `#CURLINFO_REDIRECT_COUNT` können Sie feststellen, wie oft libcurl bestehende Verbindungen erfolgreich wiederverwendet hat oder nicht. Lesen Sie die Verbindungsoptionen von `easy:SetOpt()`, um zu sehen, wie libcurl versucht, dauerhafte Verbindungen herzustellen, um Zeit zu sparen.

EINGABEN

keine

RÜCKGABEWERTE

`nump` Ausgabewert

5.26 easy:GetInfo_OS_ErrNo

BEZEICHNUNG

easy:GetInfo_OS_ErrNo – liefert die errno-Nummer vom letzten Verbindungsabbruch

ÜBERSICHT

```
errnop = easy:GetInfo_OS_ErrNo()
```

BESCHREIBUNG

Liefert die Variable errno aus einem Verbindungsabbruch. Beachten Sie, dass der Wert nur bei Ausfall gesetzt wird, er wird bei einem erfolgreichen Vorgang nicht zurückgesetzt. Die Nummer ist OS- und systemspezifisch.

EINGABEN

keine

RÜCKGABEWERTE

`errnop` Ausgabewert

5.27 easy:GetInfo_PreTransfer_Time

BEZEICHNUNG

easy:GetInfo_PreTransfer_Time – liefert die Zeit bis zum Start der Dateiübertragung

ÜBERSICHT

```
timep = easy:GetInfo_PreTransfer_Time()
```

BESCHREIBUNG

Gibt die Zeit in Sekunden zurück, die vom Start bis zum Beginn der Dateiübertragung benötigt wurde. Dazu gehören alle Befehle und Verhandlungen vor der Übertragung, die spezifisch für die jeweiligen Protokolle sind. Es handelt sich nicht um das Senden der protokollspezifischen Anforderung, die eine Übertragung auslöst.

Siehe auch die TIMES-Übersicht auf der Handbuchseite `easy:GetInfo()`.

EINGABEN

keine

RÜCKGABEWERTE

`timep` Ausgabewert

5.28 easy:GetInfo_Primary_IP

BEZEICHNUNG

easy:GetInfo_Primary_IP – liefert die IP-Adresse der letzten Verbindung

ÜBERSICHT

```
ip = easy:GetInfo_Primary_IP()
```

BESCHREIBUNG

Gibt eine Zeichenkette zurück, die die IP-Adresse der letzten Verbindung enthält, die mit diesem Curl-Handle durchgeführt wurde. Diese Zeichenkette kann IPv6 sein, wenn diese aktiviert ist.

EINGABEN

keine

RÜCKGABEWERTE

`ip` Ausgabewert

5.29 easy:GetInfo_Primary_Port

BEZEICHNUNG

easy:GetInfo_Primary_Port – ermittelt die neueste Ziel-Portnummer

ÜBERSICHT

```
portp = easy:GetInfo_Primary_Port()
```

BESCHREIBUNG

Liefert den Zielport der letzten Verbindung, die mit diesem Curl-Handle durchgeführt wurde.

EINGABEN
keine

RÜCKGABEWERTE
`portp` Ausgabewert

5.30 easy:GetInfo_Protocol

BEZEICHNUNG
easy:GetInfo_Protocol – ermittelt das Protokoll, das in der Verbindung verwendet wird

ÜBERSICHT
```
p = easy:GetInfo_Protocol()
```

BESCHREIBUNG
Liefert die Version, die in der letzten http-Verbindung verwendet wurde. Der zurückgegebene Wert ist genau einer der Werte `#CURLPROTO_XXX`:

```
#CURLPROTO_DICT
#CURLPROTO_FILE
#CURLPROTO_FTP
#CURLPROTO_FTPS
#CURLPROTO_GOPHER
#CURLPROTO_HTTP
#CURLPROTO_HTTPS
#CURLPROTO_IMAP
#CURLPROTO_IMAPS
#CURLPROTO_LDAP
#CURLPROTO_LDAPS
#CURLPROTO_POP3
#CURLPROTO_POP3S
#CURLPROTO_RTMP
#CURLPROTO_RTMPE
#CURLPROTO_RTMPS
#CURLPROTO_RTMPT
#CURLPROTO_RTMPTE
#CURLPROTO_RTMPTS
#CURLPROTO_RTSP
#CURLPROTO_SCP
#CURLPROTO_SFTP
#CURLPROTO_SMB
#CURLPROTO_SMBS
#CURLPROTO_SMTP
#CURLPROTO_SMTPS
#CURLPROTO_TELNET
#CURLPROTO_TFTP
```

EINGABEN
keine

RÜCKGABEWERTE

`p` Ausgabewert

5.31 easy:GetInfo_Proxy_SSL_VerifyResult

BEZEICHNUNG

easy:GetInfo_Proxy_SSL_VerifyResult – ermittelt das Ergebnis der Proxy-Zertifikatsverifizierung

ÜBERSICHT

```
result = easy:GetInfo_Proxy_SSL_VerifyResult()
```

BESCHREIBUNG

Liefert das Ergebnis der angeforderten Zertifikatsverifizierung (mit der Option `#CURLOPT_PROXY_SSL_VERIFYPEER`). Dies wird nur für HTTPS-Proxies verwendet.

EINGABEN

keine

RÜCKGABEWERTE

`result` Ausgabewert

5.32 easy:GetInfo_ProxyAuth_Avail

BEZEICHNUNG

easy:GetInfo_ProxyAuth_Avail – gibt die verfügbaren HTTP-Proxy-Authentifizierungsmethoden zurück

ÜBERSICHT

```
authp = easy:GetInfo_ProxyAuth_Avail()
```

BESCHREIBUNG

Gibt eine Bitmaske zurück, die die Authentifizierungsmethode(n) angibt, die gemäß der vorherigen Antwort verfügbar sind. Die Bedeutung der Bits wird in der Option `#CURLOPT_PROXYAUTH` für `easy:SetOpt()` erläutert.

EINGABEN

keine

RÜCKGABEWERTE

`authp` Ausgabewert

5.33 easy:GetInfo_Redirect_Count

BEZEICHNUNG

easy:GetInfo_Redirect_Count – ermittelt die Anzahl der Umleitungungen

ÜBERSICHT

```
countp = easy:GetInfo_Redirect_Count()
```

BESCHREIBUNG
Gibt die Gesamtzahl der Umleitungungen zurück, die tatsächlich ausgeführt wurden.

EINGABEN
keine

RÜCKGABEWERTE
`countp` Ausgabewert

5.34 easy:GetInfo_Redirect_Time

BEZEICHNUNG
easy:GetInfo_Redirect_Time – liefert die Zeit für alle Umleitungungsschritte

ÜBERSICHT
```
timep = easy:GetInfo_Redirect_Time()
```

BESCHREIBUNG
Liefert die Gesamtzeit in Sekunden, die für alle Umleitungungsschritte wie Namensrecherche, Verbindung, Vorübertragung und Übertragung benötigt wurde, bevor die endgültige Transaktion gestartet wurde. `#CURLINFO_REDIRECT_TIME` enthält die komplette Ausführungszeit für mehrere Umleitungungen.

Siehe auch die TIMES-Übersicht auf der Handbuchseite `easy:GetInfo()`.

EINGABEN
keine

RÜCKGABEWERTE
`timep` Ausgabewert

5.35 easy:GetInfo_Redirect_URL

BEZEICHNUNG
easy:GetInfo_Redirect_URL – gibt die URL zurück, zu der eine Umleitungung gehen würde

ÜBERSICHT
```
urlp = easy:GetInfo_Redirect_URL()
```

BESCHREIBUNG
Liefert die URL, zu der eine Umleitungung Sie führen würde, wenn Sie `#CURLOPT_FOLLOWLOCATION` aktivieren würden. Dies kann sehr nützlich sein, wenn Sie denken, dass die Verwendung der eingebauten libcurl-Umleitungungs-Logik nicht gut genug für Sie ist, aber Sie würden es trotzdem vorziehen, die ganzen Abläufe der Ermittlung der neuen URL zu vermeiden.

Diese URL wird auch gesetzt, wenn die Begrenzung `#CURLOPT_MAXREDIRS` eine Umleitungung verhindert hat (seit 7.54.1).

EINGABEN
keine

RÜCKGABEWERTE

`urlp` Ausgabewert

5.36 easy:GetInfo_Request_Size

BEZEICHNUNG

easy:GetInfo_Request_Size – ermittelt die Größe der gesendeten Abfrage

ÜBERSICHT

```
sizep = easy:GetInfo_Request_Size()
```

BESCHREIBUNG

Liefert die Gesamtgröße der ausgegebenen Abfragen. Dies gilt bisher nur für HTTP-Abfragen. Beachten Sie, dass dies mehr als eine Abffrage sein kann, wenn `#CURLOPT_FOLLOWLOCATION` aktiviert ist.

EINGABEN

keine

RÜCKGABEWERTE

`sizep` Ausgabewert

5.37 easy:GetInfo_Response_Code

BEZEICHNUNG

easy:GetInfo_Response_Code – liefert den letzten Antwortcode

ÜBERSICHT

```
codep = easy:GetInfo_Response_Code()
```

BESCHREIBUNG

Liefert den zuletzt empfangenen HTTP-, FTP- oder SMTP-Antwortcode. Diese Option war in libcurl 7.10.7 und früher als `#CURLINFO_HTTP_CODE` bekannt. Der gespeicherte Wert ist Null, wenn kein Server-Antwortcode empfangen wurde. Beachten Sie, dass die CONNECT-Antwort eines Proxy mit `#CURLINFO_HTTP_CONNECTCODE` gelesen werden sollte und nicht mit diesem Befehl.

Die Unterstützung für SMTP-Antworten wurde in 7.25.0 hinzugefügt.

EINGABEN

keine

RÜCKGABEWERTE

`codep` Ausgabewert

5.38 easy:GetInfo_RTSP_Client_CSeq

BEZEICHNUNG
easy:GetInfo_RTSP_Client_CSeq – gibt den nächsten RTSP-Client CSeq zurück

ÜBERSICHT
```
cseq = easy:GetInfo_RTSP_Client_CSeq()
```

BESCHREIBUNG
Liefert den nächsten CSeq, der von der Anwendung verwendet wird.

EINGABEN
keine

RÜCKGABEWERTE

`cseq`	Ausgabewert

5.39 easy:GetInfo_RTSP_CSeq_Recv

BEZEICHNUNG
easy:GetInfo_RTSP_CSeq_Recv – ermittelt das zuletzt empfangene CSeq

ÜBERSICHT
```
cseq = easy:GetInfo_RTSP_CSeq_Recv()
```

BESCHREIBUNG
Liefert den zuletzt empfangenen CSeq vom Server. Wenn Ihre Anwendung auf einen `#CURLE_RTSP_CSEQ_ERROR` stößt, dann können Sie die CSeq-Unstimmigkeit durch einen Blick auf diesen Wert beheben.

EINGABEN
keine

RÜCKGABEWERTE

`cseq`	Ausgabewert

5.40 easy:GetInfo_RTSP_Server_CSeq

BEZEICHNUNG
easy:GetInfo_RTSP_Server_CSeq – ermittelt den nächsten RTSP-Server CSeq

ÜBERSICHT
```
cseq = easy:GetInfo_RTSP_Server_CSeq()
```

BESCHREIBUNG
Liefert den nächsten CSeq, der von der Anwendung erwartet wird.

Das Überwachen von serverinitiierten Abfragen ist derzeit nicht implementiert!

Anwendungen, die eine RTSP-Sitzung bei einer anderen Verbindung wieder aufnehmen möchten, sollten diese Informationen abrufen, bevor sie die aktive Verbindung schließen.

EINGABEN
keine

RÜCKGABEWERTE

`cseq` Ausgabewert

5.41 easy:GetInfo_RTSP_Session_ID

BEZEICHNUNG
easy:GetInfo_RTSP_Session_ID – ermittelt die RTSP-Sitzungs-ID

ÜBERSICHT

```
id = easy:GetInfo_RTSP_Session_ID()
```

BESCHREIBUNG
Liefert eine Zeichenkette mit der aktuellsten RTSP-Sitzungs-ID.

Anwendungen, die eine RTSP-Sitzung bei einer anderen Verbindung wieder aufnehmen möchten, sollten diese Informationen abrufen, bevor sie die aktive Verbindung schließen.

EINGABEN
keine

RÜCKGABEWERTE

`id` Ausgabewert

5.42 easy:GetInfo_Scheme

BEZEICHNUNG
easy:GetInfo_Scheme – ermittelt das URL-Schema, das in der Verbindung verwendet wird

ÜBERSICHT

```
scheme = easy:GetInfo_Scheme()
```

BESCHREIBUNG
Liefert eine Zeichenkette mit dem URL-Schema (manchmal auch Protokoll genannt), das für die letzte Verbindung mit diesem CURL-Handle verwendet wird.

EINGABEN
keine

RÜCKGABEWERTE

`scheme` Ausgabewert

5.43 easy:GetInfo_Size_Download

BEZEICHNUNG

easy:GetInfo_Size_Download – gibt die Anzahl der heruntergeladenen Bytes zurück (veraltet)

ÜBERSICHT

```
dlp = easy:GetInfo_Size_Download()
```

BESCHREIBUNG

Liefert die Gesamtzahl der Bytes, die heruntergeladen wurden. Der Betrag gilt nur für die letzte Sendung und wird bei jeder neuen Sendung erneut zurückgesetzt. Diese zählt die tatsächlichen Nutzlastdaten, die auch als Rohdaten bezeichnet werden. Alle Meta- und Header-Daten sind ausgeschlossen und werden in dieser Zahl nicht gezählt.

`#CURLINFO_SIZE_DOWNLOAD_T` ist ein neuerer Ersatz, der einen sinnvolleren Variablentyp zurückgibt. Siehe Abschnitt 5.44 [easy:GetInfo_Size_Download_t], Seite 39, für Details.

EINGABEN

keine

RÜCKGABEWERTE

`dlp`	Ausgabewert

5.44 easy:GetInfo_Size_Download_t

BEZEICHNUNG

easy:GetInfo_Size_Download_t – gibt die Anzahl der heruntergeladenen Bytes zurück

ÜBERSICHT

```
dlp = easy:GetInfo_Size_Download_t()
```

BESCHREIBUNG

Liefert die Gesamtzahl der Bytes, die heruntergeladen wurden. Der Betrag gilt nur für die letzte Sendung und wird bei jeder neuen Sendung erneut zurückgesetzt. Diese zählt die tatsächlichen Nutzlastdaten, die auch als Rohdaten bezeichnet werden. Alle Meta- und Header-Daten sind ausgeschlossen und werden in dieser Zahl nicht gezählt.

EINGABEN

keine

RÜCKGABEWERTE

`dlp`	Ausgabewert

5.45 easy:GetInfo_Size_Upload

BEZEICHNUNG

easy:GetInfo_Size_Upload – gibt die Anzahl der hochgeladenen Bytes zurück (veraltet)

ÜBERSICHT

```
uploadp = easy:GetInfo_Size_Upload()
```

BESCHREIBUNG

Liefert die Gesamtzahl der Bytes, die hochgeladen wurden.

`#CURLINFO_SIZE_UPLOAD_T` ist ein neuerer Ersatz, der einen sinnvolleren Variablentyp zurückgibt. Siehe Abschnitt 5.46 [easy:GetInfo_Size_Upload_t], Seite 40, für Details.

EINGABEN

keine

RÜCKGABEWERTE

`uploadp` Ausgabewert

5.46 easy:GetInfo_Size_Upload_t

BEZEICHNUNG

easy:GetInfo_Size_Upload_t – gibt die Anzahl der hochgeladenen Bytes zurück

ÜBERSICHT

```
uploadp = easy:GetInfo_Size_Upload_t()
```

BESCHREIBUNG

Liefert die Gesamtzahl der Bytes, die hochgeladen wurden.

EINGABEN

keine

RÜCKGABEWERTE

`uploadp` Ausgabewert

5.47 easy:GetInfo_Speed_Download

BEZEICHNUNG

easy:GetInfo_Speed_Download – gibt die Download-Geschwindigkeit zurück (veraltet)

ÜBERSICHT

```
speed = easy:GetInfo_Speed_Download()
```

BESCHREIBUNG

Liefert die durchschnittliche Download-Geschwindigkeit, die für den gesamten Download gemessen wurde. Gemessen wird in Bytes/Sekunde.

`#CURLINFO_SPEED_DOWNLOAD_T` ist ein neuerer Ersatz, der einen sinnvolleren Variablentyp zurückgibt. Siehe Abschnitt 5.48 [easy:GetInfo_Speed_Download_t], Seite 41, für Details.

EINGABEN

keine

RÜCKGABEWERTE

`speed` Ausgabewert

5.48 easy:GetInfo_Speed_Download_t

BEZEICHNUNG
easy:GetInfo_Speed_Download_t – gibt die Download-Geschwindigkeit zurück

ÜBERSICHT
```
speed = easy:GetInfo_Speed_Download_t()
```

BESCHREIBUNG
Liefert die durchschnittliche Download-Geschwindigkeit, die für den gesamten Download gemessen wurde. Gemessen wird in Bytes/Sekunde.

EINGABEN
keine

RÜCKGABEWERTE
`speed` Ausgabewert

5.49 easy:GetInfo_Speed_Upload

BEZEICHNUNG
easy:GetInfo_Speed_Upload – gibt die Upload-Geschwindigkeit zurück (veraltet)

ÜBERSICHT
```
speed = easy:GetInfo_Speed_Upload()
```

BESCHREIBUNG
Liefert die durchschnittliche Upload-Geschwindigkeit, die für den gesamten Upload gemessen wurde. Gemessen wird in Bytes/Sekunde.

`#CURLINFO_SPEED_UPLOAD_T` ist ein neuerer Ersatz, der einen sinnvolleren Variablentyp zurückgibt.

EINGABEN
keine

RÜCKGABEWERTE
`speed` Ausgabewert

5.50 easy:GetInfo_Speed_Upload_t

BEZEICHNUNG
easy:GetInfo_Speed_Upload_t – gibt die Upload-Geschwindigkeit zurück

ÜBERSICHT
```
speed = easy:GetInfo_Speed_Upload_t()
```

BESCHREIBUNG
Liefert die durchschnittliche Upload-Geschwindigkeit, die für den gesamten Upload gemessen wurde. Gemessen wird in Bytes/Sekunde.

EINGABEN
keine

RÜCKGABEWERTE

`speed` Ausgabewert

5.51 easy:GetInfo_SSL_Engines

BEZEICHNUNG
easy:GetInfo_SSL_Engines – gibt eine Liste der OpenSSL-Crypto-Engines zurück

ÜBERSICHT

```
engine_list = easy:GetInfo_SSL_Engines()
```

BESCHREIBUNG
Liefert eine Tabelle mit einer Liste der unterstützten OpenSSL-Kryptoverfahren. Beachten Sie, dass diese Verfahren normalerweise in separaten dynamischen Bibliotheken implementiert sind. Daher sind möglicherweise nicht alle zurückgegebenen Verfahren zur Laufzeit verfügbar.

EINGABEN
keine

RÜCKGABEWERTE

`engine_list`
Ausgabewert

5.52 easy:GetInfo_SSL_VerifyResult

BEZEICHNUNG
easy:GetInfo_SSL_VerifyResult – gibt das Ergebnis der Zertifikatsprüfung zurück

ÜBERSICHT

```
result = easy:GetInfo_SSL_VerifyResult()
```

BESCHREIBUNG
Liefert das Ergebnis der Server-SSL-Zertifikatsverifizierung, die (mit der Option `#CURLOPT_SSL_VERIFYPEER`) angefordert wurde.

0 ist ein positives Ergebnis. Nicht Null ist ein Fehler.

EINGABEN
keine

RÜCKGABEWERTE

`result` Ausgabewert

5.53 easy:GetInfo_StartTransfer_Time

BEZEICHNUNG

easy:GetInfo_StartTransfer_Time – ermittelt die Zeit, bis das erste Byte empfangen wird

ÜBERSICHT

```
timep = easy:GetInfo_StartTransfer_Time()
```

BESCHREIBUNG

Liefert die Zeit in Sekunden, die vom Start bis zum Empfang des ersten Bytes durch libcurl benötigt wurde. Dazu gehört `#CURLINFO_PRETRANSFER_TIME` und auch die Zeit, die der Server für die Berechnung des Ergebnisses benötigt.

Siehe auch die TIMES-Übersicht auf der Handbuchseite `easy:GetInfo()`.

EINGABEN

keine

RÜCKGABEWERTE

`timep` Ausgabewert

5.54 easy:GetInfo_Total_Time

BEZEICHNUNG

easy:GetInfo_Total_Time – gibt die Gesamtzeit der vorherigen Übertragung zurück

ÜBERSICHT

```
timep = easy:GetInfo_Total_Time()
```

BESCHREIBUNG

Liefert die Gesamtzeit in Sekunden für die vorherige Übertragung, einschließlich Namensauflösung, TCP-Verbindung usw. Das Ergebnis repräsentiert die Zeit in Sekunden, einschließlich Bruchteilen.

Siehe auch die TIMES-Übersicht auf der Handbuchseite `easy:GetInfo()`.

EINGABEN

keine

RÜCKGABEWERTE

`timep` Ausgabewert

5.55 easy:Pause

BEZEICHNUNG

easy:Pause – pausiert eine Verbindung und setzt sie wieder fort

ÜBERSICHT

```
easy:Pause(bitmask)
```

BESCHREIBUNG

Mit diesem Befehl können Sie eine laufende Verbindung explizit markieren, um angehalten zu werden und Sie können die zuvor angehaltene Verbindung wieder aufnehmen.

Eine Verbindung kann mithilfe von diesem Befehl oder indem die Lese- oder Schreib-Callbacks den richtigen Rückkehrcode (`#CURL_READFUNC_PAUSE` und `#CURL_WRITEFUNC_PAUSE`) zurückgeben, unterbrochen werden.

Ein Schreib-Callback, der Pausen an die Bibliothek zurückgibt, signalisiert, dass er sich überhaupt nicht um die Daten kümmern konnte und dass die Daten dann wieder an den Callback übergeben werden, wenn das Schreiben später nicht unterbrochen wird.

Auch wenn es sich verlockend anfühlen mag, seien Sie vorsichtig und beachten Sie, dass Sie diesen Befehl nicht von einem anderen Thread aus aufrufen können. Um die Pause aufzuheben, können Sie ihn zum Beispiel vom Fortschritts-Callback (`#CURLOPT_PROGRESSFUNCTION`) aus aufrufen, der mindestens einmal pro Sekunde aufgerufen wird, selbst wenn die Verbindung angehalten wird.

Wenn dieser Befehl aufgerufen wird, um das Lesen zu unterbrechen, ist die Wahrscheinlichkeit groß, dass Sie Ihren Schreib-Callback aufrufen lassen, bevor dieser Befehl beendet ist.

Das Argument `bitmask` ist ein Satz von Bits, der den neuen Zustand der Verbindung setzt. Die folgenden Bits können verwendet werden:

`#CURLPAUSE_RECV`
: Pausiert den Empfang von Daten. Es werden keine Daten über diese Verbindung mehr empfangen, bis dieser Befehl ohne dieses Bit-Set erneut aufgerufen wird. Daher wird der Schreib-Callback (`#CURLOPT_WRITEFUNCTION`) nicht aufgerufen.

`#CURLPAUSE_SEND`
: Pausiert das Senden von Daten. Es werden keine Daten über diese Verbindung gesendet, bis dieser Befehl ohne dieses Bit-Set erneut aufgerufen wird. Daher wird der Lese-Callback (`#CURLOPT_READFUNCTION`) nicht aufgerufen.

`#CURLPAUSE_ALL`
: Pausiert das Senden und den Empfang von Daten.

`#CURLPAUSE_CONT`
: Setzt das Senden und den Empfang wieder fort.

EINGABEN

`bitmask` gewünschter neuer Zustand der Verbindung

5.56 easy:Perform

BEZEICHNUNG

easy:Perform – führt eine blockierende Dateiübertragung durch

ÜBERSICHT

```
easy:Perform()
```

BESCHREIBUNG

Rufen Sie diesen Befehl nach `hurl.Easy()` auf und alle `easy:SetOpt()` Aufrufe werden ausgeführt und führen die Übertragung wie in den Optionen beschrieben durch. Es muss mit dem gleichen Easy-Handle wie die Eingabe aufgerufen werden, der `hurl.Easy()` Aufruf zurückgegeben hat.

`easy:Perform()` führt die gesamte Dateiübertragung blockierend aus und kehrt nach Abschluss oder bei Fehlschlagen zurück. Für die nicht blockierende Dateiübertragung siehe `multi:Perform()`.

Sie können beliebig viele Aufrufe von `easy:Perform()` durchführen und dabei den gleichen Easy-Handle verwenden. Wenn Sie beabsichtigen, mehr als eine Datei zu übertragen, wird Ihnen sogar empfohlen, dies zu tun. libcurl wird dann versuchen, die gleiche Verbindung für die folgenden Übertragungen wiederzuverwenden, was die Operationen schneller, weniger CPU-intensiv und mit weniger Netzwerkressourcen erledigt. Beachten Sie nur, dass Sie `easy:SetOpt()` zwischen den Aufrufen verwenden müssen, um Optionen für die folgenden `easy:Perform()` festzulegen.

Sie dürfen diesen Befehl niemals gleichzeitig von zwei Stellen aus mit dem gleichen Easy-Handler aufrufen. Lassen Sie den Befehl zuerst zurückkehren, bevor Sie ihn ein anderes Mal aufrufen. Wenn Sie parallele Transfers wünschen, müssen Sie mehrere curl-Easy-Handle verwenden.

Wenn der Easy-Handle zu einem Multi-Handle hinzugefügt wird, kann er nicht von `easy:Perform()` verwendet werden.

EINGABEN

keine

5.57 easy:Recv

BEZEICHNUNG

easy:Recv – empfängt Rohdaten über eine Easy-Verbindung

ÜBERSICHT

```
data$, n = easy:Recv(len)
```

BESCHREIBUNG

Dieser Befehl empfängt Rohdaten von der hergestellten Verbindung. Sie können ihn zusammen mit `easy:Send()` verwenden, um eigene Protokolle mit libcurl zu implementieren. Diese Funktionalität kann besonders nützlich sein, wenn Sie Proxies und/oder SSL-Verschlüsselung verwenden: libcurl übernimmt die Proxy-Verhandlung und den Verbindungsaufbau. Die Anzahl der zu empfangenden Bytes müssen Sie in `len` übergeben.

Um die Verbindung herzustellen, setzen Sie die Option `#CURLOPT_CONNECT_ONLY`, bevor Sie `easy:Perform()` oder `multi:Perform()` aufrufen. Beachten Sie, dass `easy:Recv()` bei Verbindungen, die ohne diese Option erstellt wurden, nicht funktioniert.

Der Aufruf gibt -1 in `n` zurück, wenn keine Daten zu lesen sind - der Socket wird intern im Non-Blocking-Modus verwendet. Wenn -1 zurückgegeben wird, warten Sie einige Millisekunden, auf Daten. Sie sollten nur ein paar Sekunden warten, bis `easy:Recv()` -1 in `n` zurückgibt. Der Grund dafür ist, dass libcurl oder die SSL-Bibliothek intern einige

Daten zwischenspeichert. Daher sollten Sie erst `easy:Recv()` aufrufen, wenn alle Daten gelesen sind, die zwischengespeicherte Daten enthalten würden.

Darüber hinaus kann `easy:Recv()` -1 in `n` zurückgeben, wenn die einzigen Daten, die gelesen wurden, für die interne SSL-Verarbeitung bestimmt waren und keine anderen Daten verfügbar sind.

EINGABEN

`len` Anzahl der zu lesenden Bytes

RÜCKGABEWERTE

`data$` Daten lesen

`n` Anzahl der gelesenen Bytes

5.58 easy:Reset

BEZEICHNUNG

easy:Reset – setzt alle Optionen einer libcurl Sitzungs-Handle zurück

ÜBERSICHT

```
easy:Reset()
```

BESCHREIBUNG

Setzt alle Optionen, die zuvor an einer bestimmten curl-Easy-Handle eingestellt wurden, auf die Standardwerte zurück. Dadurch wird der Handle wieder in den gleichen Zustand versetzt, in dem er sich befand, als er gerade mit `hurl.Easy()` erstellt wurde.

Es ändert nichts an den folgenden Informationen: Live-Verbindungen, der Sitzungs-ID-Cache, der DNS-Cache, die Cookies und Shares.

EINGABEN

keine

5.59 easy:Send

BEZEICHNUNG

easy:Send – sendet Rohdaten über eine Easy-Verbindung

ÜBERSICHT

```
sent = easy:Send(data$)
```

BESCHREIBUNG

Dieser Befehl sendet beliebige Daten über die aufgebaute Verbindung. Sie können ihn zusammen mit `easy:Recv()` verwenden, um eigene Protokolle mit libcurl zu implementieren. Diese Funktionalität kann besonders nützlich sein, wenn Sie Proxies und/oder SSL-Verschlüsselung verwenden: libcurl übernimmt die Proxy-Vermittlung und den Verbindungsaufbau. Sie müssen die zu sendenden Daten in `data$` übergeben und dürfen auch auch binär sein.

Um die Verbindung herzustellen, setzen Sie die Option `#CURLOPT_CONNECT_ONLY`, bevor Sie `easy:Perform()` oder `multi:Perform()` aufrufen. Beachten Sie, dass `easy:Send()` bei Verbindungen, die ohne diese Option erstellt wurden, nicht funktioniert.

Der Aufruf gibt -1 zurück, wenn es im Moment nicht möglich ist, Daten zu senden. In diesem Fall müssen Sie versuchen, die Daten erneut zu senden, da Curl nicht blockierende Sockets verwendet.

Außerdem kann `easy:Send()` -1 zurückgeben, wenn die einzigen gesendeten Daten für die interne SSL-Verarbeitung bestimmt waren und keine anderen Daten gesendet werden können.

EINGABEN

`data$` zu sendende Daten

RÜCKGABEWERTE

`sent` Anzahl der gesendeten Bytes

5.60 easy:SetOpt

BEZEICHNUNG

easy:SetOpt – stellt die Optionen für einen curl-Easy-Handle ein

ÜBERSICHT

```
easy:SetOpt(option, parameter)
easy:SetOpt(table)
```

BESCHREIBUNG

`easy:SetOpt()` wird verwendet, um durch die Einstellung der entsprechenden Optionen das Verhalten von libcurl durch eine Anwendung zu ändern. Alle Optionen werden mit einer Option gesetzt, gefolgt von einem Parameter. Dieser Parameter kann eine Zahl, eine Zeichenkette, eine Tabelle oder eine Funktionsreferenz sein, je nachdem, was die angegebene Option erwartet. Lesen Sie dieses Handbuch sorgfältig durch, da schlechte Eingabewerte dazu führen können, dass sich libcurl nicht gut verhält!

Die mit diesem Befehl gesetzten Optionen gelten für alle zukünftigen Übertragungen, die mit diesem Handle durchgeführt werden. Die Optionen werden zwischen den Transfers in keiner Weise zurückgesetzt, so dass Sie, wenn Sie spätere Transfers mit unterschiedlichen Optionen wünschen, diese zwischen den Transfers ändern müssen. Optional können Sie alle Optionen mit `easy:Reset()` auf den internen Standard zurücksetzen.

`easy:SetOpt()` kann auf zwei verschiedene Arten verwendet werden: Sie können entweder eine einzelne Option setzen, indem Sie die Argumente `option` und `parameter` übergeben, oder Sie können mehrere Optionen auf einmal festlegen, indem Sie ein Tabellenargument an `easy:SetOpt()` übergeben. Siehe unten für ein Beispiel.

Die Reihenfolge, in der die Optionen eingestellt sind, spielt keine Rolle.

Die folgenden Optionstypen werden derzeit unterstützt:

`#CURLOPT_ABSTRACT_UNIX_SOCKET`

Setzt einen abstrakten Unix-Domänen-Socket. Siehe Abschnitt 5.61 [easy:SetOpt_Abstract_Unix_Socket], Seite 64, für Details.

`#CURLOPT_ACCEPT_ENCODING`
: Ermöglicht die automatische Dekompression von HTTP-Downloads. Siehe Abschnitt 5.62 [`easy:SetOpt_Accept_Encoding`], Seite 65, für Details.

`#CURLOPT_ACCEPTTIMEOUT_MS`
: Setzt die Zeitüberschreitung beim Warten auf die erneute Verbindung des FTP-Servers. Siehe Abschnitt 5.63 [`easy:SetOpt_AcceptTimeout_MS`], Seite 66, für Details.

`#CURLOPT_ADDRESS_SCOPE`
: Legt den Bereich für lokale IPv6-Adressen fest. Siehe Abschnitt 5.64 [`easy:SetOpt_Address_Scope`], Seite 66, für Details.

`#CURLOPT_APPEND`
: Aktiviert das Anhängen an die Remote-Datei. Siehe Abschnitt 5.65 [`easy:SetOpt_Append`], Seite 66, für Details.

`#CURLOPT_AUTOREFERER`
: Setzt die automatische Aktualisierung des Referer-Header. Siehe Abschnitt 5.66 [`easy:SetOpt_AutoReferer`], Seite 67, für Details.

`#CURLOPT_BUFFERSIZE`
: Stellt die bevorzugte Empfangspuffergröße ein. Siehe Abschnitt 5.67 [`easy:SetOpt_BufferSize`], Seite 67, für Details.

`#CURLOPT_CAINFO`
: Setzt den Pfad zum Paket der Zertifizierungsstelle (CA). Siehe Abschnitt 5.68 [`easy:SetOpt_CAInfo`], Seite 67, für Details.

`#CURLOPT_CAPATH`
: Gibt ein Verzeichnis mit CA-Zertifikaten an. Siehe Abschnitt 5.69 [`easy:SetOpt_CAPath`], Seite 68, für Details.

`#CURLOPT_CERTINFO`
: Fordert SSL-Zertifikat-Informationen an. Siehe Abschnitt 5.70 [`easy:SetOpt_CertInfo`], Seite 69, für Details.

`#CURLOPT_CHUNK_BGN_FUNCTION`
: Setzt den Callback vor einer Übertragung mit FTP Platzhalter Übereinstimmung. Siehe Abschnitt 5.71 [`easy:SetOpt_Chunk_BGN_Function`], Seite 69, für Details.

`#CURLOPT_CHUNK_END_FUNCTION`
: Setzt den Callback nach einer Übertragung mit FTP Platzhalter Übereinstimmung. Siehe Abschnitt 5.72 [`easy:SetOpt_Chunk_End_Function`], Seite 70, für Details.

`#CURLOPT_CONNECT_ONLY`
: Stoppt, wenn eine Verbindung zum Zielserver besteht. Siehe Abschnitt 5.73 [`easy:SetOpt_Connect_Only`], Seite 71, für Details.

`#CURLOPT_CONNECT_TO`
: Verbindet mit einem bestimmten Host und Port anstelle des Hosts/Ports der URL. Siehe Abschnitt 5.74 [`easy:SetOpt_Connect_To`], Seite 71, für Details.

`#CURLOPT_CONNECTTIMEOUT`
: Setzt die Zeitüberschreitung für die Verbindungsphase in s. Siehe Abschnitt 5.75 [`easy:SetOpt_ConnectTimeout`], Seite 72, für Details.

`#CURLOPT_CONNECTTIMEOUT_MS`
: Setzt die Zeitüberschreitung für die Verbindungsphase in ms. Siehe Abschnitt 5.76 [`easy:SetOpt_ConnectTimeout_MS`], Seite 73, für Details.

`#CURLOPT_COOKIE`
: Setzt den Inhalt des HTTP-Cookie-Headers. Siehe Abschnitt 5.77 [`easy:SetOpt_Cookie`], Seite 73, für Details.

`#CURLOPT_COOKIEFILE`
: Setzt den Dateinamen, um Cookies zu lesen. Siehe Abschnitt 5.78 [`easy:SetOpt_CookieFile`], Seite 74, für Details.

`#CURLOPT_COOKIEJAR`
: Setzt den Dateinamen, um Cookies zu speichern. Siehe Abschnitt 5.79 [`easy:SetOpt_CookieJar`], Seite 75, für Details.

`#CURLOPT_COOKIELIST`
: Fügt hinzu oder manipuliert von im Speicher befindliche Cookies. Siehe Abschnitt 5.80 [`easy:SetOpt_CookieList`], Seite 75, für Details.

`#CURLOPT_COOKIESESSION`
: Startet eine neue Cookie-Sitzung. Siehe Abschnitt 5.81 [`easy:SetOpt_CookieSession`], Seite 76, für Details.

`#CURLOPT_CRLF`
: Aktiviert/deaktiviert die CRLF-Konvertierung. Siehe Abschnitt 5.82 [`easy:SetOpt_CRLF`], Seite 77, für Details.

`#CURLOPT_CRLFILE`
: Gibt eine Datei für Zertifikatssperrlisten an. Siehe Abschnitt 5.83 [`easy:SetOpt_CRLFile`], Seite 77, für Details.

`#CURLOPT_CUSTOMREQUEST`
: Setzt die benutzerdefinierte Zeichenkette für die Anforderung. Siehe Abschnitt 5.84 [`easy:SetOpt_CustomRequest`], Seite 78, für Details.

`#CURLOPT_DEBUGFUNCTION`
: Setzt die Debug-Callback-Funktion. Siehe Abschnitt 5.85 [`easy:SetOpt_DebugFunction`], Seite 79, für Details.

`#CURLOPT_DEFAULT_PROTOCOL`
: Setzt das Standardprotokoll bei fehlendem Schemanamen. Siehe Abschnitt 5.86 [`easy:SetOpt_Default_Protocol`], Seite 80, für Details.

`#CURLOPT_DIRLISTONLY`
: Fragt nur nach Namen in einer Verzeichnisliste. Siehe Abschnitt 5.87 [`easy:SetOpt_DirListOnly`], Seite 81, für Details.

`#CURLOPT_DNS_CACHE_TIMEOUT`
: Legt die Lebensdauer für DNS-Cache-Einträge fest. Siehe Abschnitt 5.88 [`easy:SetOpt_DNS_Cache_Timeout`], Seite 82, für Details.

`#CURLOPT_DNS_INTERFACE`
: Stellt die Schnittstelle so ein, dass über DNS kommuniziert wird. Siehe Abschnitt 5.89 [`easy:SetOpt_DNS_Interface`], Seite 82, für Details.

`#CURLOPT_DNS_LOCAL_IP4`
: Setzt die IPv4-Adresse, an die DNS-Auflösungen gebunden werden sollen. Siehe Abschnitt 5.90 [`easy:SetOpt_DNS_Local_IP4`], Seite 83, für Details.

`#CURLOPT_DNS_LOCAL_IP6`
: Setzt die IPv6-Adresse, an die DNS-Auflösungen gebunden werden sollen. Siehe Abschnitt 5.91 [`easy:SetOpt_DNS_Local_IP6`], Seite 83, für Details.

`#CURLOPT_DNS_SERVERS`
: Legt bevorzugte DNS-Server fest. Siehe Abschnitt 5.92 [`easy:SetOpt_DNS_Servers`], Seite 83, für Details.

`#CURLOPT_DNS_USE_GLOBAL_CACHE`
: VERALTET: Aktiviert/deaktiviert den globalen DNS-Cache. Siehe Abschnitt 5.93 [`easy:SetOpt_DNS_Use_Global_Cache`], Seite 84, für Details.

`#CURLOPT_EGDSOCKET`
: Stellt den EGD-Socketpfad ein. Siehe Abschnitt 5.94 [`easy:SetOpt_EGDSocket`], Seite 84, für Details.

`#CURLOPT_EXPECT_100_TIMEOUT_MS`
: Setzt die Zeitüberschreitung bei der Antwort Expect: 100-continue. Siehe Abschnitt 5.95 [`easy:SetOpt_Expect_100_Timeout_MS`], Seite 84, für Details.

`#CURLOPT_FAILONERROR`
: Setzt den Anforderungsfehler bei HTTP-Antwort >= 400. Siehe Abschnitt 5.96 [`easy:SetOpt_FailOnError`], Seite 85, für Details.

`#CURLOPT_FILETIME`
: Liefert die Änderungszeit der Remote-Datenquelle. Siehe Abschnitt 5.97 [`easy:SetOpt_FileTime`], Seite 85, für Details.

`#CURLOPT_FNMATCH_FUNCTION`
: Setzt die Callback-Platzhalterabgleich-Funktion. Siehe Abschnitt 5.98 [`easy:SetOpt_FNMatch_Function`], Seite 86, für Details.

`#CURLOPT_FOLLOWLOCATION`
: Folgt HTTP 3xx Umleitungen. Siehe Abschnitt 5.99 [`easy:SetOpt_FollowLocation`], Seite 86, für Details.

`#CURLOPT_FORBID_REUSE`
: Schließt die Verbindung sofort, nachdem die Übertragung beendet ist. Siehe Abschnitt 5.100 [`easy:SetOpt_Forbid_Reuse`], Seite 87, für Details.

`#CURLOPT_FRESH_CONNECT`
: Erzwingt, dass eine neue Verbindung verwendet wird. Siehe Abschnitt 5.101 [`easy:SetOpt_Fresh_Connect`], Seite 87, für Details.

`#CURLOPT_FTP_ACCOUNT`
: Sendet Kontoinformationen mit ACCT an FTP-Server. Siehe Abschnitt 5.102 [easy:SetOpt_FTP_Account], Seite 88, für Details.

`#CURLOPT_FTP_ALTERNATIVE_TO_USER`
: Legt fest, dass FTP anstelle von USER verwendet wird. Siehe Abschnitt 5.103 [easy:SetOpt_FTP_Alternative_To_User], Seite 88, für Details.

`#CURLOPT_FTP_CREATE_MISSING_DIRS`
: Erstellt fehlende Verzeichnisse für FTP und SFTP. Siehe Abschnitt 5.104 [easy:SetOpt_FTP_Create_Missing_Dirs], Seite 89, für Details.

`#CURLOPT_FTP_FILEMETHOD`
: Wählt das Verzeichnisdurchlaufverfahren für FTP aus. Siehe Abschnitt 5.105 [easy:SetOpt_FTP_FileMethod], Seite 89, für Details.

`#CURLOPT_FTP_RESPONSE_TIMEOUT`
: Zeit, die auf die FTP-Antwort gewartet wird. Siehe Abschnitt 5.106 [easy:SetOpt_FTP_Response_Timeout], Seite 90, für Details.

`#CURLOPT_FTP_SKIP_PASV_IP`
: Ignoriert die IP-Adresse in der PASV-Antwort. Siehe Abschnitt 5.107 [easy:SetOpt_FTP_Skip_PASV_IP], Seite 90, für Details.

`#CURLOPT_FTP_SSL_CCC`
: Schaltet SSL mit FTP nach der Authentifizierung wieder aus. Siehe Abschnitt 5.108 [easy:SetOpt_FTP_SSL_CCC], Seite 91, für Details.

`#CURLOPT_FTP_USE_EPRT`
: Aktiviert/deaktiviert die Nutzung von EPRT mit FTP. Siehe Abschnitt 5.109 [easy:SetOpt_FTP_Use_Eprt], Seite 91, für Details.

`#CURLOPT_FTP_USE_EPSV`
: Aktiviert/deaktiviert die Nutzung von EPSV mit FTP. Siehe Abschnitt 5.110 [easy:SetOpt_FTP_Use_Epsv], Seite 92, für Details.

`#CURLOPT_FTP_USE_PRET`
: Aktiviert/deaktiviert den PRET-Befehl mit FTP. Siehe Abschnitt 5.111 [easy:SetOpt_FTP_Use_Pret], Seite 92, für Details.

`#CURLOPT_FTPPORT`
: Aktiviert die FTP-Übertragung. Siehe Abschnitt 5.112 [easy:SetOpt_FTPPort], Seite 93, für Details.

`#CURLOPT_FTPSSLAUTH`
: Legt die Reihenfolge fest, in der TLS vs. SSL bei der Verwendung von FTP versucht werden soll. Siehe Abschnitt 5.113 [easy:SetOpt_FTPSSLAuth], Seite 93, für Details.

`#CURLOPT_GSSAPI_DELEGATION`
: Legt die erlaubte Zuordnung von GSS-APIs fest. Siehe Abschnitt 5.114 [easy:SetOpt_GSSAPI_Delegation], Seite 94, für Details.

`#CURLOPT_HEADER`
: Ubergibt den Header an den Datenstrom. Siehe Abschnitt 5.115 [`easy:SetOpt_Header`], Seite 94, für Details.

`#CURLOPT_HEADERFUNCTION`
: Setzt die Callback-Funktion, die Header-Daten empfängt. Siehe Abschnitt 5.116 [`easy:SetOpt_HeaderFunction`], Seite 95, für Details.

`#CURLOPT_HEADEROPT`
: Legt fest, wie HTTP-Header gesendet werden sollen. Siehe Abschnitt 5.117 [`easy:SetOpt_HeaderOpt`], Seite 96, für Details.

`#CURLOPT_HTTP200ALIASES`
: Setzt alternative Übereinstimmungen für HTTP 200 OK. Siehe Abschnitt 5.118 [`easy:SetOpt_HTTP200Aliases`], Seite 97, für Details.

`#CURLOPT_HTTP_CONTENT_DECODING`
: Aktiviert/deaktiviert die Dekodierung von HTTP-Inhalten. Siehe Abschnitt 5.119 [`easy:SetOpt_HTTP_Content_Decoding`], Seite 97, für Details.

`#CURLOPT_HTTP_TRANSFER_DECODING`
: Aktiviert/deaktiviert die Dekodierung der HTTP-Übertragung. Siehe Abschnitt 5.120 [`easy:SetOpt_HTTP_Transfer_Decoding`], Seite 98, für Details.

`#CURLOPT_HTTP_VERSION`
: Gibt die zu verwendende HTTP-Protokollversion an. Siehe Abschnitt 5.121 [`easy:SetOpt_HTTP_Version`], Seite 98, für Details.

`#CURLOPT_HTTPAUTH`
: Legt die HTTP-Server-Authentifizierungsmethoden für den Versuch fest. Siehe Abschnitt 5.122 [`easy:SetOpt_HTTPAuth`], Seite 99, für Details.

`#CURLOPT_HTTPGET`
: Fragt nach einer HTTP GET-Anfrage. Siehe Abschnitt 5.123 [`easy:SetOpt_HTTPGet`], Seite 101, für Details.

`#CURLOPT_HTTPHEADER`
: Legt den benutzerdefinierten HTTP-Header fest. Siehe Abschnitt 5.124 [`easy:SetOpt_HTTPHeader`], Seite 101, für Details.

`#CURLOPT_HTTPPOST`
: Gibt den mehrteiligen Formpost-Inhalt an. Siehe Abschnitt 5.125 [`easy:SetOpt_HTTPPost`], Seite 102, für Details.

`#CURLOPT_HTTPPROXYTUNNEL`
: Setzt den Tunnel durch einen HTTP-Proxy. Siehe Abschnitt 5.126 [`easy:SetOpt_HTTPProxyTunnel`], Seite 103, für Details.

`#CURLOPT_IGNORE_CONTENT_LENGTH`
: Ignoriert den Content-Length-Header. Siehe Abschnitt 5.127 [`easy:SetOpt_Ignore_Content_Length`], Seite 103, für Details.

`#CURLOPT_INFILESIZE`
: Legt die Größe der zu sendenden Eingabedatei fest. Siehe Abschnitt 5.128 [`easy:SetOpt_InFileSize`], Seite 104, für Details.

`#CURLOPT_INFILESIZE_LARGE`
: Legt die Größe der zu sendenden Eingabedatei fest. Siehe Abschnitt 5.129 [`easy:SetOpt_InFileSize_Large`], Seite 104, für Details.

`#CURLOPT_INTERFACE`
: Legt die Quellschnittstelle für ausgehenden Datenverkehr fest. Siehe Abschnitt 5.130 [`easy:SetOpt_Interface`], Seite 105, für Details.

`#CURLOPT_IPRESOLVE`
: Gibt an, welche IP-Protokollversion verwendet werden soll. Siehe Abschnitt 5.131 [`easy:SetOpt_IPResolve`], Seite 105, für Details.

`#CURLOPT_ISSUERCERT`
: Setzt den Dateiname des SSL-Zertifikats des Ausstellers. Siehe Abschnitt 5.132 [`easy:SetOpt_IssuerCert`], Seite 106, für Details.

`#CURLOPT_KEEP_SENDING_ON_ERROR`
: Sendet weiter mit einer frühen HTTP-Antwort >= 300. Siehe Abschnitt 5.133 [`easy:SetOpt_Keep_Sending_On_Error`], Seite 107, für Details.

`#CURLOPT_KEYPASSWD`
: Setzt die Passphrase auf privaten Schlüssel. Siehe Abschnitt 5.134 [`easy:SetOpt_KeyPasswd`], Seite 107, für Details.

`#CURLOPT_KRBLEVEL`
: Legt die FTP-Kerberos-Sicherheitsstufe fest. Siehe Abschnitt 5.135 [`easy:SetOpt_KRBLevel`], Seite 107, für Details.

`#CURLOPT_LOCALPORT`
: Legt die lokale Portnummer für Socket fest. Siehe Abschnitt 5.136 [`easy:SetOpt_LocalPort`], Seite 108, für Details.

`#CURLOPT_LOCALPORTRANGE`
: Legt die Anzahl zusätzlicher lokaler Ports zum Testen fest. Siehe Abschnitt 5.137 [`easy:SetOpt_LocalPortRange`], Seite 108, für Details.

`#CURLOPT_LOGIN_OPTIONS`
: Legt die Login-Optionen fest. Siehe Abschnitt 5.138 [`easy:SetOpt_Login_Options`], Seite 109, für Details.

`#CURLOPT_LOW_SPEED_LIMIT`
: Stellt eine niedrige Geschwindigkeitsbegrenzung in Bytes pro Sekunde ein. Siehe Abschnitt 5.139 [`easy:SetOpt_Low_Speed_Limit`], Seite 109, für Details.

`#CURLOPT_LOW_SPEED_TIME`
: Stellt das Zeitlimit für niedrige Geschwindigkeit ein. Siehe Abschnitt 5.140 [`easy:SetOpt_Low_Speed_Time`], Seite 110, für Details.

`#CURLOPT_MAIL_AUTH`
: Legt die SMTP-Authentifizierungsadresse fest. Siehe Abschnitt 5.141 [easy:SetOpt_Mail_Auth], Seite 110, für Details.

`#CURLOPT_MAIL_FROM`
: Gibt die SMTP-Absenderadresse an. Siehe Abschnitt 5.142 [easy:SetOpt_Mail_From], Seite 110, für Details.

`#CURLOPT_MAIL_RCPT`
: Gibt die Liste der SMTP-Mail-Empfänger an. Siehe Abschnitt 5.143 [easy:SetOpt_Mail_RCPT], Seite 111, für Details.

`#CURLOPT_MAX_RECV_SPEED_LARGE`
: Setzt die Geschwindigkeitslimit für das Herunterladen von Daten. Siehe Abschnitt 5.144 [easy:SetOpt_Max_Recv_Speed_Large], Seite 111, für Details.

`#CURLOPT_MAX_SEND_SPEED_LARGE`
: Setzt die Geschwindigkeitslimit für das Hochladen von Daten. Siehe Abschnitt 5.145 [easy:SetOpt_Max_Send_Speed_Large], Seite 112, für Details.

`#CURLOPT_MAXCONNECTS`
: Setzt die maximale Verbindungs-Cache-Größe. Siehe Abschnitt 5.146 [easy:SetOpt_MaxConnects], Seite 112, für Details.

`#CURLOPT_MAXFILESIZE`
: Setzt die maximal zulässige Dateigröße für das Herunterladen. Siehe Abschnitt 5.147 [easy:SetOpt_MaxFileSize], Seite 113, für Details.

`#CURLOPT_MAXFILESIZE_LARGE`
: Setzt die maximal zulässige Dateigröße für das Herunterladen. Siehe Abschnitt 5.148 [easy:SetOpt_MaxFileSize_Large], Seite 113, für Details.

`#CURLOPT_MAXREDIRS`
: Setzt die maximale Anzahl von erlaubten Umleitungen. Siehe Abschnitt 5.149 [easy:SetOpt_MaxRedirs], Seite 114, für Details.

`#CURLOPT_NETRC`
: Fordert an, dass .netrc verwendet wird. Siehe Abschnitt 5.150 [easy:SetOpt_Netrc], Seite 114, für Details.

`#CURLOPT_NETRC_FILE`
: Setzt den Dateiname zum Lesen von .netrc-Informationen. Siehe Abschnitt 5.151 [easy:SetOpt_Netrc_File], Seite 115, für Details.

`#CURLOPT_NEW_DIRECTORY_PERMS`
: Setzt die Berechtigungen für neu erstellte Remote-Verzeichnisse. Siehe Abschnitt 5.152 [easy:SetOpt_New_Directory_Perms], Seite 115, für Details.

`#CURLOPT_NEW_FILE_PERMS`
: Setzt die Berechtigungen für neu erstellte Remote-Dateien. Siehe Abschnitt 5.153 [easy:SetOpt_New_File_Perms], Seite 116, für Details.

`#CURLOPT_NOBODY`
: Führt die Download-Anfrage durch, ohne den Body zu erhalten. Siehe Abschnitt 5.154 [`easy:SetOpt_Nobody`], Seite 116, für Details.

`#CURLOPT_NOPROGRESS`
: Schaltet die Fortschrittsanzeige aus. Siehe Abschnitt 5.155 [`easy:SetOpt_NoProgress`], Seite 116, für Details.

`#CURLOPT_NOPROXY`
: Deaktiviert die Proxy-Nutzung für bestimmte Hosts. Siehe Abschnitt 5.156 [`easy:SetOpt_NoProxy`], Seite 117, für Details.

`#CURLOPT_NOSIGNAL`
: Überspringt die gesamte Signalverarbeitung. Siehe Abschnitt 5.157 [`easy:SetOpt_NoSignal`], Seite 117, für Details.

`#CURLOPT_PASSWORD`
: Setzt das Passwort zur Verwendung bei der Authentifizierung. Siehe Abschnitt 5.158 [`easy:SetOpt_Password`], Seite 118, für Details.

`#CURLOPT_PATH_AS_IS`
: Verwendet keine Punkt-Punkt-Sequenzen. Siehe Abschnitt 5.159 [`easy:SetOpt_Path_As_Is`], Seite 118, für Details.

`#CURLOPT_PINNEDPUBLICKEY`
: Legt das Public Key Pinning fest. Siehe Abschnitt 5.160 [`easy:SetOpt_PinnedPublicKey`], Seite 119, für Details.

`#CURLOPT_PIPEWAIT`
: Wartet auf Pipelining/Multiplexing. Siehe Abschnitt 5.161 [`easy:SetOpt_PipeWait`], Seite 119, für Details.

`#CURLOPT_PORT`
: Stellt die Nummer des Remote-Ports ein, mit dem gearbeitet werden soll. Siehe Abschnitt 5.162 [`easy:SetOpt_Port`], Seite 120, für Details.

`#CURLOPT_POST`
: Fordert einen HTTP-POST an. Siehe Abschnitt 5.163 [`easy:SetOpt_Post`], Seite 121, für Details.

`#CURLOPT_POSTFIELDS`
: Setzt die Daten, welche an den Server gesendet werden. Siehe Abschnitt 5.164 [`easy:SetOpt_PostFields`], Seite 121, für Details.

`#CURLOPT_POSTQUOTE`
: Setzt die (S)FTP-Befehle zur Ausführung nach der Übertragung. Siehe Abschnitt 5.165 [`easy:SetOpt_PostQuote`], Seite 122, für Details.

`#CURLOPT_POSTREDIR`
: Setzt die Vorgehensweise bei einer HTTP-POST-Umleitung. Siehe Abschnitt 5.166 [`easy:SetOpt_PostRedir`], Seite 123, für Details.

`#CURLOPT_PRE_PROXY`
: Stellt den Prä-Proxy für die Verwendung ein. Siehe Abschnitt 5.167 [`easy:SetOpt_Pre_Proxy`], Seite 123, für Details.

`#CURLOPT_PREQUOTE`
: Setzt die Befehle, die vor einer FTP-Übertragung ausgeführt werden sollen. Siehe Abschnitt 5.168 [`easy:SetOpt_Prequote`], Seite 124, für Details.

`#CURLOPT_PROGRESSFUNCTION`
: Bestimmt den Callback zur Fortschrittsanzeige-Funktion. Siehe Abschnitt 5.169 [`easy:SetOpt_ProgressFunction`], Seite 124, für Details.

`#CURLOPT_PROTOCOLS`
: Stellt die erlaubten Protokolle ein. Siehe Abschnitt 5.170 [`easy:SetOpt_Protocols`], Seite 125, für Details.

`#CURLOPT_PROXY`
: Stellt den Proxy für die Verwendung ein. Siehe Abschnitt 5.171 [`easy:SetOpt_Proxy`], Seite 126, für Details.

`#CURLOPT_PROXY_CAINFO`
: Setzt den Pfad zum Proxy Certificate Authority (CA)-Paket. Siehe Abschnitt 5.172 [`easy:SetOpt_Proxy_CAInfo`], Seite 127, für Details.

`#CURLOPT_PROXY_CAPATH`
: Gibt ein Verzeichnis mit Proxy-CA-Zertifikaten an. Siehe Abschnitt 5.173 [`easy:SetOpt_Proxy_CAPath`], Seite 128, für Details.

`#CURLOPT_PROXY_CRLFILE`
: Gibt eine Datei für Proxy-Zertifikatssperrlisten an. Siehe Abschnitt 5.174 [`easy:SetOpt_Proxy_CRLFile`], Seite 129, für Details.

`#CURLOPT_PROXY_KEYPASSWD`
: Setzt die Passphrase auf privaten Proxy-Schlüssel. Siehe Abschnitt 5.175 [`easy:SetOpt_Proxy_KeyPasswd`], Seite 129, für Details.

`#CURLOPT_PROXY_PINNEDPUBLICKEY`
: Legt das Public Key Pinning für https-Proxy fest. Siehe Abschnitt 5.176 [`easy:SetOpt_Proxy_PinnedPublicKey`], Seite 130, für Details.

`#CURLOPT_PROXY_SERVICE_NAME`
: Setzt den Namen des Proxy-Authentifizierungsdienstes. Siehe Abschnitt 5.177 [`easy:SetOpt_Proxy_Service_Name`], Seite 130, für Details.

`#CURLOPT_PROXY_SSL_CIPHER_LIST`
: Legt die für Proxy-TLS zu verwendenden Verschlüsselungsart fest. Siehe Abschnitt 5.178 [`easy:SetOpt_Proxy_SSL_Cipher_List`], Seite 130, für Details.

`#CURLOPT_PROXY_SSL_OPTIONS`
: Legt die Proxy-SSL-Verhaltensoptionen fest. Siehe Abschnitt 5.179 [`easy:SetOpt_Proxy_SSL_Options`], Seite 131, für Details.

`#CURLOPT_PROXY_SSL_VERIFYHOST`
: Überprüft den Namen des Proxy-Zertifikats anhand des Hosts. Siehe Abschnitt 5.180 [`easy:SetOpt_Proxy_SSL_VerifyHost`], Seite 132, für Details.

`#CURLOPT_PROXY_SSL_VERIFYPEER`
: Aktiviert/deaktiviert die Überprüfung des SSL-Zertifikats des Proxys. Siehe Abschnitt 5.181 [`easy:SetOpt_Proxy_SSL_VerifyPeer`], Seite 132, für Details.

`#CURLOPT_PROXY_SSLCERT`
: Legt das SSL-Proxy-Client-Zertifikat fest. Siehe Abschnitt 5.182 [`easy:SetOpt_Proxy_SSLCert`], Seite 133, für Details.

`#CURLOPT_PROXY_SSLCERTTYPE`
: Setzt den Typ des Proxy-Client-SSL-Zertifikats. Siehe Abschnitt 5.183 [`easy:SetOpt_Proxy_SSLCertType`], Seite 134, für Details.

`#CURLOPT_PROXY_SSLKEY`
: Setzt eine private Schlüsseldatei für das TLS- und SSL-Proxy-Client-Zertifikat. Siehe Abschnitt 5.184 [`easy:SetOpt_Proxy_SSLKey`], Seite 134, für Details.

`#CURLOPT_PROXY_SSLKEYTYPE`
: Legt den Typ der privaten Proxy-Schlüsseldatei fest. Siehe Abschnitt 5.185 [`easy:SetOpt_Proxy_SSLKeyType`], Seite 135, für Details.

`#CURLOPT_PROXY_SSLVERSION`
: Legt die bevorzugte Proxy-TLS/SSL-Version fest. Siehe Abschnitt 5.186 [`easy:SetOpt_Proxy_SSLVersion`], Seite 135, für Details.

`#CURLOPT_PROXY_TLSAUTH_PASSWORD`
: Setzt das Passwort für die Proxy-TLS-Authentifizierung. Siehe Abschnitt 5.187 [`easy:SetOpt_Proxy_TLSAuth_Password`], Seite 136, für Details.

`#CURLOPT_PROXY_TLSAUTH_TYPE`
: Legt die Proxy-TLS-Authentifizierungsmethoden fest. Siehe Abschnitt 5.188 [`easy:SetOpt_Proxy_TLSAuth_Type`], Seite 137, für Details.

`#CURLOPT_PROXY_TLSAUTH_USERNAME`
: Setzt den Benutzername zur Verwendung für die Proxy-TLS-Authentifizierung. Siehe Abschnitt 5.189 [`easy:SetOpt_Proxy_TLSAuth_UserName`], Seite 137, für Details.

`#CURLOPT_PROXY_TRANSFER_MODE`
: Hängt den FTP-Übertragungsmodus an die URL für Proxy an. Siehe Abschnitt 5.190 [`easy:SetOpt_Proxy_Transfer_Mode`], Seite 138, für Details.

`#CURLOPT_PROXYAUTH`
: Legt die HTTP-Proxy-Authentifizierungsmethoden für den Versuch fest. Siehe Abschnitt 5.191 [`easy:SetOpt_ProxyAuth`], Seite 138, für Details.

`#CURLOPT_PROXYHEADER`
: Setzt die an den Proxy zu übergebenden benutzerdefinierte HTTP-Header. Siehe Abschnitt 5.192 [`easy:SetOpt_ProxyHeader`], Seite 138, für Details.

`#CURLOPT_PROXYPASSWORD`
: Setzt das Passwort für die Proxy-Authentifizierung. Siehe Abschnitt 5.193 [`easy:SetOpt_ProxyPassword`], Seite 139, für Details.

`#CURLOPT_PROXYPORT`
: Setzt die Portnummer für den Proxy. Siehe Abschnitt 5.194 [`easy:SetOpt_ProxyPort`], Seite 139, für Details.

`#CURLOPT_PROXYTYPE`
: Setzt den Proxy-Protokolltyp. Siehe Abschnitt 5.195 [`easy:SetOpt_ProxyType`], Seite 140, für Details.

`#CURLOPT_PROXYUSERNAME`
: Setzt den Benutzername für die Proxy-Authentifizierung. Siehe Abschnitt 5.196 [`easy:SetOpt_ProxyUserName`], Seite 140, für Details.

`#CURLOPT_PROXYUSERPWD`
: Setzt den Benutzernamen und das Passwort für die Proxy-Authentifizierung. Siehe Abschnitt 5.197 [`easy:SetOpt_ProxyUserPwd`], Seite 141, für Details.

`#CURLOPT_PUT`
: Stellt eine HTTP-PUT-Anfrage. Siehe Abschnitt 5.198 [`easy:SetOpt_Put`], Seite 141, für Details.

`#CURLOPT_QUOTE`
: Setzt die (S)FTP-Befehle, die vor der Übertragung ausgeführt werden sollen. Siehe Abschnitt 5.199 [`easy:SetOpt_Quote`], Seite 142, für Details.

`#CURLOPT_RANDOM_FILE`
: Gibt eine Quelle für zufällige Daten an. Siehe Abschnitt 5.200 [`easy:SetOpt_Random_File`], Seite 143, für Details.

`#CURLOPT_RANGE`
: Stellt den anzuforderenden Bytebereich ein. Siehe Abschnitt 5.201 [`easy:SetOpt_Range`], Seite 143, für Details.

`#CURLOPT_READFUNCTION`
: Liest den Callback für Daten-Uploads. Siehe Abschnitt 5.202 [`easy:SetOpt_ReadFunction`], Seite 144, für Details.

`#CURLOPT_REDIR_PROTOCOLS`
: Legt die Protokolle fest, zu denen umgeleitet werden darf. Siehe Abschnitt 5.203 [`easy:SetOpt_Redir_Protocols`], Seite 145, für Details.

`#CURLOPT_REFERER`
: Setzt den HTTP Referer: Header. Siehe Abschnitt 5.204 [`easy:SetOpt_Referer`], Seite 146, für Details.

`#CURLOPT_REQUEST_TARGET`
: Gibt ein alternatives Ziel für diese Anforderung an. Siehe Abschnitt 5.205 [`easy:SetOpt_Request_Target`], Seite 147, für Details.

`#CURLOPT_RESOLVE`
: Gibt einen benutzerdefinierten Hostnamen für IP-Adressauflösungen an. Siehe Abschnitt 5.206 [`easy:SetOpt_Resolve`], Seite 147, für Details.

`#CURLOPT_RESUME_FROM`
: Setzt eine Übertragung fort. Siehe Abschnitt 5.207 [`easy:SetOpt_Resume_From`], Seite 148, für Details.

`#CURLOPT_RESUME_FROM_LARGE`
: Legt die Position fest, von der aus die Übertragung fortgesetzt wird. Siehe Abschnitt 5.208 [`easy:SetOpt_Resume_From_Large`], Seite 148, für Details.

`#CURLOPT_RTSP_CLIENT_CSEQ`
: Legt die RTSP-Client-CSEQ-Nummer fest. Siehe Abschnitt 5.209 [`easy:SetOpt_RTSP_Client_CSeq`], Seite 149, für Details.

`#CURLOPT_RTSP_REQUEST`
: Gibt die RTSP-Anfrage an. Siehe Abschnitt 5.210 [`easy:SetOpt_RTSP_Request`], Seite 149, für Details.

`#CURLOPT_RTSP_SERVER_CSEQ`
: Stellt die CSEQ-Nummer des RTSP-Servers ein. Siehe Abschnitt 5.211 [`easy:SetOpt_RTSP_Server_CSeq`], Seite 151, für Details.

`#CURLOPT_RTSP_SESSION_ID`
: Legt die RTSP-Sitzungs-ID fest. Siehe Abschnitt 5.212 [`easy:SetOpt_RTSP_Session_ID`], Seite 151, für Details.

`#CURLOPT_RTSP_STREAM_URI`
: Stellt die RTSP-Stream-URI ein. Siehe Abschnitt 5.213 [`easy:SetOpt_RTSP_Stream_URI`], Seite 152, für Details.

`#CURLOPT_RTSP_TRANSPORT`
: Setzt den RTSP Transport: Header. Siehe Abschnitt 5.214 [`easy:SetOpt_RTSP_Transport`], Seite 152, für Details.

`#CURLOPT_SASL_IR`
: Aktiviert das Senden der ersten Antwort im ersten Paket. Siehe Abschnitt 5.215 [`easy:SetOpt_SASL_IR`], Seite 153, für Details.

`#CURLOPT_SEEKFUNCTION`
: Legt den Benutzer-Callback zum Suchen im Eingabedatenstrom fest. Siehe Abschnitt 5.216 [`easy:SetOpt_SeekFunction`], Seite 153, für Details.

`#CURLOPT_SERVICE_NAME`
: Setzt den Namen des Authentifizierungsdienstes. Siehe Abschnitt 5.217 [`easy:SetOpt_Service_Name`], Seite 154, für Details.

`#CURLOPT_SHARE`
: Gibt den Share-Handle an. Siehe Abschnitt 5.218 [`easy:SetOpt_Share`], Seite 154, für Details.

`#CURLOPT_SOCKS5_AUTH`
: Legt die zulässigen Methoden für die SOCKS5-Proxyauthentifizierung fest. Siehe Abschnitt 5.219 [`easy:SetOpt_Socks5_Auth`], Seite 155, für Details.

`#CURLOPT_SOCKS5_GSSAPI_NEC`
: Setzt den Socks Proxy gssapi Übertragungsschutz. Siehe Abschnitt 5.220 [`easy:SetOpt_Socks5_GSSAPI_NEC`], Seite 155, für Details.

`#CURLOPT_SOCKS5_GSSAPI_SERVICE`
: Setzt den SOCKS5-Name des Proxy-Authentifizierungsdienstes. Siehe Abschnitt 5.221 [`easy:SetOpt_Socks5_GSSAPI_Service`], Seite 156, für Details.

`#CURLOPT_SSH_AUTH_TYPES`
: Stellt den gewünschten Authentifizierungstypen für SFTP und SCP ein. Siehe Abschnitt 5.222 [`easy:SetOpt_SSH_Auth_Types`], Seite 156, für Details.

`#CURLOPT_SSH_HOST_PUBLIC_KEY_MD5`
: Setzt die Prüfsumme des öffentlichen Schlüssels des SSH-Servers. Siehe Abschnitt 5.223 [`easy:SetOpt_SSH_Host_Public_Key_MD5`], Seite 157, für Details.

`#CURLOPT_SSH_KNOWNHOSTS`
: Setzt den Dateiname mit den bekannten SSH-Hosts. Siehe Abschnitt 5.224 [`easy:SetOpt_SSH_KnownHosts`], Seite 157, für Details.

`#CURLOPT_SSH_PRIVATE_KEYFILE`
: Legt die private Schlüsseldatei für SSH-Authentifizierung fest. Siehe Abschnitt 5.225 [`easy:SetOpt_SSH_Private_KeyFile`], Seite 157, für Details.

`#CURLOPT_SSH_PUBLIC_KEYFILE`
: Legt die öffentliche Schlüsseldatei für die SSH-Authentifizierung fest. Siehe Abschnitt 5.226 [`easy:SetOpt_SSH_Public_KeyFile`], Seite 158, für Details.

`#CURLOPT_SSL_CIPHER_LIST`
: Gibt die Verschlüsselung an, die für TLS verwendet werden soll. Siehe Abschnitt 5.227 [`easy:SetOpt_SSL_Cipher_List`], Seite 158, für Details.

`#CURLOPT_SSL_ENABLE_ALPN`
: Aktiviert/deaktiviert ALPN. Siehe Abschnitt 5.228 [`easy:SetOpt_SSL_Enable_Alpn`], Seite 159, für Details.

`#CURLOPT_SSL_ENABLE_NPN`
: Aktiviert/deaktiviert NPN. Siehe Abschnitt 5.229 [`easy:SetOpt_SSL_Enable_Npn`], Seite 159, für Details.

`#CURLOPT_SSL_FALSESTART`
: Aktiviert/deaktiviert TLS-Fehlstart. Siehe Abschnitt 5.230 [`easy:SetOpt_SSL_FalseStart`], Seite 160, für Details.

`#CURLOPT_SSL_OPTIONS`
: Legt SSL-Verhaltensoptionen fest. Siehe Abschnitt 5.231 [`easy:SetOpt_SSL_Options`], Seite 160, für Details.

`#CURLOPT_SSL_SESSIONID_CACHE`
: Aktiviert/deaktiviert die Verwendung des SSL-Sitzungs-ID-Cache. Siehe Abschnitt 5.232 [`easy:SetOpt_SSL_SessionID_Cache`], Seite 161, für Details.

`#CURLOPT_SSL_VERIFYHOST`
: Überprüft den Namen des Zertifikats anhand des Hosts. Siehe Abschnitt 5.233 [`easy:SetOpt_SSL_VerifyHost`], Seite 161, für Details.

`#CURLOPT_SSL_VERIFYPEER`
: Überprüft das SSL-Zertifikat des Peers. Siehe Abschnitt 5.234 [`easy:SetOpt_SSL_VerifyPeer`], Seite 162, für Details.

`#CURLOPT_SSL_VERIFYSTATUS`
: Überprüft den Status des Zertifikats. Siehe Abschnitt 5.235 [`easy:SetOpt_SSL_VerifyStatus`], Seite 163, für Details.

`#CURLOPT_SSLCERT`
: Stellt das SSL-Client-Zertifikat ein. Siehe Abschnitt 5.236 [`easy:SetOpt_SSLCert`], Seite 163, für Details.

`#CURLOPT_SSLCERTTYPE`
: Gibt den Typ des Client-SSL-Zertifikats an. Siehe Abschnitt 5.237 [`easy:SetOpt_SSLCertType`], Seite 164, für Details.

`#CURLOPT_SSLENGINE`
: Setzt die SSL-System-ID. Siehe Abschnitt 5.238 [`easy:SetOpt_SSLEngine`], Seite 164, für Details.

`#CURLOPT_SSLENGINE_DEFAULT`
: Legt das SSL-System als Standard fest. Siehe Abschnitt 5.239 [`easy:SetOpt_SSLEngine_Default`], Seite 165, für Details.

`#CURLOPT_SSLKEY`
: Gibt eine private Schlüsseldatei für TLS- und SSL-Client-Zertifikate an. Siehe Abschnitt 5.240 [`easy:SetOpt_SSLKey`], Seite 165, für Details.

`#CURLOPT_SSLKEYTYPE`
: Setzt den Typ der privaten Schlüsseldatei. Siehe Abschnitt 5.241 [`easy:SetOpt_SSLKeyType`], Seite 166, für Details.

`#CURLOPT_SSLVERSION`
: Stellt die bevorzugte TLS/SSL-Version ein. Siehe Abschnitt 5.242 [`easy:SetOpt_SSLVersion`], Seite 166, für Details.

`#CURLOPT_STREAM_DEPENDS`
: Stellt den Stream ein, von dem diese Übertragung abhängt. Siehe Abschnitt 5.243 [`easy:SetOpt_Stream_Depends`], Seite 167, für Details.

`#CURLOPT_STREAM_DEPENDS_E`
: Stellt den Stream ein, von dem diese Übertragung ausschließlich abhängt. Siehe Abschnitt 5.244 [`easy:SetOpt_Stream_Depends_e`], Seite 168, für Details.

`#CURLOPT_STREAM_WEIGHT`
: Setzt die Gewichtung des numerischen Datenstroms. Siehe Abschnitt 5.245 [`easy:SetOpt_Stream_Weight`], Seite 168, für Details.

`#CURLOPT_SUPPRESS_CONNECT_HEADERS`
: Unterdrückt Proxy-CONNECT-Antwort-Header von Benutzer-Callbacks. Siehe Abschnitt 5.246 [`easy:SetOpt_Suppress_Connect_Headers`], Seite 169, für Details.

`#CURLOPT_TCP_FASTOPEN`
: Aktiviert/deaktiviert TCP Fast Open. Siehe Abschnitt 5.247 [`easy:SetOpt_TCP_FastOpen`], Seite 170, für Details.

`#CURLOPT_TCP_KEEPALIVE`
: Aktiviert die Tests TCP-Keep-Alive. Siehe Abschnitt 5.248 [`easy:SetOpt_TCP_KeepAlive`], Seite 170, für Details.

`#CURLOPT_TCP_KEEPIDLE`
: Setzt die TCP-Keep-Alive Leerlaufzeit. Siehe Abschnitt 5.249 [`easy:SetOpt_TCP_KeepIdle`], Seite 171, für Details.

`#CURLOPT_TCP_KEEPINTVL`
: Legt den Intervall für TCP-Keep-Alive fest. Siehe Abschnitt 5.250 [`easy:SetOpt_TCP_KeepIntvl`], Seite 171, für Details.

`#CURLOPT_TCP_NODELAY`
: Aktiviert/deaktiviert die Option TCP_NODELAY. Siehe Abschnitt 5.251 [`easy:SetOpt_TCP_NoDelay`], Seite 171, für Details.

`#CURLOPT_TELNETOPTIONS`
: Setzt die benutzerdefinierten Telnet-Optionen. Siehe Abschnitt 5.252 [`easy:SetOpt_TelnetOptions`], Seite 172, für Details.

`#CURLOPT_TFTP_BLKSIZE`
: Setzt die TFTP-Blockgröße. Siehe Abschnitt 5.253 [`easy:SetOpt_TFTP_BlkSize`], Seite 172, für Details.

`#CURLOPT_TFTP_NO_OPTIONS`
: Sendet keine TFTP-Optionsanforderungen. Siehe Abschnitt 5.254 [`easy:SetOpt_TFTP_No_Options`], Seite 173, für Details.

`#CURLOPT_TIMECONDITION`
: Wählt die Bedingung für eine Zeitanfrage aus. Siehe Abschnitt 5.255 [`easy:SetOpt_TimeCondition`], Seite 173, für Details.

`#CURLOPT_TIMEOUT`
: Legt die maximale Zeit fest, die die Anforderung dauern darf. Siehe Abschnitt 5.256 [`easy:SetOpt_Timeout`], Seite 174, für Details.

`#CURLOPT_TIMEOUT_MS`
: Legt die maximale Zeit in ms fest, die die Anforderung dauern darf. Siehe Abschnitt 5.257 [`easy:SetOpt_Timeout_MS`], Seite 174, für Details.

`#CURLOPT_TIMEVALUE`
: Setzt den Zeitwert für bedingtes Verhalten. Siehe Abschnitt 5.258 [`easy:SetOpt_TimeValue`], Seite 175, für Details.

`#CURLOPT_TLSAUTH_PASSWORD`
: Setzt das Passwort für die TLS-Authentifizierung. Siehe Abschnitt 5.259 [`easy:SetOpt_TLSAuth_Password`], Seite 175, für Details.

`#CURLOPT_TLSAUTH_TYPE`
: Legt die TLS-Authentifizierungsmethoden fest. Siehe Abschnitt 5.260 [`easy:SetOpt_TLSAuth_Type`], Seite 176, für Details.

`#CURLOPT_TLSAUTH_USERNAME`
: Legt den Benutzernamen fest, der für die TLS-Authentifizierung verwendet wird. Siehe Abschnitt 5.261 [`easy:SetOpt_TLSAuth_UserName`], Seite 176, für Details.

`#CURLOPT_TRANSFER_ENCODING`
: Fordert die Übertragungscodierung an. Siehe Abschnitt 5.262 [`easy:SetOpt_Transfer_Encoding`], Seite 176, für Details.

`#CURLOPT_TRANSFERTEXT`
: Fordet eine textbasierte Übertragung für FTP an. Siehe Abschnitt 5.263 [`easy:SetOpt_TransferText`], Seite 177, für Details.

`#CURLOPT_UNIX_SOCKET_PATH`
: Setzt den Unix Domain Socket. Siehe Abschnitt 5.264 [`easy:SetOpt_Unix_Socket_Path`], Seite 177, für Details.

`#CURLOPT_UNRESTRICTED_AUTH`
: Sendet Authentifizierungsdaten auch an andere Hosts. Siehe Abschnitt 5.265 [`easy:SetOpt_Unrestricted_Auth`], Seite 178, für Details.

`#CURLOPT_UPLOAD`
: Aktiviert das Hochladen von Daten. Siehe Abschnitt 5.266 [`easy:SetOpt_Upload`], Seite 178, für Details.

`#CURLOPT_URL`
: Gibt die URL an, die in der Anfrage verwendet werden soll. Siehe Abschnitt 5.267 [`easy:SetOpt_URL`], Seite 179, für Details.

`#CURLOPT_USE_SSL`
: Fordert für die Übertragung SSL/TLS an. Siehe Abschnitt 5.268 [`easy:SetOpt_Use_SSL`], Seite 185, für Details.

`#CURLOPT_USERAGENT`
: Setzt den HTTP-User-Agent-Header. Siehe Abschnitt 5.269 [`easy:SetOpt_UserAgent`], Seite 185, für Details.

`#CURLOPT_USERNAME`
: Legt den Benutzername für die Authentifizierung fest. Siehe Abschnitt 5.270 [`easy:SetOpt_UserName`], Seite 186, für Details.

`#CURLOPT_USERPWD`
: Legt den Benutzername und das Passwort für die Authentifizierung fest. Siehe Abschnitt 5.271 [`easy:SetOpt_UserPwd`], Seite 186, für Details.

`#CURLOPT_VERBOSE`
: Schaltet den ausführlichen Modus ein/aus. Siehe Abschnitt 5.272 [`easy:SetOpt_Verbose`], Seite 187, für Details.

`#CURLOPT_WILDCARDMATCH`
: Aktiviert die Übertragung von Verzeichnis-Platzhaltern. Siehe Abschnitt 5.273 [`easy:SetOpt_WildcardMatch`], Seite 188, für Details.

`#CURLOPT_WRITEFUNCTION`
: Setzt den Callback zum Schreiben empfangener Daten. Siehe Abschnitt 5.274 [easy:SetOpt_WriteFunction], Seite 189, für Details.

`#CURLOPT_XOAUTH2_BEARER`
: Gibt den OAuth 2.0 Access Token an. Siehe Abschnitt 5.275 [easy:SetOpt_XOAuth2_Bearer], Seite 190, für Details.

EINGABEN

`option` Optionstyp, der eingestellt werden soll

`parameter`
: Wert, auf den die Option gesetzt werden soll

BEISPIEL

```
e:SetOpt(#CURLOPT_URL, "http://www.hollywood-mal.com")
e:SetOpt(#CURLOPT_VERBOSE, True)
e:SetOpt(#CURLOPT_FOLLOWLOCATION, True)
```

Der obige Code setzt einige Optionen auf einen Easy-Handle.

```
e:SetOpt({URL = "http://www.hollywood-mal.com",
    Verbose = True, FollowLocation = True})
```

Der obige Code entspricht dem ersten Codeausschnitt, aber anstatt die Optionen nacheinander zu setzen, setzt er sie alle auf einmal. Der Effekt ist derselbe, da die Reihenfolge, in der die Optionen gesetzt werden, keine Rolle spielt.

5.61 easy:SetOpt_Abstract_Unix_Socket

BEZEICHNUNG

easy:SetOpt_Abstract_Unix_Socket – setzt einen abstrakten Unix-Domänen-Socket

ÜBERSICHT

```
easy:SetOpt_Abstract_Unix_Socket(path)
```

BESCHREIBUNG

Ermöglicht die Verwendung eines abstrakten Unix-Domänen-Sockets, anstatt eine TCP-Verbindung zu einem Host herzustellen. Der Parameter sollte eine Zeichenkette sein, die den Pfad des Sockets enthält. Der Pfad wird auf `path` gesetzt, dem ein NULL-Byte vorangestellt ist (dies ist die Konvention für abstrakte Sockets, es sollte jedoch betont werden, dass der an diesen Befehl übergebene Pfad keine führende `NULL` enthalten sollte).

Auf nicht unterstützenden Plattformen wird die abstrakte Adresse als leere Zeichenkette interpretiert und schlägt fehl, was zu einem Laufzeitfehler führt.

Diese Option teilt die gleiche Semantik wie `#CURLOPT_UNIX_SOCKET_PATH`. In der Dokumentation finden Sie weitere Details. Intern teilen sich diese beiden Optionen den gleichen Speicherplatz und daher kann nur eine davon pro Aktion eingestellt werden.

EINGABEN

`path` Eingabewert

5.62 easy:SetOpt_Accept_Encoding

BEZEICHNUNG

easy:SetOpt_Accept_Encoding – ermöglicht die automatische Dekompression von HTTP-Downloads

ÜBERSICHT

```
easy:SetOpt_Accept_Encoding(enc)
```

BESCHREIBUNG

Übergibt eine Zeichenkette, die angibt, welche Kodierung Sie wünschen.

Setzt den Inhalt des Accept-Encoding: Header, der in einer HTTP-Anfrage gesendet wird und ermöglicht die Dekodierung einer Antwort, wenn ein Content-Encoding: Header empfangen wird.

libcurl unterstützt potenziell mehrere verschiedene komprimierte Kodierungen, je nachdem, welche Unterstützung eingebaut wurde.

Um Anwendungen zu unterstützen, die sich nicht darum kümmern müssen, welche spezifischen Algorithmen dieser spezielle libcurl-Build unterstützt, erlaubt libcurl, eine Null-Länge-Zeichenkette zu setzen (`""`), um nach einer Accept-Encoding: Header zu fragen, die alle eingebauten unterstützten Kodierungen enthält.

Alternativ können Sie auch genau die Kodierung oder die Liste der Kodierungen angeben, die Sie in der Antwort wünschen. Es werden vier Kodierungen unterstützt: `identity`, d.h. nicht komprimiert, `deflate`, die den Server auffordert, seine Antwort mit dem zlib-Algorithmus zu komprimieren, `gzip`, das den gzip-Algorithmus anfordert und (seit Curl 7.57.0) `br`, der Browser ist. Stellen Sie sie in der Zeichenkette als kommagetrennte Liste der akzeptierten Kodierungen bereit, wie z.B.:

```
"br, gzip, deflate".
```

Setzen Sie `#CURLOPT_ACCEPT_ENCODING` auf Null, um es explizit zu deaktivieren, was dazu führt, dass libcurl keinen Accept-Encoding: Header sendet und empfangene Inhalte nicht automatisch dekomprimiert.

Sie können sich auch dafür entscheiden, den Accept-Encoding: Header einfach in Ihre Anfrage mit `#CURLOPT_HTTPHEADER` aufzunehmen, aber dann gibt es keine automatische Dekompression beim Empfangen von Daten.

Dies ist eine Abfrage, nicht eine Anweisung; der Server kann es tun oder auch nicht. Diese Option muss gesetzt sein (auf einen Nicht-Nil-Wert), sonst wird jede unerwünschte Verschlüsselung durch den Server ignoriert.

Server können mit Content-Encoding antworten, auch ohne einen Accept-Encoding: in der Anfrage. Server können mit einer anderen Content-Kodierung antworten, als in der Anfrage gefordert wurde.

Die Content-Length: Server, die eine komprimierte Antwort senden, sollen die Länge des komprimierten Content angeben, so dass bei aktivierter automatischer Dekodierung die Summe der von den Schreib-Callbacks gemeldeten Bytes nicht übereinstimmen kann (obwohl das Senden der Länge des nicht komprimierten Inhalts ein häufiger Serverfehler ist).

EINGABEN

`enc`	Eingabewert

5.63 easy:SetOpt_AcceptTimeout_MS

BEZEICHNUNG

easy:SetOpt_AcceptTimeout_MS – setzt die Zeitüberschreitung beim Warten auf die erneute Verbindung des FTP-Servers

ÜBERSICHT

```
easy:SetOpt_AcceptTimeout_MS(ms)
```

BESCHREIBUNG

Übergibt einen Wert, welcher der libcurl die maximale Anzahl von Millisekunden angibt, der darauf warten soll, bis sich ein Server bei einer aktiven FTP-Verbindung wieder mit libcurl verbindet.

EINGABEN

`ms` Eingabewert

5.64 easy:SetOpt_Address_Scope

BEZEICHNUNG

easy:SetOpt_Address_Scope – legt den Bereich für lokale IPv6-Adressen fest

ÜBERSICHT

```
easy:SetOpt_Address_Scope(scope)
```

BESCHREIBUNG

Übergibt einen Wert, der den Wert scope_id angibt, der bei der Verbindung mit IPv6-Link-Local- oder Site-Local-Adressen verwendet werden soll.

EINGABEN

`scope` Eingabewert

5.65 easy:SetOpt_Append

BEZEICHNUNG

easy:SetOpt_Append – aktiviert das Anhängen an die Remote-Datei

ÜBERSICHT

```
easy:SetOpt_Append(append)
```

BESCHREIBUNG

Ein auf 1 gesetzter numerischer Parameter weist die Bibliothek an, an die Remote-Datei anzuhängen, anstatt sie zu überschreiben. Dies ist nur nützlich, wenn Sie auf eine FTP-Seite hochladen.

EINGABEN

`append` Eingabewert

5.66 easy:SetOpt_AutoReferer

BEZEICHNUNG

easy:SetOpt_AutoReferer – setzt die automatische Aktualisierung des Referer-Header

ÜBERSICHT

```
easy:SetOpt_AutoReferer(autorefer)
```

BESCHREIBUNG

Übergeben Sie im Parameter `autorefer` eine 1, um dies zu aktivieren. Wenn aktiviert, setzt libcurl automatisch den Referer-Header in HTTP-Anfragen, wo sie einer Positionsumleitung folgt.

EINGABEN

`autorefer`
: Eingabewert

5.67 easy:SetOpt_BufferSize

BEZEICHNUNG

easy:SetOpt_BufferSize – stellt die bevorzugte Empfangspuffergröße ein

ÜBERSICHT

```
easy:SetOpt_BufferSize(size)
```

BESCHREIBUNG

Übergeben Sie einen Wert im Argument `size`, der Ihre bevorzugte Größe (in Bytes) für den Empfangspuffer in libcurl angibt. Der Hauptpunkt dabei wäre, dass der Schreib-Callback häufiger und mit kleineren Teilen aufgerufen wird. Zweitens gibt es bei einigen Protokollen den Vorteil, dass ein größerer Puffer für die Leistung zur Verfügung steht.

Dies wird nur als Anfrage und nicht als Anweisung behandelt. Es kann nicht garantiert werden, dass Sie die angegebene Größe tatsächlich erhalten.

Diese Puffergröße ist standardmäßig `#CURL_MAX_WRITE_SIZE` (16kB). Die maximal zulässige Puffergröße ist `#CURL_MAX_READ_SIZE` (512kB). Die minimal zulässige Puffergröße ist 1024.

EINGABEN

`size` Eingabewert

5.68 easy:SetOpt_CAInfo

BEZEICHNUNG

easy:SetOpt_CAInfo – setzt den Pfad zum Paket der Zertifizierungsstelle (CA)

ÜBERSICHT

```
easy:SetOpt_CAInfo(path)
```

BESCHREIBUNG

Übergibt eine Zeichenkette, die eine Datei mit einem oder mehreren Zertifikaten benennt, um den Peer zu überprüfen.

Wenn `#CURLOPT_SSL_VERIFYPEER` Null ist und Sie die Überprüfung des Zertifikats des Servers vermeiden, muss `#CURLOPT_CAINFO` nicht einmal eine zugängliche Datei angeben.

Diese Option ist standardmäßig auf den Systempfad eingestellt, auf dem das CAcert-Paket von libcurl als gespeichert angenommen wird, wie es zur Aufbauzeit festgelegt wurde.

Wenn curl entgegen die NSS-SSL-Bibliothek erstellt wird, muss das NSS PEM PKCS#11-Modul (libnsspem.so) verfügbar sein, damit diese Option ordnungsgemäß funktioniert. Ab curl 7.55.0, wenn sowohl `#CURLOPT_CAINFO` als auch `#CURLOPT_CAPATH` nicht gesetzt sind, versucht die NSS-verknüpfte libcurl, libnssckbi.so zu laden, der einen umfassenderen Satz von Vertrauensinformationen enthält als von nss-pem unterstützt, da libnssckbi.so auch Informationen über nicht vertrauenswürdige Zertifikate enthält.

(nur iOS und macOS) Wenn curl entgegen eine sichere Übertragung aufgebaut wird, dann wird diese Option aus Gründen der Abwärtskompatibilität mit anderen SSL-Engines unterstützt, sollte aber nicht gesetzt werden. Wenn die Option nicht gesetzt ist, verwendet curl die Zertifikate im System und die Schlüsselkette des Benutzers, um den Peer zu überprüfen, was die bevorzugte Methode zur Überprüfung der Zertifikatskette des Peer's ist.

(nur Schannel) Diese Option wird für Schannel in Windows 7 oder höher mit libcurl 7.60 oder höher unterstützt. Diese Option wird aus Gründen der Abwärtskompatibilität mit anderen SSL-Engines unterstützt; stattdessen wird empfohlen, den Windows-Speicher für Root-Zertifikate zu verwenden (Standard für Schannel).

EINGABEN

`path` Eingabewert

5.69 easy:SetOpt_CAPath

BEZEICHNUNG

easy:SetOpt_CAPath – gibt ein Verzeichnis mit CA-Zertifikaten an

ÜBERSICHT

```
easy:SetOpt_CAPath(capath)
```

BESCHREIBUNG

Übergibt eine Zeichenkette, die ein Verzeichnis mit mehreren CA-Zertifikaten benennt, um den Peer mit zu überprüfen. Wenn libcurl entgegen OpenSSL erstellt wird, muss das Zertifikatsverzeichnis mit dem Dienstprogramm openssl c_rehash vorbereitet werden. Dies ist nur in Kombination mit der Option `#CURLOPT_SSL_VERIFYPEER` sinnvoll.

Die Funktion `#CURLOPT_CAPATH` funktioniert unter Windows anscheinend nicht, da es einige Einschränkungen in openssl gibt.

EINGABEN

`capath` Eingabewert

5.70 easy:SetOpt_CertInfo

BEZEICHNUNG

easy:SetOpt_CertInfo – fordert SSL-Zertifikat-Informationen an

ÜBERSICHT

```
easy:SetOpt_CertInfo(certinfo)
```

BESCHREIBUNG

Übergeben Sie im Parameter `certinfo` eine 1, um den libcurl's Zertifikatskette-Info-Sammler zu aktivieren. Wenn diese Option aktiviert ist, extrahiert libcurl viele Informationen und Daten über die Zertifikate in der SSL-Verbindung verwendeten Zertifikatskette. Diese Daten können dann nach einer Übertragung mit `easy:GetInfo()` und der Option `#CURLINFO_CERTINFO` abgerufen werden.

EINGABEN

`certinfo` Eingabewert

5.71 easy:SetOpt_Chunk_BGN_Function

BEZEICHNUNG

easy:SetOpt_Chunk_BGN_Function – setzt den Callback vor einer Übertragung mit FTP Platzhalter Übereinstimmung

ÜBERSICHT

```
easy:SetOpt_Chunk_BGN_Function(chunk_bgn_callback[, userdata])
```

BESCHREIBUNG

Übergibt eine Callback-Funktion. Diese Callback-Funktion wird von libcurl aufgerufen, bevor ein Teil des Datenstroms übertragen wird (wenn die Übertragung Blöcke unterstützt).

Der Callback erhält zwei Parameter: Der erste Parameter ist eine wie folgt initialisierte Tabelle:

`Filename:`
: Dateiname.

`Filetype:`
: Dateityp.

`Time:` Zeitstempel.

`Perm:` Dateiberechtigungen.

`UID:` Datei UID.

`GID:` Datei GID.

`Size:` Dateigröße.

`HardLinks:`
: Hardlink-Flag.

`Flags:` Zusätzliche Flags.

`Strings:` Dies ist eine Tabelle, die die folgenden Felder enthalten kann (alle sind Zeichenketten):

`Time:` Dateizeit.

`Perm:` Dateiberechtigungen.

`User:` Dateibenutzer.

`Group:` Dateigruppe.

`Target:` Dateiziel.

Der zweite Parameter enthält die Anzahl der verbleibenden Blöcke pro Übertragung. Wenn das Merkmal nicht verfügbar ist, hat der Parameter den Wert Null.

Wenn Sie das optionale Argument `userdata` übergeben, wird der Wert als dritter Parameter an Ihre Callback-Funktion übergeben. Der Parameter `userdata` kann von beliebigem Typ sein.

Dieser Callback ist vorerst nur sinnvoll, wenn die Option `#CURLOPT_WILDCARDMATCH` verwendet wird.

Liefert `#CURL_CHUNK_BGN_FUNC_OK`, wenn alles in Ordnung ist, `#CURL_CHUNK_BGN_FUNC_SKIP`, wenn Sie den konkreten Block überspringen wollen oder `#CURL_CHUNK_BGN_FUNC_FAIL`, um libcurl anzuweisen, dass er anhalten soll, wenn ein Fehler aufgetreten ist.

EINGABEN

`chunk_bgn_callback`
Eingabewert

`userdata` optional: Benutzerdaten, die an die Callback-Funktion übergeben werden sollen

5.72 easy:SetOpt_Chunk_End_Function

BEZEICHNUNG

easy:SetOpt_Chunk_End_Function – setzt den Callback nach einer Übertragung mit FTP-Platzhalter-Übereinstimmung

ÜBERSICHT

```
easy:SetOpt_Chunk_End_Function(chunk_end_callback[, userdata])
```

BESCHREIBUNG

Übergibt eine Callback-Funktion. Diese Funktion wird von libcurl aufgerufen, sobald ein Teil des Datenstroms übertragen (oder übersprungen) wurde.

Der Callback erhält keine Parameter, es sei denn, Sie übergeben das optionale Argument `userdata`. In diesem Fall wird der Wert, den Sie in `userdata` übergeben, als Parameter an Ihre Callback-Funktion übergeben. Der Parameter `userdata` kann von beliebigem Typ sein.

Liefert `#CURL_CHUNK_END_FUNC_OK`, wenn alles in Ordnung ist, oder `#CURL_CHUNK_END_FUNC_FAIL`, um der Lib zu sagen, dass sie stoppen soll, wenn ein Fehler auftritt.

EINGABEN

`chunk_end_callback`
: Eingabewert

`userdata`
: optional: Benutzerdaten, die an die Callback-Funktion übergeben werden sollen

5.73 easy:SetOpt_Connect_Only

BEZEICHNUNG

easy:SetOpt_Connect_Only – stoppt, wenn eine Verbindung zum Zielserver besteht

ÜBERSICHT

```
easy:SetOpt_Connect_Only(only)
```

BESCHREIBUNG

Übergibt einen Wert. Wenn der Parameter gleich 1 ist, weist er die Bibliothek an, alle erforderlichen Proxy-Authentifizierungs- und Verbindungseinstellungen, aber keinen Datentransfer durchzuführen und dann zurückzukehren.

Die Option kann verwendet werden, um eine Verbindung zu einem Server einfach zu testen, ist aber nützlicher, wenn sie mit der Option `#CURLINFO_ACTIVESOCKET` für `easy:GetInfo()` verwendet wird, da die Bibliothek die Verbindung aufbauen kann und die Anwendung dann den zuletzt verwendeten Socket für spezielle Datenübertragungen erhalten kann.

(Beachten Sie, dass `#CURLINFO_ACTIVESOCKET` in hURL 1.0 nicht unterstützt wird.)

EINGABEN

`only` Eingabewert

5.74 easy:SetOpt_Connect_To

BEZEICHNUNG

easy:SetOpt_Connect_To – verbindet mit einem bestimmten Host und Port anstelle des Hosts/Ports der URL

ÜBERSICHT

```
easy:SetOpt_Connect_To(connect_to)
```

BESCHREIBUNG

Übergibt eine Tabelle, die eine Liste von Zeichenketten mit "connect to"-Informationen enthält, die zum Aufbau von Netzwerkverbindungen mit diesem Gerät verwendet werden sollen.

Jede einzelne Zeichenkette sollte im Format HOST:PORT:CONNECT-TO-HOST:CONNECT-TO-PORT geschrieben werden, wobei HOST der Host der Anfrage ist, PORT der Port der Anfrage ist, CONNECT-TO-HOST der Hostname ist, mit dem eine Verbindung hergestellt werden soll, und CONNECT-TO-PORT der Port ist, mit dem eine Verbindung hergestellt werden soll.

Die erste Zeichenkette, die mit dem Host und dem Port der Anfrage übereinstimmt, wird verwendet.

Gepunktete numerische IP-Adressen werden für HOST und CONNECT-TO-HOST unterstützt. Eine numerische IPv6-Adresse muss in [Klammern] geschrieben werden.

Jeder der vier Werte kann leer sein. Wenn der HOST oder PORT leer ist, stimmt der Host oder Port immer überein (der Host oder Port der Anfrage wird ignoriert). Wenn CONNECT-TO-HOST oder CONNECT-TO-PORT leer ist, wird die Funktion "connect to" für den Host oder Port deaktiviert und der Host oder Port der Anfrage wird zum Aufbau der Netzwerkverbindung verwendet.

Diese Option eignet sich, um die Anfrage an einen bestimmten Server zu richten, z.B. an einen bestimmten Clusterknoten in einem Cluster von Servern.

Der "connect to" Host und Port werden nur zum Aufbau der Netzwerkverbindung verwendet. Sie wirken sich NICHT auf den Host und Port aus, die für TLS/SSL (z.B. SNI, Zertifikatsverifizierung) oder für die Anwendungsprotokolle verwendet werden.

Im Gegensatz zu `#CURLOPT_RESOLVE` füllt die Option `#CURLOPT_CONNECT_TO` den DNS-Cache nicht vorab aus und wirkt sich daher nicht auf zukünftige Übertragungen anderer Easy-Handles-Methoden aus, die dem gleichen Multi-Handling hinzugefügt wurden.

Der "connect to" Host und Port werden ignoriert, wenn sie gleich dem Host und dem Port in der Anfrage-URL sind, da die Verbindung zum Host und dem Port in der Anfrage-URL das Standardverhalten ist.

Wenn ein HTTP-Proxy für eine Anfrage mit einem speziellen "connect to"-Host oder -Port verwendet wird und sich der "connect to"-Host oder -Port vom Host und Port der Anfrage unterscheidet, wird der HTTP-Proxy für diese spezielle Anfrage automatisch in den Tunnelmodus geschaltet. Dies ist notwendig, da es nicht möglich ist, sich im normalen (Nicht-Tunnel-)Modus mit einem bestimmten Host oder Port zu verbinden.

EINGABEN

`connect_to`
: Eingabewert

5.75 easy:SetOpt_ConnectTimeout

BEZEICHNUNG

easy:SetOpt_ConnectTimeout – setzt die Zeitüberschreitung für die Verbindungsphase in s

ÜBERSICHT

```
easy:SetOpt_ConnectTimeout(timeout)
```

BESCHREIBUNG

Übergibt einen Wert. Es sollte die maximale Zeit in Sekunden enthalten, die Sie für die Verbindungsphase zum Server benötigen. Dadurch wird die Verbindungsphase nur begrenzt, sie hat nach der Verbindung keine Auswirkungen. Wenn sie sie auf Null setzen, wird zum standardmäßig eingebauten Verbindungs-Timeout - 300 Sekunden - gewechselt. Siehe auch die Option `#CURLOPT_TIMEOUT`.

In Unix-ähnlichen Systemen kann dies dazu führen, dass Signale verwendet werden, es sei denn, `#CURLOPT_NOSIGNAL` ist gesetzt.

Wenn sowohl `#CURLOPT_CONNECTTIMEOUT` als auch `#CURLOPT_CONNECTTIMEOUT_MS` gesetzt sind, wird der zuletzt gesetzte Wert verwendet.

EINGABEN

`timeout` Eingabewert

5.76 easy:SetOpt_ConnectTimeout_MS

BEZEICHNUNG

easy:SetOpt_ConnectTimeout_MS – setzt die Zeitüberschreitung für die Verbindungsphase in ms

ÜBERSICHT

```
easy:SetOpt_ConnectTimeout_MS(timeout)
```

BESCHREIBUNG

Übergibt einen Wert. Es sollte die maximale Zeit in Millisekunden enthalten, die Sie für die Verbindungsphase zum Server benötigen. Dadurch wird die Verbindungsphase nur begrenzt, sie hat nach der Verbindung keine Auswirkungen. Wenn sie sie auf Null setzen, wird zum standardmäßig eingebauten Verbindungs-Timeout - 300 Sekunden - gewechselt. Siehe auch die Option `#CURLOPT_TIMEOUT_MS`.

In Unix-ähnlichen Systemen kann dies dazu führen, dass Signale verwendet werden, es sei denn, `#CURLOPT_NOSIGNAL` ist gesetzt.

Wenn sowohl `#CURLOPT_CONNECTTIMEOUT` als auch `#CURLOPT_CONNECTTIMEOUT_MS` gesetzt sind, wird der zuletzt gesetzte Wert verwendet.

EINGABEN

`timeout` Eingabewert

5.77 easy:SetOpt_Cookie

BEZEICHNUNG

easy:SetOpt_Cookie – setzt den Inhalt des HTTP-Cookie-Headers

ÜBERSICHT

```
easy:SetOpt_Cookie(cookie)
```

BESCHREIBUNG

Übergibt eine Zeichenkette als Parameter. Er wird verwendet, um ein Cookie in der HTTP-Anfrage zu setzen. Das Format der Zeichenkette sollte NAME=CONTENTS sein, wobei NAME der Cookie-Name und CONTENTS das ist, was das Cookie enthalten sollte.

Wenn Sie mehrere Cookies setzen müssen, setzen Sie sie alle mit einer einzigen Option, die wie folgt verknüpft ist: "name1=content1; name2=content2;" etc.

Diese Option setzt den Cookie-Kopfbereich explizit in der ausgehenden Anfrage. Wenn mehrere Anfragen aufgrund von Authentifizierung, nachfolgenden Umleitungen oder ähnlichem ausgeführt werden, werden diese alle an dieses Cookie weitergeleitet.

Die mit dieser Option gesetzten Cookies sind unabhängig von der internen Cookie-Speicherung die von der Cookie-Engine gespeichert und von ihr nicht verändert wird. Wenn Sie die Option Cookie-Engine verwenden, haben Sie entweder ein Cookie mit dem gleichen Namen importiert (z.B.'foo') oder der Server hat eins gesetzt. Dies hat keinen Einfluss auf die hier von Ihnen gesetzten Cookies. Eine Anfrage an den Server sendet sowohl den von der Cookie-Engine gehaltenen "foo" als auch den von der Cookie-Engine gespeicherten "foo", die von dieser Option gehaltene "foo". Um ein Cookie zu setzen, das stattdessen von der Cookie-Engine modifiziert wird, kann durch den Server bei Gebrauch der `#CURLOPT_COOKIELIST` geändert werden.

Wenn Sie diese Option mehrmals verwenden, wird nur die neueste Zeichenkette die vorherigen überschreiben.

Diese Option aktiviert die Cookie-Engine nicht. Verwenden Sie `#CURLOPT_COOKIEFILE` oder `#CURLOPT_COOKIEJAR`, um das automatische Parsen und Senden von Cookies zu ermöglichen.

EINGABEN

`cookie` Eingabewert

5.78 easy:SetOpt_CookieFile

BEZEICHNUNG

easy:SetOpt_CookieFile – setzt den Dateinamen, um Cookies zu lesen

ÜBERSICHT

```
easy:SetOpt_CookieFile(filename)
```

BESCHREIBUNG

Übergibt eine Zeichenkette als Parameter. Es sollte auf den Dateinamen Ihrer Datei zeigen, die Cookie-Daten zum Lesen enthält. Die Cookie-Daten können entweder im alten Netscape-/Mozilla-Cookie-Datenformat oder einfach nur in normale HTTP-Header (Set-Cookie-Stil) abgelegt werden.

Es aktiviert auch die Cookie-Engine, die libcurl parsen und bei späteren Anfragen mit diesem Zugriffsrecht Cookies senden lässt.

Bei einer leeren oder nicht existierenden Datei oder durch Übergabe der leeren Zeichenkette ("") an diese Option können Sie die Cookie-Engine aktivieren, ohne anfängliche Cookies zu lesen. Wenn Sie libcurl mitteilen, dass der Dateiname "-" ist (nur ein einziges Minuszeichen), liest libcurl stattdessen von stdin.

Diese Option liest nur Cookies. Um libcurl dazu zu bringen, Cookies in eine Datei zu schreiben, siehe `#CURLOPT_COOKIEJAR`.

Seien Sie vorsichtig. Wenn Sie diese Option verwenden kann es zu mehreren Übertragungen kommen. Wenn Sie das Set-Cookie-Format verwenden und keine Domäne angeben, wird das Cookie für jede Domäne gesendet (auch wenn Umleitungen befolgt werden) und kann nicht durch ein Cookie vom Server geändert werden.

Wenn ein Server ein Cookie mit dem gleichen Namen setzt, werden beide bei einer zukünftigen Übertragung an diesen Server gesendet. Wahrscheinlich ist das nicht das, was Sie beabsichtigt haben. Um diese Probleme zu beheben, setzen Sie eine Domäne in Set-Cookie (dazu gehören unter anderem Subdomains) oder verwenden Sie das Netscape-Format.

Wenn Sie diese Option mehrmals verwenden, fügen Sie einfach weitere Dateien zum Lesen hinzu. Nachfolgende Dateien werden weitere Cookies hinzufügen.

EINGABEN

`filename` Eingabewert

5.79 easy:SetOpt_CookieJar

BEZEICHNUNG

easy:SetOpt_CookieJar – setzt den Dateinamen, um Cookies zu speichern

ÜBERSICHT

```
easy:SetOpt_CookieJar(filename)
```

BESCHREIBUNG

In `filename` wird eine Zeichenkette übergeben. Dadurch schreibt libcurl alle intern bekannten Cookies in die angegebene Datei, wenn `easy:Close()` aufgerufen wird. Wenn keine Cookies bekannt sind, wird keine Datei erstellt. Geben Sie als Dateinamen `"-"` an, damit die Cookies stattdessen in stdout geschrieben werden. Wenn Sie diese Option verwenden, werden auch Cookies für diese Sitzung aktiviert. Wenn Sie also z.B. einem Standort folgen, werden entsprechende Cookies dementsprechend gesendet.

Beachten Sie, dass libcurl keine Cookies aus der Cookiesammlung liest. Wenn Sie Cookies aus einer Datei lesen möchten, verwenden Sie `#CURLOPT_COOKIEFILE`.

Wenn die Cookie-Sammlungsdatei nicht erstellt oder beschrieben werden kann (wenn `easy:Close()` aufgerufen wird), wird libcurl keinen Fehler melden und kann dies auch nicht. Wenn Sie `#CURLOPT_VERBOSE` oder `#CURLOPT_DEBUGFUNCTION` verwenden, wird eine Warnung angezeigt, aber das ist die einzige sichtbare Rückmeldung, die Sie über diese möglicherweise prekäre Situation erhalten.

Seit 7.43.0 werden Cookies, die im Set-Cookie-Format ohne Domain-Namen importiert wurden, von dieser Option nicht mehr exportiert.

EINGABEN

`filename` Eingabewert

5.80 easy:SetOpt_CookieList

BEZEICHNUNG

easy:SetOpt_CookieList – fügt hinzu oder manipuliert von im Speicher befindliche Cookies

ÜBERSICHT

```
easy:SetOpt_CookieList(cookie)
```

BESCHREIBUNG

Übergibt eine Cookie-Zeichenkette.

Ein solches Cookie kann entweder eine einzelne Zeile im Netscape-/Mozilla-Format oder einfach nur ein normaler HTTP-ähnlicher Header (Set-Cookie: ....) Format sein. Dadurch wird auch die Cookie-Engine aktiviert und dieses einzelne Cookie wird dem internen Cookie-Speicher hinzugefügt.

Seien Sie vorsichtig, wenn Sie diese Option verwenden, da es zu mehreren Übertragungen kommen kann. Wenn Sie das Set-Cookie-Format verwenden und keine Domäne angeben, wird das Cookie für jede Domain gesendet (auch wenn Umleitungen befolgt werden) und nicht durch ein vom Server gesetztes Cookie geändert werden kann. Wenn ein Server ein Cookie mit dem gleichen Namen setzt (oder möglicherweise haben Sie eines importiert), dann werden beide auf eine zukünftige Übertragung an diesen Server gesendet. Wahrscheinlich nicht das, was Sie beabsichtigt haben. Um diese Probleme zu beheben, setzen Sie eine Domäne unter Set-Cookie (dabei werden Subdomains einbezogen) oder verwenden Sie das Netscape-Format, wie im BEISPIEL dargestellt.

Zusätzlich stehen Befehle zur Verfügung, die Aktionen ausführen, wenn Sie genau diese Zeichenketten übergeben:

`ALL` Löscht alle im Speicher befindlichen Cookies

`SESS` Löscht alle Sitzungs-Cookies, die sich im Speicher befinden

`FLUSH` Schreibt alle bekannten Cookies in die durch `#CURLOPT_COOKIEJAR` angegebene Datei

`RELOAD` Lädt alle Cookies aus den Dateien, die durch `#CURLOPT_COOKIEFILE` angegeben sind

EINGABEN

`cookie` Eingabewert

5.81 easy:SetOpt_CookieSession

BEZEICHNUNG

easy:SetOpt_CookieSession – startet eine neue Cookie-Sitzung

ÜBERSICHT

```
easy:SetOpt_CookieSession(init)
```

BESCHREIBUNG

Übergibt einen Wert, der auf 1 gesetzt ist, um ihn als neue Cookie-"Session" (Sitzung) zu markieren. Es zwingt libcurl, alle zu ladenden Cookies zu ignorieren, die "Session Cookies" der vorherigen Sitzung sind. Standardmäßig speichert und lädt libcurl immer alle Cookies, unabhängig davon, ob es sich um Sitzungs-Cookies handelt oder nicht. Sitzungs-Cookies sind Cookies ohne Verfallsdatum und sollen nur für diese "Session" aktiv und vorhanden sein.

Eine "Session" ist in der Regel im Browserverzeichnis definiert, solange Sie Ihren Browser mehr oder weniger in Betrieb haben.

EINGABEN

`init` Eingabewert

5.82 easy:SetOpt_CRLF

BEZEICHNUNG

easy:SetOpt_CRLF – aktiviert/deaktiviert die CRLF-Konvertierung

ÜBERSICHT

```
easy:SetOpt_CRLF(conv)
```

BESCHREIBUNG

Übergibt einen Wert. Wenn der Wert auf 1 (eins) gesetzt ist, konvertiert libcurl Unix-Transfers in CRLF-Transfers bei der Übertragung. Deaktivieren Sie diese Option wieder, indem Sie den Wert auf 0 (Null) setzen.

Dies ist eine alte Option mit fragwürdigem Nutzen.

EINGABEN

`conv` Eingabewert

5.83 easy:SetOpt_CRLFile

BEZEICHNUNG

easy:SetOpt_CRLFile – gibt eine Datei für Zertifikatssperrlisten an

ÜBERSICHT

```
easy:SetOpt_CRLFile(file)
```

BESCHREIBUNG

Übergibt eine Zeichenkette, die in `file` eine Datei mit der Verkettung von CRL (im PEM-Format) benennt, die bei der Zertifikatsvalidierung verwendet werden soll, die während des SSL-Austauschs stattfindet.

Wenn curl für die Verwendung von NSS oder GnuTLS erstellt wird, gibt es keine Möglichkeit, die Verwendung der übergebenen CRL zu beeinflussen, um den Verifikationsprozess zu unterstützen. Wenn libcurl mit OpenSSL-Unterstützung erstellt wird, sind `X509_V_FLAG_CRL_CHECK` und `X509_V_FLAG_CRL_CHECK_ALL` beide gesetzt und erfordern eine CRL-Prüfung gegen alle Elemente der Zertifikatskette, wenn eine CRL-Datei übergeben wird.

Diese Option ist nur sinnvoll, wenn sie in Kombination mit der Option `#CURLOPT_SSL_VERIFYPEER` verwendet wird.

Mit der Option wird ein bestimmter Fehlercode (`#CURLE_SSL_CRL_CRL_BADFILE`) definiert. Er wird zurückgegeben, wenn der SSL-Austausch fehlschlägt, weil die CRL-Datei nicht geladen werden kann. Ein Fehler bei der Zertifikatsprüfung aufgrund einer Widerrufsinformation in der GRL löst diesen Fehler nicht aus.

EINGABEN

`file` Eingabewert

5.84 easy:SetOpt_CustomRequest

BEZEICHNUNG

easy:SetOpt_CustomRequest – setzt die benutzerdefinierte Zeichenkette für die Anforderung

ÜBERSICHT

```
easy:SetOpt_CustomRequest(request)
```

BESCHREIBUNG

Übergibt eine Zeichenkette als Parameter.

Wenn Sie die Anforderungsmethode ändern, indem Sie `#CURLOPT_CUSTOMREQUEST` auf etwas setzen, ändern Sie nicht, wie sich libcurl in Bezug auf die jeweilige Anforderungsmethode verhält, sondern nur die tatsächlich in der Anforderung gesendete Zeichenkette.

Stellen Sie den internen Standard wieder her, indem Sie diese auf Null setzen.

Diese Option kann verwendet werden, um die Anforderung zu spezifizieren:

`HTTP` Anstelle von GET oder HEAD bei der Ausführung von HTTP-basierten Anfragen. Dies ist besonders nützlich, z.B. für die Ausführung eines HTTP DELETE-Requests.

Zum Beispiel:

Wenn Sie libcurl anweisen, eine HEAD-Anforderung auszuführen, aber dann ein GET angeben, obwohl eine benutzerdefinierte Anforderung immer noch so funktioniert, als ob ein HEAD gesendet worden wäre. Um zu einem richtigen HEAD zu wechseln, verwenden Sie `#CURLOPT_NOBODY`, um zu einem richtigen POST zu wechseln, verwenden Sie `#CURLOPT_POST` oder `#CURLOPT_POSTFIELDS` und um zu einem richtigen GET zu wechseln verwenden Sie `#CURLOPT_HTTPGET`.

Viele Leute haben diese Option zu Unrecht benutzt, um die gesamte Anfrage durch ihre eigene zu ersetzen, einschließlich mehrerer Header und POST-Inhalte. Obwohl das in vielen Fällen funktionieren könnte, führt es dazu, dass libcurl ungültige Anfragen sendet und den Remote-Server möglicherweise stark verwirrt. Verwenden Sie `#CURLOPT_POST` und `#CURLOPT_POSTFIELDS`, um POST-Daten zu setzen. Verwenden Sie `#CURLOPT_HTTPHEADER`, um den von libcurl gesendeten Satz von Headern zu ersetzen oder zu erweitern. Verwenden Sie `#CURLOPT_HTTP_VERSION`, um die HTTP-Version zu ändern.

`FTP` Anstelle von LIST und NLST bei der Durchführung von FTP-Verzeichnisauflistungen.

`IMAP` Anstelle von LIST bei der Ausgabe von IMAP-basierten Anfragen.

`POP3` Anstelle von LIST und RETR bei der Ausgabe von POP3-basierten Anfragen.

Zum Beispiel:

Wenn Sie libcurl anweisen, eine benutzerdefinierte Anforderung zu verwenden, verhält es sich so, als ob ein LIST- oder RETR-Befehl gesendet wurde, wo libcurl erwartet, dass Daten vom Server zurückgegeben werden. Daher sollte `#CURLOPT_NOBODY` verwendet werden, wenn Befehle wie DELE und NOOP angegeben werden.

`SMTP` Anstelle von HELP oder VRFY bei der Ausgabe von SMTP-basierten Anfragen.

Zum Beispiel:

Normalerweise wird eine mehrzeilige Antwort zurückgegeben, die in Verbindung mit `#CURLOPT_MAIL_RCPT` verwendet werden kann, um eine EXPN-Anfrage anzugeben. Wenn die Option `#CURLOPT_NOBODY` angegeben ist, kann die Anforderung zur Ausgabe von NOOP- und RSET-Befehlen verwendet werden.

EINGABEN

`request` Eingabewert

5.85 easy:SetOpt_DebugFunction

BEZEICHNUNG

easy:SetOpt_DebugFunction – setzt die Debug-Callback-Funktion

ÜBERSICHT

```
easy:SetOpt_DebugFunction(debug_callback[, userdata])
```

BESCHREIBUNG

Übergibt eine Callback-Funktion. Diese Funktion ersetzt die Standard-Debug-Funktion, die verwendet wird, wenn `#CURLOPT_VERBOSE` in Kraft ist. Dieser Callback erhält zwei Parameter: Der erste Parameter gibt die Art der Debug-Informationen an, die sich im zweiten Parameter befinden. Dies kann derzeit einer der folgenden speziellen Werte sein:

`#CURLINFO_TEXT`
: Die Daten sind Informationstexte.

`#CURLINFO_HEADER_IN`
: Die Daten sind Header- (oder Header-ähnliche) Daten, die vom Peer empfangen werden.

`#CURLINFO_HEADER_OUT`
: Die Daten sind Header- (oder Header-ähnliche) Daten, die an dem Peer gesendet werden.

`#CURLINFO_DATA_IN`
: Die Daten sind Protokolldaten, die vom Peer empfangen werden.

`#CURLINFO_DATA_OUT`
: Die Daten sind Protokolldaten, die an den Peer gesendet werden.

`#CURLINFO_SSL_DATA_OUT`
: Bei den Daten handelt es sich um SSL/TLS-Daten (binär), die an den Peer gesendet werden.

`#CURLINFO_SSL_DATA_IN`
: Bei den Daten handelt es sich um SSL/TLS (binäre) Daten, die vom Peer empfangen werden.

Der zweite Parameter, der an Ihre Callback-Funktion übergeben wird, ist eine Zeichenkette, die die aktuellen Debug-Informationen enthält.

Wenn Sie das optionale Argument `userdata` übergeben, wird der Wert, den Sie in `userdata` übergeben, als dritter Parameter an Ihre Callback-Funktion übergeben. Der Parameter `userdata` kann von beliebigem Typ sein.

Ihr Debug-Callback sollte nichts zurückgeben.

EINGABEN

`debug_callback`
: Eingabewert

`userdata`
: optional: Benutzerdaten, die an die Callback-Funktion übergeben werden sollen

5.86 easy:SetOpt_Default_Protocol

BEZEICHNUNG

easy:SetOpt_Default_Protocol – setzt das Standardprotokoll bei fehlendem Schemanamen

ÜBERSICHT

```
easy:SetOpt_Default_Protocol(protocol)
```

BESCHREIBUNG

Diese Option weist libcurl an, `protocol` zu verwenden, wenn der URL ein Schemaname fehlt.

Verwenden Sie einen dieser Protokoll-(Schema-)Namen:

```
dict
file
ftp
ftps
gopher
http
https
imap
imaps
ldap
ldaps
pop3,
pop3s
rtsp
scp
sftp
smb
smbs
smtp
smtps
```

```
telnet
tftp
```

Ein unbekanntes oder nicht unterstütztes Protokoll verursacht den Fehler `#CURLE_UNSUPPORTED_PROTOCOL`, wenn libcurl eine schemaabweichende URL analysiert. Das Parsen erfolgt, wenn `easy:Perform()` oder `multi:Perform()` aufgerufen wird. Die von libcurl unterstützten Protokolle variieren je nachdem, wie sie erstellt wurden. Verwenden Sie `hurl.VersionInfo()`, wenn Sie eine Liste von Protokollnamen benötigen, die vom Aufbau der von Ihnen verwendeten libcurl unterstützt wird.

Diese Option ändert das Standard-Proxy-Protokoll (http) nicht.

Ohne diese Option würde libcurl eine Schätzung basierend auf dem Host vornehmen, siehe `#CURLOPT_URL` für Details.

EINGABEN

`protocol` Eingabewert

5.87 easy:SetOpt_DirListOnly

BEZEICHNUNG

easy:SetOpt_DirListOnly – fragt nur nach Namen in einer Verzeichnisliste

ÜBERSICHT

```
easy:SetOpt_DirListOnly(listonly)
```

BESCHREIBUNG

Bei FTP- und SFTP-basierten URLs weist ein auf 1 gesetzter Parameter die Bibliothek an, die Namen der Dateien in einem Verzeichnis aufzulisten, anstatt eine vollständige Verzeichnisliste durchzuführen, die normalerweise Dateigrößen, Datumsangaben usw. enthält.

Für POP3 weist der Parameter 1 die Bibliothek an, die E-Mail-Nachricht oder Nachrichten auf dem POP3-Server aufzulisten. Dies kann verwendet werden, um das Standardverhalten von libcurl in Kombination mit einer URL, die eine Nachrichten-ID enthält, zu ändern und eine "scan listing" durchzuführen, mit der dann die Größe einer E-Mail bestimmt werden kann.

Hinweis: Bei FTP wird dadurch ein NLST-Befehl an den FTP-Server gesendet. Beachten Sie, dass einige FTP-Server in ihrer Antwort auf NLST nur Dateien auflisten. Sie enthalten möglicherweise keine Unterverzeichnisse und symbolische Links.

Wenn Sie diese Option auf 1 setzen, wird auch dann eine Verzeichnisliste angezeigt, wenn die URL nicht mit einem Schrägstrich endet, was ansonsten erforderlich ist.

Verwenden Sie diese Option NICHT, wenn Sie auch `#CURLOPT_WILDCARDMATCH` verwenden, da dies diesen Befehl dann effektiv beeinträchtigt.

EINGABEN

`listonly` Eingabewert

5.88 easy:SetOpt_DNS_Cache_Timeout

BEZEICHNUNG

easy:SetOpt_DNS_Cache_Timeout – legt die Lebensdauer für DNS-Cache-Einträge fest

ÜBERSICHT

```
easy:SetOpt_DNS_Cache_Timeout(age)
```

BESCHREIBUNG

Wenn Sie einen Wert übergeben, wird der Wartezeitraum in Sekunden festgelegt. Namensauflösungen werden gespeichert und für diese Anzahl von Sekunden verwendet. Setzen Sie den Wert auf Null, um die Zwischenspeicherung vollständig zu deaktivieren, oder auf -1, damit die zwischengespeicherten Einträge für immer erhalten bleiben. Standardmäßig speichert libcurl diese Informationen für 60 Sekunden im Cache.

Die Namensauflösungsfunktionen verschiedener libc-Implementierungen lesen die Servernamen-Informationen nur dann erneut, wenn dies ausdrücklich angegeben wird (z.B. durch Aufrufen von `res_init`). Dies kann dazu führen, dass libcurl weiterhin den älteren Server verwendet, auch wenn DHCP die Serverinformationen aktualisiert hat. Dies kann für den gelegentlichen Benutzer der libcurl-app wie ein DNS-Cache-Problem aussehen.

Beachten Sie, dass DNS-Einträge eine "TTL"-Eigenschaft haben, die von libcurl jedoch nicht verwendet wird. Dieses Zeitlimit für den DNS-Cache ist nur spekulativ, da ein Name in der Zukunft für eine gewisse Zeit in dieselbe Adresse aufgelöst wird.

EINGABEN

`age` Eingabewert

5.89 easy:SetOpt_DNS_Interface

BEZEICHNUNG

easy:SetOpt_DNS_Interface – stellt die Schnittstelle so ein, dass über DNS kommuniziert wird

ÜBERSICHT

```
easy:SetOpt_DNS_Interface(ifname)
```

BESCHREIBUNG

Übergibt eine Zeichenkette als Parameter. Legen Sie den Namen der Netzwerkschnittstelle fest, an die sich der DNS-Redundanz-Server binden soll. Dies muss ein Interface-Name (keine Adresse) sein. Setzen Sie diese Option auf Null, um die Standardeinstellung zu verwenden (nicht an eine bestimmte Schnittstelle binden).

EINGABEN

`ifname` Eingabewert

5.90 easy:SetOpt_DNS_Local_IP4

BEZEICHNUNG

easy:SetOpt_DNS_Local_IP4 – setzt die IPv4-Adresse, an die DNS-Auflösungen gebunden werden sollen

ÜBERSICHT

```
easy:SetOpt_DNS_Local_IP4(address)
```

BESCHREIBUNG

Stellt die lokale IPv4 `adresse` ein, an die sich der Resolver binden soll. Das Argument sollte vom Typ Zeichenkette sein und eine einzelne numerische IPv4-Adresse als Zeichenkette enthalten. Setzen Sie diese Option auf Null, um die Standardeinstellung zu verwenden (nicht an eine bestimmte IP-Adresse binden).

EINGABEN

`address` Eingabewert

5.91 easy:SetOpt_DNS_Local_IP6

BEZEICHNUNG

easy:SetOpt_DNS_Local_IP6 – setzt die IPv6-Adresse, an die DNS-Auflösungen gebunden werden sollen

ÜBERSICHT

```
easy:SetOpt_DNS_Local_IP6(address)
```

BESCHREIBUNG

Stellt die lokale IPv6 `adresse` ein, an die sich der Resolver binden soll. Das Argument sollte vom Typ Zeichenkette sein und eine einzelne IPv6-Adresse als Zeichenkette enthalten. Setzen Sie diese Option auf Null, um die Standardeinstellung zu verwenden (nicht an eine bestimmte IP-Adresse binden).

EINGABEN

`address` Eingabewert

5.92 easy:SetOpt_DNS_Servers

BEZEICHNUNG

easy:SetOpt_DNS_Servers – legt bevorzugte DNS-Server fest

ÜBERSICHT

```
easy:SetOpt_DNS_Servers(servers)
```

BESCHREIBUNG

Übergibt eine Zeichenkette, die anstelle der Systemvoreinstellung die Liste der zu verwendenden DNS-Server enthält. Das Format der Option `servers` ist:

```
host[:port][,host[:port]]...
```

Zum Beispiel:

```
192.168.1.100, 192.168.1.101, 3.4.5.6
```

EINGABEN

`servers` Eingabewert

5.93 easy:SetOpt_DNS_Use_Global_Cache

BEZEICHNUNG

easy:SetOpt_DNS_Use_Global_Cache – aktiviert/deaktiviert den globalen DNS-Cache

ÜBERSICHT

```
easy:SetOpt_DNS_Use_Global_Cache(enable)
```

BESCHREIBUNG

Übergibt einen Wert. Wenn der Wert von `enable` 1 ist, weist er curl an, einen globalen DNS-Cache zu verwenden, der zwischen Erstellungen und Löschungen von Easy-Handler bestehen bleibt. Dies ist nicht thread-safe und verwendet eine globale Variable.

WARNUNG: Diese Option gilt als veraltet. Benutzen Sie die nicht, wechseln Sie stattdessen zur Share-Schnittstelle! Siehe `#CURLOPT_SHARE` und `hurl.Share()`.

EINGABEN

`enable` Eingabewert

5.94 easy:SetOpt_EGDSocket

BEZEICHNUNG

easy:SetOpt_EGDSocket – stellt den EGD-Socketpfad ein

ÜBERSICHT

```
easy:SetOpt_EGDSocket(path)
```

BESCHREIBUNG

Übergibt eine Zeichenkette an den Pfadnamen des Entropy Gathering Daemon Socket. Sie wird verwendet, um den Zufallsgenerator für SSL zu setzen.

EINGABEN

`path` Eingabewert

5.95 easy:SetOpt_Expect_100_Timeout_MS

BEZEICHNUNG

easy:SetOpt_Expect_100_Timeout_MS – setzt die Zeitüberschreitung bei der Antwort Expect: 100-continue

ÜBERSICHT

```
easy:SetOpt_Expect_100_Timeout_MS(milliseconds)
```

BESCHREIBUNG

Übergeben Sie einen Wert, um libcurl die Anzahl der Millisekunden mitzuteilen, die auf eine Serverantwort mit dem HTTP-Status 100 (Continue/Fortfahren), 417 (Expectation

Failed/Erwartung fehlgeschlagen) oder ähnlichem gewartet werden soll, nachdem eine HTTP-Anforderung gesendet wurde, die eine Expect: 100-continue des Headers enthält. Wenn dieses Zeitlimit überschritten wird, bevor eine Antwort empfangen wird, wird der Anforderungs-Body trotzdem gesendet.

EINGABEN

`milliseconds`
: Eingabewert

5.96 easy:SetOpt_FailOnError

BEZEICHNUNG

easy:SetOpt_FailOnError – setzt den Anforderungsfehler bei HTTP-Antwort >= 400

ÜBERSICHT

```
easy:SetOpt_FailOnError(fail)
```

BESCHREIBUNG

Ein auf 1 gesetzter Werteparameter weist die Bibliothek an, die Anforderung nicht zu erfüllen, wenn der zurückgegebene HTTP-Code gleich oder größer 400 ist. Die Standardaktion wäre, die Seite normal zurückzugeben und diesen Code zu ignorieren.

Diese Methode ist nicht ausfallsicher und es gibt Fälle, in denen erfolglose Antwortcodes durchschlüpfen, insbesondere bei der Authentifizierung (Antwortcodes 401 und 407).

Es können einige Header-Daten übertragen werden, bevor diese Situation erkannt wird, z.B. wenn ein "100-continue" als Antwort auf einen POST/PUT empfangen wird und ein 401 oder 407 unmittelbar danach empfangen wird.

Wenn diese Option verwendet wird und ein Fehler erkannt wird, wird die Verbindung geschlossen und `#CURLE_HTTP_RETURNED_ERROR` zurückgegeben.

EINGABEN

`fail` Eingabewert

5.97 easy:SetOpt_FileTime

BEZEICHNUNG

easy:SetOpt_FileTime – liefert die Änderungszeit der Remote-Datenquelle

ÜBERSICHT

```
easy:SetOpt_FileTime(gettime)
```

BESCHREIBUNG

Übergibt einen Wert. Wenn es 1 ist, versucht libcurl, die Änderungszeit des Remote-Dokuments in diesem Vorgang zu ermitteln. Dies erfordert, dass der Remote-Server die Uhrzeit oder Antworten auf einen Befehl zur Zeitabfrage sendet. Der Befehl `easy:GetInfo()` mit dem Argument `#CURLINFO_FILETIME` kann nach einer Übertragung verwendet werden, um die empfangene Zeit (falls vorhanden) zu extrahieren.

EINGABEN

`gettime` Eingabewert

5.98 easy:SetOpt_FNMatch_Function

BEZEICHNUNG

easy:SetOpt_FNMatch_Function – setzt die Callback-Platzhalterabgleich-Funktion

ÜBERSICHT

```
easy:SetOpt_FNMatch_Function(fnmatch_callback[, userdata])
```

BESCHREIBUNG

Übergibt eine Callback-Funktion, die für den Platzhalterabgleich verwendet wird. Die Callback-Funktion erhält zwei Parameter: Der erste Parameter ist eine Zeichenkette, die das Muster enthält, der zweite Parameter ist die zu prüfende Zeichenkette.

Wenn Sie das optionale Argument `userdata` übergeben, wird der Wert als dritter Parameter an Ihre Callback-Funktion übergeben. Der Parameter `userdata` kann von beliebigem Typ sein.

Liefert `#CURL_FNMATCHFUNC_MATCH`, wenn das Muster mit der Zeichenkette übereinstimmt, `#CURL_FNMATCHFUNC_NOMATCH`, wenn nicht oder `#CURL_FNMATCHFUNC_FAIL`, wenn ein Fehler aufgetreten ist.

EINGABEN

`fnmatch_callback`
Eingabewert

`userdata` optional: Benutzerdaten, die an die Callback-Funktion übergeben werden sollen

5.99 easy:SetOpt_FollowLocation

BEZEICHNUNG

easy:SetOpt_FollowLocation – folgt HTTP 3xx Umleitungen

ÜBERSICHT

```
easy:SetOpt_FollowLocation(enable)
```

BESCHREIBUNG

Ein Parameter, der auf 1 gesetzt ist, weist die Bibliothek an, jedem Location: Header zu folgen, die der Server als Teil eines HTTP-Headers in einer 3xx-Antwort sendet. Der Location: Header kann eine relative oder absolute URL angeben, dem er folgen soll.

libcurl wird eine weitere Anfrage für die neue URL stellen und dem neuen Location: Header folgen: Den ganzen Weg, bis keine solchen Header mehr zurückgegeben werden. `#CURLOPT_MAXREDIRS` kann verwendet werden, um die Anzahl der Umleitungen zu begrenzen, die libcurl folgen wird.

libcurl begrenzt die Protokolle, denen es automatisch folgt. Die akzeptierten Protokolle werden mit `#CURLOPT_REDIR_PROTOCOLS` gesetzt. Standardmäßig erlaubt libcurl alle Umleitungsprotokolle, außer denen, die aus Sicherheitsgründen deaktiviert sind: Seit 7.19.4 sind FILE und SCP deaktiviert, seit 7.40.0 sind auch SMB und SMBS deaktiviert.

Wenn Sie einem Location: Header folgen, bestimmt der 3xx-Antwortcode, der ihn umgeleitet hat, auch, welche Anforderungsmethode er in der nachfolgenden Anforderung verwenden wird: Für 301, 302 und 303 Antworten wechselt libcurl die Methode auf GET, es sei denn, `#CURLOPT_POSTREDIR` weist libcurl anders an. Alle anderen 3xx-Codes lassen libcurl die gleiche Methode wieder senden.

Für Benutzer, die der Meinung sind, dass die bestehende Standortverfolgung zu naiv oder zu einfach ist oder einfach nur Funktionen fehlen, ist es sehr einfach, stattdessen Ihre eigene Umleitungs-Verfolgungslogik mit der Verwendung der #CURLINFO_REDIRECT_URL-Option vom Befehl `easy:GetInfo()` zu implementieren, anstatt `#CURLOPT_FOLLOWLOCATION` zu verwenden.

EINGABEN

`enable` Eingabewert

5.100 easy:SetOpt_Forbid_Reuse

BEZEICHNUNG

easy:SetOpt_Forbid_Reuse – schließt die Verbindung sofort, nachdem die Übertragung beendet ist

ÜBERSICHT

```
easy:SetOpt_Forbid_Reuse(close)
```

BESCHREIBUNG

Übergibt einen Wert. Setzen Sie `close` auf 1, damit libcurl die Verbindung explizit schließt, wenn Sie mit der Übertragung fertig sind. Normalerweise hält libcurl alle Verbindungen nach einem Transfer offen, falls ein nachfolgender folgt, der sie wiederverwenden kann. Diese Option sollte mit Vorsicht verwendet werden und nur, wenn Sie verstehen, was sie bewirkt, da sie die Leistung ernsthaft beeinträchtigen kann.

Wird auf 0 gesetzt, damit libcurl die Verbindung für eine mögliche spätere Wiederverwendung offen hält (Standardverhalten).

EINGABEN

`close` Eingabewert

5.101 easy:SetOpt_Fresh_Connect

BEZEICHNUNG

easy:SetOpt_Fresh_Connect – erzwingt, dass eine neue Verbindung verwendet wird

ÜBERSICHT

```
easy:SetOpt_Fresh_Connect(fresh)
```

BESCHREIBUNG

Übergibt einen Wert. Setzen Sie den Wert auf 1, um beim nächsten Transfer eine neue (frische) Verbindung zu verwenden, anstatt zu versuchen, eine bestehende wieder zu verwenden. Diese Option sollte mit Vorsicht und nur dann verwendet werden, wenn Sie verstehen, was sie bewirkt, da sie die Leistung ernsthaft beeinträchtigen kann.

Die zugehörige Funktionalität ist `#CURLOPT_FORBID_REUSE`, die sicherstellt, dass die Verbindung nach der Verwendung geschlossen wird, damit sie nicht wiederverwendet wird.

Setzen Sie `fresh` auf 0, damit libcurl versucht, eine bestehende Verbindung wiederzuverwenden (Standardverhalten).

EINGABEN

`fresh` Eingabewert

5.102 easy:SetOpt_FTP_Account

BEZEICHNUNG

easy:SetOpt_FTP_Account – sendet Kontoinformationen mit ACCT an FTP-Server

ÜBERSICHT

```
easy:SetOpt_FTP_Account(account)
```

BESCHREIBUNG

Übergibt eine Zeichenkette (oder Null zum Deaktivieren). Wenn ein FTP-Server nach Eingabe von Benutzername und Passwort nach "account data" fragt, werden diese Daten mit dem Befehl ACCT gesendet.

EINGABEN

`account` Eingabewert

5.103 easy:SetOpt_FTP_Alternative_To_User

BEZEICHNUNG

easy:SetOpt_FTP_Alternative_To_User – legt fest, dass FTP anstelle von USER verwendet wird

ÜBERSICHT

```
easy:SetOpt_FTP_Alternative_To_User(cmd)
```

BESCHREIBUNG

Übergibt eine Zeichenkette als Parameter, die zur Authentifizierung verwendet wird, wenn die übliche FTP-Übertragung von "USER user" und "PASS password" fehlschlägt. Derzeit ist bekannt, dass dies nur erforderlich ist, wenn eine Verbindung zum Secure Transport FTPS-Server von Tumbleweed unter Verwendung von Client-Zertifikaten zur Authentifizierung hergestellt wird.

EINGABEN

`cmd` Eingabewert

5.104 easy:SetOpt_FTP_Create_Missing_Dirs

BEZEICHNUNG

easy:SetOpt_FTP_Create_Missing_Dirs – erstellt fehlende Verzeichnisse für FTP und SFTP

ÜBERSICHT

```
easy:SetOpt_FTP_Create_Missing_Dirs(create)
```

BESCHREIBUNG

Übergibt einen Wert, der libcurl anweist, das Verzeichnis zu erstellen. Wenn der Wert `CURLFTP_CREATE_DIR` ist, wird libcurl versuchen, ein Remote-Verzeichnis zu erstellen, in das es nicht verschoben (`"move"`) werden kann.

Für FTP-Anfragen bedeutet dies, dass ein CWD-Befehl fehlschlägt. CWD ist der Befehl, der das Arbeitsverzeichnis ändert.

Für SFTP-Anfragen versucht libcurl, das externe Verzeichnis zu erstellen, wenn es kein Zugriffsrecht auf den Zielort erhalten kann. Die Erstellung schlägt fehl, wenn eine Datei mit dem gleichen Namen wie das zu erstellende Verzeichnis bereits existiert oder fehlende Berechtigungen die Erstellung verhindern.

Wenn `create` auf `CURLFTP_CREATE_DIR_RETRY` gesetzt wird, weist libcurl an, den CWD-Befehl erneut zu versuchen, wenn der nachfolgende MKD-Befehl fehlschlägt. Dies ist besonders nützlich, wenn Sie viele gleichzeitige Verbindungen zum gleichen Server durchführen und alle diese Optionen aktiviert haben. Da dann CWD zuerst fehlschlägt, aber eine andere Verbindung MKD vor dieser Verbindung erstellt und somit MKD fehlschlägt, aber der Versuch mit CWD funktioniert!

EINGABEN

`create` Eingabewert

5.105 easy:SetOpt_FTP_FileMethod

BEZEICHNUNG

easy:SetOpt_FTP_FileMethod – wählt das Verzeichnisdurchlaufverfahren für FTP aus

ÜBERSICHT

```
easy:SetOpt_FTP_FileMethod(method)
```

BESCHREIBUNG

Übergibt einen Wert in `method`, der libcurl mitteilt, mit welchem Verfahren eine Datei auf einem FTP(S)-Server erreicht werden soll.

Diese Option existiert, weil einige Serverimplementierungen nicht den Standards entsprechen, die funktionieren sollen.

Das Argument sollte eine der folgenden Alternativen sein:

`CURLFTPMETHOD_MULTICWD`

libcurl führt eine einzige CWD-Operation für jeden Pfadteil in der angegebenen URL durch. Für tiefe Hierarchien bedeutet dies viele Befehle. In der RFC1738 steht, dass es so gemacht werden soll. Dies ist der Standard, aber die langsamste Methode.

`CURLFTPMETHOD_NOCWD`
: libcurl führt überhaupt keinen CWD durch. libcurl führt SIZE, RETR, STOR usw. durch und gibt für all diese Befehle einen vollständigen Pfad zum Server an. Dies ist die schnellste Methode.

`CURLFTPMETHOD_SINGLECWD`
: libcurl führt einen CWD mit dem gesamten Zielverzeichnis durch und arbeitet dann mit der Datei normal ("normally", wie im Fall multicwd). Dies ist etwas normkonformer als 'nocwd', jedoch ohne die volle Konsequenz von 'multicwd'.

EINGABEN

`method` Eingabewert

5.106 easy:SetOpt_FTP_Response_Timeout

BEZEICHNUNG

easy:SetOpt_FTP_Response_Timeout – setzt die Zeit, die auf die FTP-Antwort gewartet wird

ÜBERSICHT

```
easy:SetOpt_FTP_Response_Timeout(timeout)
```

BESCHREIBUNG

Übergibt einen Wert. Veranlasst libcurl, eine `timeout`-Periode (in Sekunden) für die Zeitspanne festzulegen, die der Server einnehmen darf, um eine Antwortnachricht für einen Befehl zu senden, bevor die Sitzung als tot angesehen wird. Während libcurl auf eine Antwort wartet, überschreibt dieser Wert `#CURLOPT_TIMEOUT`. Es wird empfohlen, dass Sie bei Verwendung in Verbindung mit `#CURLOPT_TIMEOUT`, `#CURLOPT_FTP_RESPONSE_TIMEOUT` auf einen Wert kleiner als `#CURLOPT_TIMEOUT` setzen.

EINGABEN

`timeout` Eingabewert

5.107 easy:SetOpt_FTP_Skip_PASV_IP

BEZEICHNUNG

easy:SetOpt_FTP_Skip_PASV_IP – ignoriert die IP-Adresse in der PASV-Antwort

ÜBERSICHT

```
easy:SetOpt_FTP_Skip_PASV_IP(skip)
```

BESCHREIBUNG

Übergibt einen Wert. Wenn `skip` auf 1 gesetzt ist, weist er libcurl an, die IP-Adresse, die der Server in seiner 227-Antwort auf den PASV-Befehl von libcurl vorschlägt, nicht zu verwenden, wenn libcurl die Datenverbindung verbindet. Stattdessen wird libcurl die gleiche IP-Adresse wiederverwenden, die sie bereits für die Kontrollverbindung verwendet. Aber es wird die Portnummer aus der 227-Antwort verwenden.

Diese Option ermöglicht es libcurl somit, fehlerhafte Serverinstallationen zu umgehen, die aufgrund von NATs, Firewalls oder Inkompetenz die falsche IP-Adresse zurückmelden.

Diese Option hat keine Auswirkung, wenn PORT, EPRT oder EPSV anstelle von PASV verwendet wird.

EINGABEN

`skip` Eingabewert

5.108 easy:SetOpt_FTP_SSL_CCC

BEZEICHNUNG

easy:SetOpt_FTP_SSL_CCC – schaltet SSL mit FTP nach der Authentifizierung wieder aus

ÜBERSICHT

```
easy:SetOpt_FTP_SSL_CCC(how)
```

BESCHREIBUNG

Wenn diese Option aktiviert ist, verwendet libcurl CCC (Clear Command Channel). Nach der Authentifizierung wird die SSL/TLS-Ebene heruntergefahren. Der Rest der Steuerkanalkommunikation wird unverschlüsselt übertragen. Dadurch können NAT-Router der FTP-Transaktion folgen. Übergeben Sie einen Wert mit einem der folgenden Werte.

`CURLFTPSSL_CCC_NONE`
: Versucht nicht, CCC zu verwenden.

`CURLFTPSSL_CCC_PASSIVE`
: Leitet den Shutdown nicht ein, sondern wartet, bis der Server dies getan hat. Senden Sie keine Antwort.

`CURLFTPSSL_CCC_ACTIVE`
: Startet die Abschaltung und wartet auf eine Antwort.

EINGABEN

`how` Eingabewert

5.109 easy:SetOpt_FTP_Use_Eprt

BEZEICHNUNG

easy:SetOpt_FTP_Use_Eprt – aktiviert/deaktiviert die Nutzung von EPRT mit FTP

ÜBERSICHT

```
easy:SetOpt_FTP_Use_Eprt(enabled)
```

BESCHREIBUNG

Übergibt einen Wert. Wenn der Wert 1 ist, wird curl angewiesen, den Befehl EPRT zu verwenden, wenn aktive FTP-Downloads durchgeführt werden (was durch `#CURLOPT_FTPPORT` aktiviert wird). Die Verwendung von EPRT bedeutet, dass es zuerst versucht,

EPRT zu verwenden, bevor es PORT verwendet, aber wenn Sie dieser Option Null übergeben, wird es nicht versuchen, EPRT zu verwenden, sondern nur normalen PORT.

Wenn der Server ein IPv6-Host ist, hat diese Option keine Auswirkung, da dann EPRT erforderlich ist.

EINGABEN

`enabled` Eingabewert

5.110 easy:SetOpt_FTP_Use_Epsv

BEZEICHNUNG

easy:SetOpt_FTP_Use_Epsv – aktiviert/deaktiviert die Nutzung von EPSV mit FTP

ÜBERSICHT

```
easy:SetOpt_FTP_Use_Epsv(epsv)
```

BESCHREIBUNG

Übergibt `epsv` als Wert. Wenn der Wert 1 ist, weist er curl an, den EPSV-Befehl zu verwenden, wenn passive FTP-Downloads durchgeführt werden (was er standardmäßig tut). Die Verwendung von EPSV bedeutet, dass zuerst versucht wird, EPSV zu verwenden, bevor es PASV verwendet, aber wenn Sie dieser Option Null übergeben, wird es nicht versuchen, EPSV zu verwenden, sondern nur reine PASV.

Wenn der Server ein IPv6-Host ist, hat diese Option ab 7.12.3 keine Auswirkung mehr.

EINGABEN

`epsv` Eingabewert

5.111 easy:SetOpt_FTP_Use_Pret

BEZEICHNUNG

easy:SetOpt_FTP_Use_Pret – aktiviert den PRET-Befehl mit FTP

ÜBERSICHT

```
easy:SetOpt_FTP_Use_Pret(enable)
```

BESCHREIBUNG

Übergibt einen Wert. Wenn der Wert 1 ist, weist er curl an, einen PRET-Befehl vor PASV (und EPSV) zu senden. Bestimmte FTP-Server, hauptsächlich drftpd, benötigen diesen nicht standardmäßigen Befehl für Verzeichnisauflistungen sowie Up- und Downloads im PASV-Modus. Hat keine Wirkung bei Verwendung des aktiven FTP-Übertragungsmodus.

EINGABEN

`enable` Eingabewert

5.112 easy:SetOpt_FTPPort

BEZEICHNUNG

easy:SetOpt_FTPPort – aktiviert die FTP-Übertragung

ÜBERSICHT

```
easy:SetOpt_FTPPort(spec)
```

BESCHREIBUNG

Übergibt eine Zeichenkette als Parameter. Er legt fest, dass die FTP-Übertragung aktiv durchgeführt wird und die angegebene Zeichenkette verwendet wird, um die IP-Adresse für die FTP-PORT-Anweisung zu erhalten.

Die Anweisung PORT weist den Remote-Server an, sich mit der angegebenen IP-Adresse zu verbinden. Die Zeichenkette kann eine einfache IP-Adresse, ein Hostname, ein Name der Netzwerkschnittstelle (unter Unix) oder einfach nur ein '-' Symbol sein, damit die Bibliothek die Standard-IP-Adresse Ihres Systems verwenden kann. Standardmäßige FTP-Operationen sind passiv und verwenden daher keinen PORT.

Auf die Adresse kann ein ':' zur Angabe eines Ports folgen, optional gefolgt von einem '-' zur Angabe eines Portbereichs. Wenn der angegebene Port 0 ist, wählt das Betriebssystem einen freien Port aus. Wenn ein Bereich angegeben wird und nicht alle Ports im Bereich verfügbar sind, wird libcurl `#CURLE_FTP_PORT_FAILED` für das Zugriffsrecht melden. Ungültige Port-/Bereichseinstellungen werden ignoriert. IPv6-Adressen, gefolgt von einem Port oder Portbereich, müssen in Klammern stehen. IPv6-Adressen ohne Port-/Bereichsangabe können in Klammern angegeben werden.

Beispiele mit angegebenen Ports:

```
eth0:0
192.168.1.2:32000-33000
curl.se:32123
[::1]:1234-4567
```

Wenn Sie diese Option auf `Nil` setzen, wird der Port deaktiviert und kehrt zur Verwendung der passiven Version zurück.

EINGABEN

`spec` Eingabewert

5.113 easy:SetOpt_FTPSSLAuth

BEZEICHNUNG

easy:SetOpt_FTPSSLAuth – legt die Reihenfolge fest, in der TLS vs. SSL bei der Verwendung von FTP versucht werden soll

ÜBERSICHT

```
easy:SetOpt_FTPSSLAuth(order)
```

BESCHREIBUNG

Übergibt einen Wert mit einem der folgenden Werte, um zu ändern, wie libcurl `"AUTH TLS"` oder `"AUTH SSL"` ausgibt, wenn FTP über SSL aktiviert ist. Dies ist nur interessant, wenn auch `#CURLOPT_USE_USE_SSL` gesetzt ist.

Mögliche `order` Werte:

`#CURLFTPAUTH_DEFAULT`
: Lässt libcurl entscheiden.

`#CURLFTPAUTH_SSL`
: Versucht zuerst "AUTH SSL", und nur wenn das fehlschlägt, wird "AUTH TLS" versucht.

`#CURLFTPAUTH_TLS`
: Versucht zuerst "AUTH TLS", und nur wenn das fehlschlägt, wird "AUTH SSL" versucht.

EINGABEN

`order` Eingabewert

5.114 easy:SetOpt_GSSAPI_Delegation

BEZEICHNUNG

easy:SetOpt_GSSAPI_Delegation – legt die erlaubte Zuordnung von GSS-APIs fest

ÜBERSICHT

```
easy:SetOpt_GSSAPI_Delegation(level)
```

BESCHREIBUNG

Setzt den numerischen Parameter `level` auf `#CURLGSSAPI_DELEGATION_FLAG`, um die uneingeschränkte Übertragung von GSSAPI-Anmeldeinformationen zu ermöglichen. Die Übertragung ist seit 7.21.7 standardmäßig deaktiviert. Setzen Sie den Parameter auf `#CURLGSSAPI_DELEGATION_POLICY_FLAG`, um nur das OK-AS-DELEGATE-Flag zu übertragen, wenn dies im Service-Ticket gesetzt ist, falls dieses Merkmal von der GSS-API-Implementierung unterstützt wird und die Definition von GSS_C_DELEG_POLICY_FLAG zur Kompilierzeit verfügbar war.

EINGABEN

`level` Eingabewert

5.115 easy:SetOpt_Header

BEZEICHNUNG

easy:SetOpt_Header – übergibt den Header an den Datenstrom

ÜBERSICHT

```
easy:SetOpt_Header(onoff)
```

BESCHREIBUNG

Übergeben Sie dem Parameter `onoff` den Wert 1, um libcurl anzuweisen, den Header in den Schreib-Callback aufzunehmen (`#CURLOPT_WRITEFUNCTION`). Diese Option ist relevant für Protokolle, die tatsächlich Header oder andere Metadaten (wie HTTP und FTP) enthalten.

Wenn Sie darum ersuchen, den Header an den gleichen Callback wie des Bodys zu übergeben, ist es nicht möglich, sie wieder genau zu trennen, ohne detaillierte Kenntnisse über das verwendete Protokoll zu haben.

Ausserdem: Der #CURLOPT_WRITEFUNCTION-Callback ist darauf beschränkt, immer nur ein Maximum von #CURL_MAX_WRITE_SIZE-Bytes zu erhalten, die ihm übergeben werden (16KB), während ein Header länger sein kann und die `#CURLOPT_HEADERFUNCTION` unterstützt es, mit Header bis zu #CURL_MAX_HTTP_HEADER-Bytes groß (100KB) aufgerufen zu werden.

Es ist oft besser, `#CURLOPT_HEADERFUNCTION` zu verwenden, um die Header-Daten separat zu erhalten.

Obwohl verwirrend ähnlich benannt, wird `#CURLOPT_HTTPHEADER` verwendet, um eigene HTTP-Header zu setzen!

EINGABEN

`onoff` Eingabewert

5.116 easy:SetOpt_HeaderFunction

BEZEICHNUNG

easy:SetOpt_HeaderFunction – setzt die Callback-Funktion, die Header-Daten empfängt

ÜBERSICHT

```
easy:SetOpt_HeaderFunction(header_callback[, userdata])
```

BESCHREIBUNG

Übergibt eine Callback-Funktion. Diese Funktion wird von libcurl aufgerufen, sobald sie Header-Daten erhalten hat. Der Header-Callback wird einmal pro Header aufgerufen und nur komplette Header-Zeilen werden an den Callback übergeben. Das Parsen von Headers ist sehr einfach.

Der erste Parameter, der an Ihre Callback-Funktion übergeben wird, ist eine Zeichenkette, die die gerade empfangenen Header-Daten enthält. Wenn Sie das optionale Argument `userdata` übergeben, wird der Wert als zweiter Parameter an Ihre Callback-Funktion übergeben. Der Parameter `userdata` kann von beliebigem Typ sein.

Diese Callback-Funktion muss die Anzahl der tatsächlich behandelten Bytes zurückgeben. Wenn dieser Betrag von dem Betrag abweicht, der an Ihre Funktion übergeben wurde, wird ein Fehler an die Bibliothek gemeldet. Dies führt dazu, dass die Übertragung abgebrochen wird und die laufende libcurl-Funktion `#CURLE_WRITE_ERROR` zurückgibt.

Wenn Ihre Callback-Funktion nichts zurückgibt, signalisiert dies den Erfolg und die Übertragung wird fortgesetzt.

Ein kompletter HTTP-Header, die an diese Funktion übergeben wird, kann bis zu `#CURL_MAX_HTTP_HEADER` (100K) Bytes umfassen.

Es ist wichtig zu beachten, dass der Callback für den Header aller erhaltenen Antworten nach dem Einleiten einer Anfrage aufgerufen wird und nicht nur für die endgültige Antwort. Dazu gehören alle Antworten, die während der Authentifizierungsverhandlungen

auftreten. Wenn Sie nur die Header aus der endgültigen Antwort bearbeiten müssen, müssen Sie die Header im Callback selbst sammeln und HTTP-Statuszeilen verwenden, z.B. Antwortgrenzen zu definieren.

Wenn ein Server eine verteilte verschlüsselte Übertragung sendet, kann er einen Anhang enthalten. Dieser Anhang ist identisch mit einem HTTP-Header und wenn ein solcher empfangen wird, wird er ebenfalls über diesen Callback an die Anwendung übergeben. Es gibt mehrere Möglichkeiten, um zu erkennen, dass es sich um einen Anhang und nicht um eine gewöhnlichen Header handelt:

1) es kommt nach dem Antwort-Body.
2) es kommt nach der letzten Header-Zeile (CR LF).
3) einen Anhang: Der Header wird unter den regulären Antwort-Headern erwähnt, welche Header im Anhang zu erwarten sind.

Für Nicht-HTTP-Protokolle wie FTP, POP3, IMAP und SMTP wird diese Funktion mit den Serverantworten auf die Befehle, die libcurl sendet, aufgerufen.

EINGABEN

`header_callback`
: Eingabewert

`userdata`
: optional: Benutzerdaten, die an die Callback-Funktion übergeben werden sollen

5.117 easy:SetOpt_HeaderOpt

BEZEICHNUNG

easy:SetOpt_HeaderOpt – legt fest, wie HTTP-Header gesendet werden sollen

ÜBERSICHT

```
easy:SetOpt_HeaderOpt(bitmask)
```

BESCHREIBUNG

Übergibt einen Wert, der eine Bitmaske von Optionen für den Umgang mit Header ist. Die beiden sich gegenseitig ausschließenden Optionen sind:

`#CURLHEADER_UNIFIED`
: Die in `#CURLOPT_HTTPHEADER` angegebenen Header werden in Anfragen an Server und Proxies verwendet. Wenn diese Option aktiviert ist, hat `#CURLOPT_PROXYHEADER` keine Wirkung.

`#CURLHEADER_SEPARATE`
: Lässt `#CURLOPT_HTTPHEADER` Headers nur an einen Server und nicht an einen Proxy senden. Proxy-Header müssen mit `#CURLOPT_PROXYHEADER` gesetzt werden, um verwendet zu werden. Beachten Sie, dass libcurl sowohl Server-Header als auch Proxy-Header sendet, wenn eine Nicht-CONNECT-Anfrage an einen Proxy gesendet wird. Wenn Sie CONNECT ausführen, sendet libcurl `#CURLOPT_PROXYHEADER` Header nur an den Proxy und dann `#CURLOPT_HTTPHEADER` Header nur an den Server.

EINGABEN

`bitmask` Eingabewert

5.118 easy:SetOpt_HTTP200Aliases

BEZEICHNUNG

easy:SetOpt_HTTP200Aliases – setzt alternative Übereinstimmungen für HTTP 200 OK

ÜBERSICHT

```
easy:SetOpt_HTTP200Aliases(aliases)
```

BESCHREIBUNG

Übergibt eine Tabelle mit einer Liste von `aliases`, die als gültige HTTP 200-Antworten behandelt werden sollen. Einige Server antworten mit einer benutzerdefinierten Header-Antwortzeile. Beispielsweise antworten SHOUTcast-Server mit `"ICY 200 OK"`. Auch einige sehr alte Icecast 1.3.x-Server werden so für bestimmte User-Agent-Header oder in Abwesenheit solcher reagieren. Durch die Aufnahme dieser Zeichenkette in Ihre Liste der Aliase wird die Antwort als gültigen HTTP-Header wie beispielsweise `"HTTP/1.0 200 OK"` behandelt.

Der Alias selbst wird nicht für Versionszeichenketten analysiert. Es wird angenommen, dass das Protokoll mit HTTP 1.0 übereinstimmt, wenn ein Alias übereinstimmt.

EINGABEN

`aliases` Eingabewert

5.119 easy:SetOpt_HTTP_Content_Decoding

BEZEICHNUNG

easy:SetOpt_HTTP_Content_Decoding – aktiviert/deaktiviert die Dekodierung von HTTP-Inhalten

ÜBERSICHT

```
easy:SetOpt_HTTP_Content_Decoding(enabled)
```

BESCHREIBUNG

Übergibt einen Wert, um libcurl mitzuteilen, wie sie bei der Dekodierung von Inhalten vorgehen soll. Wenn auf Null gesetzt, wird die Inhaltsdekodierung deaktiviert. Wenn auf 1 gesetzt, ist es aktiviert. Libcurl hat keine standardmäßige Inhaltsdekodierung, erfordert aber die Verwendung von `#CURLOPT_ACCEPT_ENCODING` dafür.

EINGABEN

`enabled` Eingabewert

5.120 easy:SetOpt_HTTP_Transfer_Decoding

BEZEICHNUNG

easy:SetOpt_HTTP_Transfer_Decoding – aktiviert/deaktiviert die Dekodierung der HTTP-Übertragung

ÜBERSICHT

```
easy:SetOpt_HTTP_Transfer_Decoding(enabled)
```

BESCHREIBUNG

Übergeben Sie einen Wert, um libcurl mitzuteilen, wie sie sich bei der Transfer-Dekodierung verhalten soll. Wenn auf Null gesetzt, wird die Transfer-Dekodierung deaktiviert, wenn er auf 1 gesetzt ist, ist sie aktiviert (Standard). libcurl führt standardmäßig die Chunked-Transfer-Dekodierung durch, es sei denn, diese Option ist auf Null gesetzt.

EINGABEN

`enabled` Eingabewert

5.121 easy:SetOpt_HTTP_Version

BEZEICHNUNG

easy:SetOpt_HTTP_Version – gibt die zu verwendende HTTP-Protokollversion an

ÜBERSICHT

```
easy:SetOpt_HTTP_Version(version)
```

BESCHREIBUNG

Übergibt `version` als Wert, der auf einen der nachfolgend beschriebenen Werte gesetzt wird. Sie fragen libcurl an, ob die angegebenen HTTP-Version verwendet wird. Dies ist nicht sinnvoll, es sei denn, Sie haben einen guten Grund. Sie müssen diese Option setzen, wenn Sie die HTTP/2-Unterstützung von libcurl nutzen wollen.

Beachten Sie, dass die HTTP-Version nur eine Anfrage ist. libcurl wird weiterhin die Wiederverwendung einer bestehenden Verbindung priorisieren, so dass sie dann eine Verbindung mit einer HTTP-Version wiederverwenden kann, nach der Sie nicht gefragt haben.

`#CURL_HTTP_VERSION_NONE`
: Es ist egal, welche Version die Bibliothek verwendet. libcurl verwendet alles, was sie für richtig hält.

`#CURL_HTTP_VERSION_1_0`
: Erzwingt eine HTTP 1.0-Anfrage.

`#CURL_HTTP_VERSION_1_1`
: Erzwingt eine HTTP 1.1-Anfrage.

`#CURL_HTTP_VERSION_2_0`
: Versucht eine HTTP 2-Anfrage zu stellen. libcurl wird auf HTTP 1.1 zurückgreifen, wenn HTTP 2 nicht mit dem Server vereinbart werden kann. (Hinzugefügt in 7.33.0) Der Alias `#CURL_HTTP_VERSION_2` wurde in 7.43.0 hinzugefügt, um den tatsächlichen Protokollnamen besser wiederzugeben.

`#CURL_HTTP_VERSION_2TLS`
: Versucht HTTP 2 nur über TLS (HTTPS). libcurl fällt auf HTTP 1.1 zurück, wenn HTTP 2 nicht mit dem HTTPS-Server vereinbart werden kann. Für Klartext-HTTP-Server verwendet libcurl 1.1. (Hinzugefügt in 7.47.0)

`#CURL_HTTP_VERSION_2_PRIOR_KNOWLEDGE`
: Gibt Nicht-TLS-HTTP-Anfragen über HTTP/2 ohne HTTP/1.1-Upgrade aus. Es erfordert Vorkenntnisse, dass der Server HTTP/2 sofort unterstützt. HTTPS-Anfragen werden HTTP/2 weiterhin wie gewohnt mit der übermittelten Protokollversion im TLS-Handshake durchführt. (Hinzugefügt in 7.49.0)

EINGABEN

`version` Eingabewert

5.122 easy:SetOpt_HTTPAuth

BEZEICHNUNG

easy:SetOpt_HTTPAuth – legt die HTTP-Server-Authentifizierungsmethoden für den Versuch fest

ÜBERSICHT

```
easy:SetOpt_HTTPAuth(bitmask)
```

BESCHREIBUNG

Übergibt einen Wert als Parameter, der auf eine Bitmaske gesetzt ist, um libcurl mitzuteilen, welche Authentifizierungsmethode(n) Sie verwenden möchten, um mit dem Remote-Server zu kommunizieren.

Die verfügbaren Bits sind unten aufgelistet. Wenn mehr als ein Bit gesetzt ist, fragt libcurl zuerst die Website ab, um zu sehen, welche Authentifizierungsmethoden sie unterstützt und wählt dann die beste aus, die Sie ihr erlauben zu verwenden. Bei einigen Methoden führt dies zu einem zusätzlichen Netzwerkrundflug. Setzen Sie den aktuellen Namen und das aktuelle Passwort mit der Option `#CURLOPT_USERPWD` oder mit den Optionen `#CURLOPT_USERNAME` und `#CURLOPT_PASSWORD`.

Zur Authentifizierung mit einem Proxy siehe `#CURLOPT_PROXYAUTH`.

`#CURLAUTH_BASIC`
: HTTP Basis-Authentifizierung. Dies ist die Standardwahl und die einzige Methode, die weit verbreitet ist und praktisch überall unterstützt wird. Dadurch werden Benutzername und Passwort im Klartext über das Netzwerk gesendet und von anderen leicht erfasst.

`#CURLAUTH_DIGEST`
: HTTP Digest-Authentifizierung. Die Digest-Authentifizierung ist in RFC2617 definiert und stellt einen sichereren Weg zur Authentifizierung über öffentliche Netzwerke dar als die herkömmliche altmodische Basismethode.

`#CURLAUTH_DIGEST_IE`
: HTTP Digest-Authentifizierung mit IE-Variante. Die Digest-Authentifizierung ist in RFC2617 definiert und stellt einen sichereren Weg zur Authentifizierung über öffentliche Netzwerke dar als die herkömmliche altmodische Basismethode. Die IE-Variante ist einfach, dass libcurl eine spezielle Eigenart ("quirk") verwendet, von der bekannt ist, dass sie vor Version 7 verwendet wurde und dass einige Server den Client zur Verwendung benötigen.

`#CURLAUTH_BEARER`
: HTTP Beare Token-Authentifizierung, die hauptsächlich im OAuth 2.0-Protokoll verwendet wird. Sie können den Träger-Token für die Verwendung mit `#CURLOPT_XOAUTH2_BEARER` festlegen.

`#CURLAUTH_NEGOTIATE`
: HTTP Negotiate (SPNEGO)-Authentifizierung. Die Negotiate-Authentifizierung ist in RFC 4559 definiert und ist der sicherste Weg, um die Authentifizierung über HTTP durchzuführen. Sie müssen libcurl mit einer geeigneten GSS-API-Bibliothek oder SSPI unter Windows erstellen, damit dies funktioniert.

`#CURLAUTH_NTLM`
: HTTP NTLM-Authentifizierung. Ein proprietäres Protokoll, das von Microsoft entwickelt und verwendet wird. Es verwendet ein Challenge-Response- und Hash-Konzept ähnlich wie Digest, um zu verhindern, dass das Passwort gelesen wird. Sie müssen libcurl entweder mit OpenSSL-, GnuTLS- oder NSS-Unterstützung erstellen, damit diese Option funktioniert, oder libcurl unter Windows mit SSPI-Unterstützung erstellen.

`#CURLAUTH_NTLM_WB`
: NTLM delegiert an Winbind-Helfer. Die Authentifizierung erfolgt durch eine separate binäre Anwendung, die bei Bedarf ausgeführt wird. Der Name der Anwendung wird zur Kompilierungszeit angegeben, ist aber typischerweise /usr/bin/ntlm_auth.

  Beachten Sie, dass libcurl sich bei Bedarf teilt, um die Winbind-Anwendung auszuführen und beendet sie. Wenn sie fertig ist wird waitpid() aufgerufen, um auf seinen Ausgang zu warten. Auf POSIX-Betriebssystemen führt das Beenden des Prozesses zum Auslösen eines SIGCHLD-Signals (unabhängig davon, ob `#CURLOPT_NOSIGNAL` gesetzt ist), das von der Anwendung intelligent gehandhabt werden muss. Insbesondere darf die Anwendung nicht bedingungslos wait() in ihrem SIGCHLD-Signalhandler aufrufen, um zu vermeiden, dass sie einer Race-On-Condition unterworfen wird. Dieses Verhalten kann sich in zukünftigen Versionen von libcurl ändern.

`#CURLAUTH_ANY`
: Dies ist ein Komfortmakro, das alle Bits setzt und so libcurl dazu bringt, jedes auszuwählen, das es für geeignet hält. libcurl wählt automatisch dasjenige aus, das es für am sichersten hält.

`#CURLAUTH_ANYSAFE`
: Dies ist ein Komfortmakro, das alle Bits außer Basic setzt und so libcurl dazu bringt, alle auszuwählen, die es für geeignet hält. libcurl wählt automatisch dasjenige aus, das es für am sichersten hält.

`#CURLAUTH_ONLY`
: Dies ist ein Meta-Symbol. Verwenden Sie diesen Wert mit OR zusammen mit einem einzelnen spezifischen Auth-Wert, um libcurl dazu zu zwingen, nach uneingeschränkter Auth zu suchen, und wenn nicht, ist nur dieser einzelne Auth-Algorithmus akzeptabel.

EINGABEN

`bitmask` Eingabewert

5.123 easy:SetOpt_HTTPGet

BEZEICHNUNG

easy:SetOpt_HTTPGet – fragt nach einer HTTP GET-Anfrage

ÜBERSICHT

```
easy:SetOpt_HTTPGet(useget)
```

BESCHREIBUNG

Übergibt einen Wert. Wenn `useget` 1 ist, erzwingt dies, dass die HTTP-Anfrage wieder zur Verwendung von GET zurückkehrt. Verwendbar, wenn ein POST, HEAD, PUT, etc. zuvor mit dem gleichen curl verwendet wurde.

Wenn `#CURLOPT_HTTPGET` auf 1 gesetzt wird, wird `#CURLOPT_NOBODY` und `#CURLOPT_UPLOAD` automatisch auf 0 gesetzt.

Das Setzen dieser Option auf Null hat keine Auswirkung. Anwendungen müssen explizit auswählen, welche HTTP-Request-Methode sie verwenden möchten, sie können eine Methode nicht abwählen. Um ein Bearbeitung auf die Standardmethode zurückzusetzen, betrachten Sie `easy:Reset()`.

EINGABEN

`useget` Eingabewert

5.124 easy:SetOpt_HTTPHeader

BEZEICHNUNG

easy:SetOpt_HTTPHeader – legt den benutzerdefinierten HTTP-Header fest

ÜBERSICHT

```
easy:SetOpt_HTTPHeader(headers)
```

BESCHREIBUNG

Übergibt eine Tabelle mit einer Liste von HTTP-Header, die an den Server und/oder Proxy in Ihrer HTTP-Anfrage übergeben werden sollen. Die gleiche Liste kann sowohl für Host- als auch für Proxy-Anfragen verwendet werden!

Wenn Sie einen Header hinzufügen, der anderweitig generiert und intern von libcurl verwendet wird, wird stattdessen Ihr hinzugefügter Header verwendet. Wenn Sie einen Header ohne Inhalt wie in 'Accept:' (keine Daten auf der rechten Seite des Doppelpunktes) hinzufügen, wird der intern verwendete Header deaktiviert. Mit dieser Option können Sie neue Header hinzufügen, interne Header ersetzen und interne Header entfernen. Um einen Header ohne Inhalt (nichts auf der rechten Seite des Doppelpunktes) hinzuzufügen, verwenden Sie die Form 'MyHeader;' (beachten Sie das abschließende Semikolon).

Der in der Liste enthaltene Header darf nicht CRLF-terminiert sein, da libcurl nach jeder Header-Position CRLF hinzufügt. Die Nichteinhaltung führt zu seltsamen Fehlern, da der Server höchstwahrscheinlich einen Teil der von Ihnen angegebenen Header ignoriert.

Die erste Zeile einer Anforderung (die die Methode enthält, in der Regel ein GET oder POST) ist kein Header und kann mit dieser Option nicht ersetzt werden. Nur die Zeilen hinter der Anfragezeile sind Headers. Das Hinzufügen dieser Methodenzeile in diese Liste von Headers bewirkt nur, dass Ihre Anfrage einen ungültigen Header sendet. Verwenden Sie `#CURLOPT_CUSTOMREQUEST`, um die Methode zu ändern.

Übergeben Sie dieser Option `Nil`, um auf keinen benutzerdefinierten Header zurückzusetzen.

Die am häufigsten ersetzten Header haben "shortcuts" in den Optionen `#CURLOPT_COOKIE`, `#CURLOPT_USERAGENT` und `#CURLOPT_REFERERER`. Wir empfehlen, diese zu verwenden.

Es gibt eine alternative Option, die Headers nur für Anfragen setzt oder ersetzt, die mit CONNECT an einen Proxy gesendet werden: `#CURLOPT_PROXYHEADER`. Verwenden Sie `#CURLOPT_HEADEROPT`, um das Verhalten zu steuern.

EINGABEN

`headers` Eingabewert

BEISPIEL

```
e:SetOpt_HTTPHeader({"Custom-Header1: Test", "Custom-Header2: Test"})
```

Der obige Code fügt der HTTP-Anfrage zwei benutzerdefinierte Header hinzu.

5.125 easy:SetOpt_HTTPPost

BEZEICHNUNG

easy:SetOpt_HTTPPost – gibt den mehrteiligen Formpost-Inhalt an

ÜBERSICHT

```
easy:SetOpt_HTTPPost(formpost)
```

BESCHREIBUNG

Weist libcurl an, dass ein Multipart/Formdata HTTP POST gemacht wird und Sie weisen im Argument `formpost` an, welche Daten an den Server weitergegeben werden sollen. Übergeben Sie ein HTTP-Post-Objekt als Parameter. Der einfachste Weg, ein solches Objekt zu erstellen, ist die Verwendung von `hurl.Form()` wie dokumentiert.

Die Verwendung von POST mit HTTP 1.1 impliziert die Verwendung von "Expect: 100-continue" Header. Sie können diese Header mit `#CURLOPT_HTTPHEADER` deaktivieren.

Wenn `#CURLOPT_HTTPPOST` eingestellt wird, wird `#CURLOPT_NOBODY` automatisch auf 0 gesetzt.

EINGABEN

`formpost` Eingabewert

5.126 easy:SetOpt_HTTPProxyTunnel

BEZEICHNUNG

easy:SetOpt_HTTPProxyTunnel – setzt den Tunnel durch einen HTTP-Proxy

ÜBERSICHT

```
easy:SetOpt_HTTPProxyTunnel(tunnel)
```

BESCHREIBUNG

Setzen Sie den Tunnelparameter auf 1, um libcurl alle Operationen über den HTTP-Proxy zu tunneln (eingestellt mit `#CURLOPT_PROXY`). Es gibt einen großen Unterschied zwischen der Verwendung eines Proxys und dem Tunneln durch ihn.

Tunneln bedeutet, dass eine HTTP CONNECT-Anfrage an den Proxy gesendet wird, die ihn auffordert, sich mit einem Remote-Host unter einer bestimmten Portnummer zu verbinden und dann wird der Datenverkehr einfach durch den Proxy geleitet. Proxies neigen dazu, bestimmte Portnummern auf die Whitelist zu setzen, an die sie CONNECT-Anfragen zulassen und oft sind nur Port 80 und 443 erlaubt.

Um Proxy CONNECT-Antwort-Header aus Benutzer-Callbacks zu unterdrücken, verwenden Sie `#CURLOPT_SUPPRESS_CONNECT_HEADERS`.

HTTP-Proxies können in der Regel nur mit HTTP kommunizieren (aus offensichtlichen Gründen), was dazu führt, dass libcurl Nicht-HTTP-Anfragen in HTTP konvertiert, wenn ein HTTP-Proxy ohne diese Tunnel-Option verwendet wird. Wenn Sie beispielsweise nach einer FTP-URL fragen und einen HTTP-Proxy angeben, lässt libcurl eine FTP-URL in einer HTTP-GET-Anfrage an den Proxy senden. Indem Sie stattdessen durch den Proxy tunneln, vermeiden Sie diese Konvertierung (die ohnehin selten über den Proxy funktioniert).

EINGABEN

`tunnel` Eingabewert

5.127 easy:SetOpt_Ignore_Content_Length

BEZEICHNUNG

easy:SetOpt_Ignore_Content_Length – ignoriert den Content-Length-Header

ÜBERSICHT

```
easy:SetOpt_Ignore_Content_Length(ignore)
```

BESCHREIBUNG

Wenn `ignore` auf 1 gesetzt ist, wird der Content-Length-Header in der HTTP-Antwort ignoriert und ignorieren somit, dass bei FTP-Übertragungen danach angefragt wird oder Sie sich darauf verlassen können.

Dies ist nützlich für HTTP mit Apache 1.x (und ähnlichen Servern), die eine falsche Inhaltslänge für Dateien über 2 Gigabyte melden. Wenn diese Option verwendet wird, kann curl den Fortschritt nicht genau melden und stoppt den Download einfach, wenn der Server die Verbindung beendet.

Es ist auch nützlich bei FTP, wenn z.B. die Datei während der Übertragung wächst, was sonst dazu führt, dass libcurl einen Fehler meldet.

Verwenden Sie diese Option nur, wenn es unbedingt erforderlich ist.

EINGABEN

`ignore` Eingabewert

5.128 easy:SetOpt_InFileSize

BEZEICHNUNG

easy:SetOpt_InFileSize – legt die Größe der zu sendenden Eingabedatei fest

ÜBERSICHT

```
easy:SetOpt_InFileSize(filesize)
```

BESCHREIBUNG

Wenn Sie eine Datei auf eine Remote-Seite hochladen, sollte `filesize` verwendet werden, um libcurl mitzuteilen, wie groß die Eingabedatei sein soll. Dieser Wert muss als Wert übergeben werden. Siehe auch `#CURLOPT_INFILESIZE_LARGE` für das Senden von Dateien, die größer als 2 GB sind.

Für das Hochladen mit SCP ist diese Option oder `#CURLOPT_INFILESIZE_LARGE` obligatorisch.

Um diesen Wert wieder zurückzusetzen, setzen Sie ihn auf -1.

Beim Senden von E-Mails über SMTP kann mit diesem Befehl der optionale Parameter SIZE für den Befehl MAIL FROM angegeben werden.

Diese Option schränkt nicht ein, wie viele Daten libcurl tatsächlich gesendet hat, da dies vollständig davon gesteuert wird, was der Callback zurückgibt, aber das Anzeigen eines Wertes und das Senden eines anderen Wertes kann zu Fehlern führen.

EINGABEN

`filesize` Eingabewert

5.129 easy:SetOpt_InFileSize_Large

BEZEICHNUNG

easy:SetOpt_InFileSize_Large – legt die Größe der zu sendenden Eingabedatei fest

ÜBERSICHT

```
easy:SetOpt_InFileSize_Large(filesize)
```

BESCHREIBUNG

Wenn Sie eine Datei auf eine Remote-Seite hochladen, sollte `filesize` verwendet werden, um libcurl mitzuteilen, wie groß die Eingabedatei sein soll. Dieser Wert muss als curl_off_t übergeben werden.

Für das Hochladen mit SCP ist diese Option oder `#CURLOPT_INFILESIZE` obligatorisch.

Um diesen Wert wieder zurückzusetzen, setzen Sie ihn auf -1.

Beim Senden von E-Mails über SMTP kann mit diesem Befehl der optionale Parameter SIZE für den Befehl MAIL FROM angegeben werden.

Diese Option schränkt nicht ein, wie viele Daten libcurl tatsächlich gesendet hat, da dies vollständig davon gesteuert wird, was der Callback zurückgibt, aber das Anzeigen eines Wertes und das Senden eines anderen Wertes kann zu Fehlern führen.

EINGABEN

`filesize` Eingabewert

5.130 easy:SetOpt_Interface

BEZEICHNUNG

easy:SetOpt_Interface – legt die Quellschnittstelle für ausgehenden Datenverkehr fest

ÜBERSICHT

```
easy:SetOpt_Interface(interface)
```

BESCHREIBUNG

Übergibt eine Zeichenkette im Parameter `interface`. Hiermit wird der Name festgelegt, der als ausgehende Netzwerkschnittstelle verwendet werden soll. Der Name kann ein Schnittstellen-Name, eine IP-Adresse oder ein Hostname sein.

Beginnt der Parameter mit "if!", so wird er nur als Schnittstellenname behandelt und es wird nie ein Versuch unternommen, ihn als IP-Adresse zu verwenden oder zu versuchen eine Namensauflösung darauf zu anzuwenden. Wenn der Parameter mit "host!" beginnt, dann wird er entweder als IP-Adresse oder als Hostname behandelt. Hostnamen werden synchron aufgelöst. Die Verwendung des if!-Formats wird bei der Verwendung der Multi-Schnittstelle benutzt, um zu vermeiden, dass der Code blockiert wird. Wenn "if!" angegeben wird, aber der Parameter passt nicht zu einer vorhandenen Schnittstelle, wird von der libcurl-Funktion ein `#CURLE_INTERFACE_FAILED` zurückgegeben, mit der die Übertragung durchgeführt wurde.

libcurl unterstützt für diese Option die Verwendung von Netzwerkkartennamen unter Windows nicht.

EINGABEN

`interface`
Eingabewert

5.131 easy:SetOpt_IPResolve

BEZEICHNUNG

easy:SetOpt_IPResolve – gibt an, welche IP-Protokollversion verwendet werden soll

ÜBERSICHT

```
easy:SetOpt_IPResolve(resolve)
```

BESCHREIBUNG

Ermöglicht es einer Anwendung auszuwählen, welche Art von IP-Adressen bei der Auflösung von Hostnamen verwendet werden sollen. Dies ist nur interessant bei der Verwendung von Hostnamen, die Adressen mit mehr als einer IP-Version auflösen. Die zulässigen Werte sind:

`#CURL_IPRESOLVE_WHATEVER`
: Standard; löst Adressen für alle IP-Versionen auf, die Ihr System erlaubt.

`#CURL_IPRESOLVE_V4`
: Auflösen in IPv4-Adressen.

`#CURL_IPRESOLVE_V6`
: Auflösen in IPv6-Adressen.

EINGABEN

`resolve` Eingabewert

5.132 easy:SetOpt_IssuerCert

BEZEICHNUNG

easy:SetOpt_IssuerCert – setzt den Dateiname des SSL-Zertifikats des Ausstellers

ÜBERSICHT

```
easy:SetOpt_IssuerCert(file)
```

BESCHREIBUNG

Übergibt im Parameter `file` eine Zeichenkette mit der Bezeichnung einer Datei, die ein CA-Zertifikat im PEM-Format enthält. Wenn die Option gesetzt ist, wird eine zusätzliche Prüfung gegen das Peer-Zertifikat durchgeführt, um sicherzustellen, dass der Aussteller tatsächlich derjenige ist, der dem von der Option bereitgestellten Zertifikat zugeordnet ist. Diese zusätzliche Prüfung ist nützlich bei mehrstufigen PKI, bei denen man durchsetzen muss, dass das Peer-Zertifikat aus einem bestimmten Zweig der Struktur stammt.

Diese Option ist nur sinnvoll, wenn sie in Kombination mit der Option `#CURLOPT_SSL_VERIFYPEER` verwendet wird. Andernfalls wird das Ergebnis der Prüfung nicht als Fehler betrachtet.

Mit der Option wird ein bestimmter Fehlercode (`#CURLE_SSL_ISSUER_ERROR`) definiert, der zurückgegeben wird, wenn der Aufbau der SSL/TLS-Sitzung aufgrund einer Unstimmigkeit mit dem Aussteller des Peer-Zertifikats fehlgeschlagen ist (`#CURLOPT_SSL_VERIFYPEER` muss ebenfalls gesetzt sein, damit die Prüfung fehlschlägt). (Hinzugefügt in 7.19.0)

EINGABEN

`file` Eingabewert

5.133 easy:SetOpt_Keep_Sending_On_Error

BEZEICHNUNG

easy:SetOpt_Keep_Sending_On_Error – sendet weiter mit einer frühen HTTP-Antwort >= 300

ÜBERSICHT

```
easy:SetOpt_Keep_Sending_On_Error(keep_sending)
```

BESCHREIBUNG

Ein numerischer Parameter, der auf 1 gesetzt ist, weist die Bibliothek an, den Anfragetext weiterhin zu senden, wenn der zurückgegebene HTTP-Code gleich oder größer als 300 ist. Die Standardaktion wäre, das Senden zu stoppen und den Datenstrom oder die Verbindung zu schließen.

Diese Option eignet sich für die manuelle NTLM-Authentifizierung, d.h. wenn eine Anwendung nicht `#CURLOPT_HTTPAUTH` verwendet, sondern `"Authorization: NTLM ..."` setzt: Header manuell mit `#CURLOPT_HTTPHEADER`.

Diese Option eignet sich für die manuelle NTLM-Authentifizierung, d.h. wenn eine Anwendung nicht `#CURLOPT_HTTPAUTH` verwendet, sondern stattdessen `"Authorization: NTLM ..."` Headers manuell mit `#CURLOPT_HTTPHEADER` setzt.

Die meisten Anwendungen benötigen diese Option nicht.

EINGABEN

`keep_sending`
: Eingabewert

5.134 easy:SetOpt_KeyPasswd

BEZEICHNUNG

easy:SetOpt_KeyPasswd – setzt die Passphrase auf privaten Schlüssel

ÜBERSICHT

```
easy:SetOpt_KeyPasswd(pwd)
```

BESCHREIBUNG

Übergibt eine Zeichenkette als Parameter. Es wird als Passphrase verwendet, um den privaten Schlüssel `#CURLOPT_SSLKEY` oder `#CURLOPT_SSH_PRIVATE_KEYFILE` zu verwenden. Sie haben nie ein Passphrase benötigt, um ein Zertifikat zu laden, aber Sie benötigen eins, um Ihren privaten Schlüssel zu laden.

EINGABEN

`pwd` Eingabewert

5.135 easy:SetOpt_KRBLevel

BEZEICHNUNG

easy:SetOpt_KRBLevel – legt die FTP-Kerberos-Sicherheitsstufe fest

ÜBERSICHT

```
easy:SetOpt_KRBLevel(level)
```

BESCHREIBUNG

Übergibt eine Zeichenkette als Parameter. Legt die Kerberos-Sicherheitsstufe für FTP fest; dies ermöglicht auch die Sensibilisierung von Kerberos. Dies ist eine Zeichenkette, die einer der folgenden entsprechen sollte: 'clear', 'safe', 'confidential' oder 'private'. Wenn die Zeichenkette gesetzt ist, aber nicht mit einer dieser Zeichenketten übereinstimmt, wird 'privat' verwendet. Setzen Sie die Zeichenkette auf `Nil`, um die Kerberos-Unterstützung für FTP zu deaktivieren.

EINGABEN

`level` Eingabewert

5.136 easy:SetOpt_LocalPort

BEZEICHNUNG

easy:SetOpt_LocalPort – legt die lokale Portnummer für Socket fest

ÜBERSICHT

```
easy:SetOpt_LocalPort(port)
```

BESCHREIBUNG

Übergibt einen Wert. Hiermit wird die lokale Portnummer der für die Verbindung verwendeten Socket eingestellt. Dies kann in Kombination mit `#CURLOPT_INTERFACE` verwendet werden und es wird empfohlen, auch `#CURLOPT_LOCALPORTRANGE` zu benutzen, wenn diese Option aktiviert ist. Gültige Portnummern sind 1 - 65535.

EINGABEN

`port` Eingabewert

5.137 easy:SetOpt_LocalPortRange

BEZEICHNUNG

easy:SetOpt_LocalPortRange – legt die Anzahl zusätzlicher lokaler Ports zum Testen fest

ÜBERSICHT

```
easy:SetOpt_LocalPortRange(range)
```

BESCHREIBUNG

Übergibt einen Wert. Das Argument `range` ist die Anzahl der Versuche, die libcurl machen wird, um eine funktionierende lokale Portnummer zu finden. Es beginnt mit dem angegebenen `#CURLOPT_LOCALPORT` und fügt bei jedem erneuten Versuch eine Eins (1) zur Nummer hinzu. Wenn Sie diese Option auf 1 oder weniger setzen, wird libcurl nur einen Versuch unternehmen, die genaue Portnummer zu ermitteln. Portnummern sind von Natur aus knappe Ressourcen, die manchmal ausgelastet sind, so dass das Setzen dieses Wertes auf einen zu niedrigen Wert zu unnötigen Verbindungsaufbauausfällen führen kann.

EINGABEN

`range` Eingabewert

5.138 easy:SetOpt_Login_Options

BEZEICHNUNG

easy:SetOpt_Login_Options – legt die Login-Optionen fest

ÜBERSICHT

```
easy:SetOpt_Login_Options(options)
```

BESCHREIBUNG

Übergibt eine Zeichenkette als Parameter, die auf die Zeichenkette `options` zeigt, die für die Übertragung verwendet werden soll.

Weitere Informationen zu den Anmeldeoptionen finden Sie unter RFC2384, RFC5092 und IETF-Entwurf draft-earhart-url-smtp-00.txt.

`#CURLOPT_LOGIN_OPTIONS` kann verwendet werden, um protokollspezifische Anmeldeoptionen festzulegen, wie beispielsweise den bevorzugten Authentifizierungsmechanismus über `"AUTH=NTLM"` oder `"AUTH=*"` und sollte in Verbindung mit der Option `#CURLOPT_USERNAME` verwendet werden.

EINGABEN

`options` Eingabewert

5.139 easy:SetOpt_Low_Speed_Limit

BEZEICHNUNG

easy:SetOpt_Low_Speed_Limit – stellt eine niedrige Geschwindigkeitsbegrenzung in Bytes pro Sekunde ein

ÜBERSICHT

```
easy:SetOpt_Low_Speed_Limit(speedlimit)
```

BESCHREIBUNG

Übergeben Sie einen Wert als Parameter. Er enthält die durchschnittliche Übertragungsgeschwindigkeit in Bytes pro Sekunde, unter der die Übertragung während `#CURLOPT_LOW_SPEED_TIME` Sekunden liegen sollte, damit libcurl sie als zu langsam betrachtet und abbricht.

EINGABEN

`speedlimit`
Eingabewert

5.140 easy:SetOpt_Low_Speed_Time

BEZEICHNUNG

easy:SetOpt_Low_Speed_Time – stellt das Zeitlimit für niedrige Geschwindigkeit ein

ÜBERSICHT

```
easy:SetOpt_Low_Speed_Time(speedtime)
```

BESCHREIBUNG

Übergibt einen Wert als Parameter. Es enthält die Zeit in Anzahl Sekunden, die die Übertragungsgeschwindigkeit unter dem Wert `#CURLOPT_LOW_SPEED_LIMIT` liegen sollte, damit die Bibliothek sie für zu langsam hält und abbricht.

EINGABEN

`speedtime`
: Eingabewert

5.141 easy:SetOpt_Mail_Auth

BEZEICHNUNG

easy:SetOpt_Mail_Auth – legt die SMTP-Authentifizierungsadresse fest

ÜBERSICHT

```
easy:SetOpt_Mail_Auth(auth)
```

BESCHREIBUNG

Übergibt eine Zeichenkette als Parameter. Dies wird verwendet, um die Authentifizierungsadresse (Identität) einer übertragenen Nachricht anzugeben, die an einen anderen Server weitergeleitet wird.

Dieser optionale Parameter ermöglicht es kooperierenden Agenten in einer vertrauenswürdigen Umgebung, die Authentifizierung einzelner Nachrichten zu kommunizieren. Es sollte nur vom Anwendungsprogramm mit libcurl verwendet werden, wenn die Anwendung selbst ein Mailserver ist, der in einer solchen Umgebung agiert. Wenn die Anwendung als solche arbeitet und die AUTH-Adresse nicht bekannt oder ungültig ist, sollte für diesen Parameter eine leere Zeichenkette verwendet werden.

Im Gegensatz zu `#CURLOPT_MAIL_FROM` und `#CURLOPT_MAIL_RCPT` sollte die Adresse nicht innerhalb eines Paares von spitzen Klammern angegeben werden (`<>`). Wenn jedoch eine leere Zeichenkette verwendet wird, dann wird ein Paar Klammern von libcurl gesendet, wie von RFC2554 gefordert.

EINGABEN

`auth` Eingabewert

5.142 easy:SetOpt_Mail_From

BEZEICHNUNG

easy:SetOpt_Mail_From – gibt die SMTP-Absenderadresse an

ÜBERSICHT

```
easy:SetOpt_Mail_From(from)
```

BESCHREIBUNG

Übergibt eine Zeichenkette als Parameter. Dies sollte verwendet werden, um die E-Mail-Adresse des Absenders beim Senden von SMTP-Mails mit libcurl anzugeben.

Eine Absender-E-Mail-Adresse sollte mit spitzen Klammern (<>) um sie herum angegeben werden, die bei Nichtangabe automatisch hinzugefügt werden.

Wenn dieser Parameter nicht angegeben ist, wird eine leere Adresse an den Mailserver gesendet, was dazu führen kann, dass die E-Mail abgelehnt wird.

EINGABEN

`from` Eingabewert

5.143 easy:SetOpt_Mail_RCPT

BEZEICHNUNG

easy:SetOpt_Mail_RCPT – gibt die Liste der SMTP-Mail-Empfänger an

ÜBERSICHT

```
easy:SetOpt_Mail_RCPT(rcpts)
```

BESCHREIBUNG

Übergibt eine Tabelle mit einer Liste von Empfängern, die an den Server in Ihrer SMTP-Mail-Anfrage übergeben werden sollen.

Wenn Sie eine E-Mail-Übertragung durchführen, sollte jeder Empfänger in eckigen Klammern (<>) angegeben werden. Wenn Sie jedoch keine eckigen Klammern verwenden, geht libcurl davon aus, dass Sie eine einzelne E-Mail-Adresse angegeben haben und schließt diese Adresse in eckige Klammern ein.

Bei der Durchführung einer Adressverifizierung (VRFY-Befehl) sollte jeder Empfänger als Benutzername oder Benutzername und Domäne angegeben werden (gemäß Abschnitt 3.5 von RFC5321).

Wenn Sie eine Erweiterung der Mailingliste durchführen (EXPN-Befehl), sollte jeder Empfänger über den Namen der Mailingliste angegeben werden, z.B. "Freunde" oder "London-Office".

EINGABEN

`rcpts` Eingabewert

5.144 easy:SetOpt_Max_Recv_Speed_Large

BEZEICHNUNG

easy:SetOpt_Max_Recv_Speed_Large – setzt die Geschwindigkeitslimit für das Herunterladen von Daten

ÜBERSICHT

```
easy:SetOpt_Max_Recv_Speed_Large(speed)
```

BESCHREIBUNG

Übergeben Sie einen Wert als Parameter. Wenn ein Download diese Geschwindigkeit überschreitet (gezählt in Bytes pro Sekunde), wird die Übertragung unterbrochen, um die Geschwindigkeit unter oder gleich dem Parameterwert zu halten. Standardmäßig ist die Geschwindigkeit unbegrenzt.

Diese Option wirkt sich nicht auf die mit FILE:// URLs durchgeführten Übertragungsgeschwindigkeiten aus.

EINGABEN

`speed` Eingabewert

5.145 easy:SetOpt_Max_Send_Speed_Large

BEZEICHNUNG

easy:SetOpt_Max_Send_Speed_Large – setzt die Geschwindigkeitslimit für das Hochladen von Daten

ÜBERSICHT

```
easy:SetOpt_Max_Send_Speed_Large(maxspeed)
```

BESCHREIBUNG

Übergibt mit `maxspeed` einen Wert als Parameter. Wenn das Hochladen diese Geschwindigkeit überschreitet (in Bytes pro Sekunde), wird die Übertragung unterbrochen, um die Geschwindigkeit auf dem Wert des Parameters oder darunter zu halten. Standardmäßig ist die Geschwindigkeit unbegrenzt.

Diese Option wirkt sich nicht auf die mit FILE:// URLs durchgeführten Übertragungsgeschwindigkeiten aus.

EINGABEN

`maxspeed` Eingabewert

5.146 easy:SetOpt_MaxConnects

BEZEICHNUNG

easy:SetOpt_MaxConnects – setzt die maximale Verbindungs-Cache-Größe

ÜBERSICHT

```
easy:SetOpt_MaxConnects(amount)
```

BESCHREIBUNG

Übergibt einen Wert. Der in `amount` angegebene Wert ist die maximale Anzahl von gleichzeitig geöffneten permanenten Verbindungen, die libcurl in dem Handle zugeordneten Pool zwischenspeichern darf. Die Voreinstellung ist 5 und es macht nicht viel Sinn, diesen Wert zu ändern, es sei denn, Sie wissen genau, wie das funktioniert und ändern das Verhalten von libcurl. Dies betrifft Verbindungen, die eines der Protokolle verwenden, die permanente Verbindungen unterstützen.

Bei Erreichen der maximalen Grenze schließt curl die älteste im Cache, um zu verhindern, dass die Anzahl der offenen Verbindungen erhöht wird.

Wenn Sie bereits Übertragungen mit diesem Curl-Handle durchgeführt haben, kann das Setzen eines kleineren `#CURLOPT_MAXCONNECTS` als bisher dazu führen, dass offene Verbindungen unnötig geschlossen werden.

Wenn Sie diesen Easy-Handle zu einem Multi-Handle hinzufügen, wird diese Einstellung nicht berücksichtigt, sondern Sie müssen `multi:SetOpt()` und die Option `#CURLMOPT_MAXCONNECTS` verwenden.

EINGABEN

`amount` Eingabewert

5.147 easy:SetOpt_MaxFileSize

BEZEICHNUNG

easy:SetOpt_MaxFileSize – setzt die maximal zulässige Dateigröße für das Herunterladen

ÜBERSICHT

```
easy:SetOpt_MaxFileSize(size)
```

BESCHREIBUNG

Übergibt einen Wert als Parameter. Auf diese Weise können Sie im Parameter `size` die maximale Größe (in Bytes) einer herunterzuladenden Datei angeben. Wenn die angeforderte Datei größer als dieser Wert ist, wird die Übertragung nicht gestartet und `#CURLE_FILESIZE_EXCEEDED` wird zurückgegeben.

Die Dateigröße ist vor dem Herunterladen nicht immer bekannt und für solche Dateien hat diese Option keine Auswirkung, auch wenn die Dateiübertragung am Ende größer als die vorgegebene Grenze ist. Dies betrifft sowohl FTP- als auch HTTP-Übertragungen.

Wenn Sie ein Limit über 2 GB wünschen, verwenden Sie `#CURLOPT_MAXFILESIZE_LARGE`.

EINGABEN

`size` Eingabewert

5.148 easy:SetOpt_MaxFileSize_Large

BEZEICHNUNG

easy:SetOpt_MaxFileSize_Large – setzt die maximal zulässige Dateigröße für das Herunterladen

ÜBERSICHT

```
easy:SetOpt_MaxFileSize_Large(size)
```

BESCHREIBUNG

Übergibt einen Wert als Parameter. Auf diese Weise können Sie im Parameter `size` die maximale Größe (in Bytes) einer herunterzuladenden Datei angeben. Wenn die angeforderte Datei größer als dieser Wert ist, wird die Übertragung nicht gestartet und `#CURLE_FILESIZE_EXCEEDED` wird zurückgegeben.

Die Dateigröße ist vor dem Herunterladen nicht immer bekannt und für solche Dateien hat diese Option keine Auswirkung, auch wenn die Dateiübertragung am Ende größer als die vorgegebene Grenze ist. Dies betrifft sowohl FTP- als auch HTTP-Übertragungen.

EINGABEN

`size` Eingabewert

5.149 easy:SetOpt_MaxRedirs

BEZEICHNUNG

easy:SetOpt_MaxRedirs – setzt die maximale Anzahl von erlaubten Umleitungen

ÜBERSICHT

```
easy:SetOpt_MaxRedirs(amount)
```

BESCHREIBUNG

Übergibt einen Wert. Die eingestellte Nummer im Parameter `amount` ist die Umleitungsgrenze. Wenn zu viele Umleitungen durchgeführt wurden, führt die nächste Umleitung zu einem Fehler (`#CURLE_TOO_MANY_REDIRECTS`). Diese Option ist nur sinnvoll, wenn gleichzeitig die `#CURLOPT_FOLLOWLOCATION` verwendet wird.

Wenn Sie das Limit auf 0 setzen, wird libcurl jede Umleitung ablehnen.

Stellen Sie ihn für eine unbegrenzte Anzahl von Umleitungen auf -1.

EINGABEN

`amount` Eingabewert

5.150 easy:SetOpt_Netrc

BEZEICHNUNG

easy:SetOpt_Netrc – fordert an, dass .netrc verwendet wird

ÜBERSICHT

```
easy:SetOpt_Netrc(level)
```

BESCHREIBUNG

Dieser Parameter steuert die Einstellung `level` von libcurl zwischen der Verwendung von Benutzernamen und Passwörtern aus Ihrer `~/.netrc`-Datei, bezogen auf Benutzernamen und Passwörter in der mit `#CURLOPT_URL` ausgelieferten URL. Unter Windows verwendet libcurl die Datei als `%HOME%/_netrc`, aber Sie können libcurl auch einen anderen Dateinamen für `#CURLOPT_NETRC_FILE` mitteilen.

libcurl verwendet einen Benutzernamen (und ein mitgeliefertes oder abgefragtes Passwort), der mit `#CURLOPT_USERPWD` oder `#CURLOPT_USERNAME` geliefert wird, anstelle einer der durch diesen Parameter gesteuerten Option.

Es werden nur Rechnername, Benutzername und Passwort berücksichtigt (Init-Makros und ähnliche Dinge werden nicht unterstützt).

libcurl überprüft nicht, ob die Datei die richtigen Eigenschaften hat (wie der Standard Unix ftp Client). Es sollte nur für den Benutzer lesbar sein.

`level` sollte auf einen der nachfolgend beschriebenen Werte eingestellt werden.

`#CURL_NETRC_OPTIONAL`
: Die Verwendung der Datei `~/.netrc` ist optional, und Informationen in der URL sind zu bevorzugen. Die Datei wird nach dem Host- und dem Benutzernamen (nur um das Passwort zu finden) oder nur nach dem Host durchsucht, um den ersten Benutzernamen und das Passwort von diesem `Rechner` zu finden, je nachdem, welche Informationen nicht angegeben sind. Undefinierte Werte der Option haben diesen Effekt.

`#CURL_NETRC_IGNORED`
: Die Bibliothek ignoriert die Datei `~/.netrc`. Dies ist die Voreinstellung.

`#CURL_NETRC_REQUIRED`
: Die Verwendung der Datei `~/.netrc` ist erforderlich und Informationen in der URL sind zu ignorieren. Die Datei wird nach dem Host- und dem Benutzernamen (nur um das Passwort zu finden) oder nur nach dem Host durchsucht, um den ersten Benutzernamen und das Passwort von diesem `Rechner` zu finden, je nachdem, welche Informationen nicht angegeben sind.

EINGABEN

`level` Eingabewert

5.151 easy:SetOpt_Netrc_File

BEZEICHNUNG

easy:SetOpt_Netrc_File – setzt den Dateiname zum Lesen von .netrc-Informationen

ÜBERSICHT

```
easy:SetOpt_Netrc_File(file)
```

BESCHREIBUNG

Übergibt eine Zeichenkette im Parameter `file`, die den vollständigen Pfadnamen der Datei enthält, die libcurl als `.netrc`-Datei verwenden soll. Wenn diese Option weggelassen wird und `#CURLOPT_NETRC` gesetzt ist, versucht libcurl, eine .netrc-Datei im Hauptverzeichnis des aktuellen Benutzers zu finden.

EINGABEN

`file` Eingabewert

5.152 easy:SetOpt_New_Directory_Perms

BEZEICHNUNG

easy:SetOpt_New_Directory_Perms – setzt die Berechtigungen für neu erstellte Remote-Verzeichnisse

ÜBERSICHT

```
easy:SetOpt_New_Directory_Perms(mode)
```

BESCHREIBUNG

Übergibt einen Wert als Parameter, der den Wert der Berechtigungen enthält, die neu erstellten Verzeichnissen auf dem Remote-Server zugewiesen werden. Der Standardwert

ist `0755`, aber es kann jeder gültige Wert verwendet werden. Die einzigen Protokolle, die dies verwenden können, sind `sftp://`, `scp://` und `file://`.

EINGABEN

`mode` Eingabewert

5.153 easy:SetOpt_New_File_Perms

BEZEICHNUNG

easy:SetOpt_New_File_Perms – setzt die Berechtigungen für neu erstellte Remote-Dateien

ÜBERSICHT

```
easy:SetOpt_New_File_Perms(mode)
```

BESCHREIBUNG

Übergibt einen Wert als Parameter, der den Wert der Berechtigungen enthält, die neu erstellten Dateien auf dem Remote-Server zugewiesen werden. Der Standardwert ist `0644`, aber es kann jeder gültige Wert verwendet werden. Die einzigen Protokolle, die dies verwenden können, sind `sftp://`, `scp://` und `file://`.

EINGABEN

`mode` Eingabewert

5.154 easy:SetOpt_Nobody

BEZEICHNUNG

easy:SetOpt_Nobody – führt die Download-Anfrage durch, ohne den Body zu erhalten

ÜBERSICHT

```
easy:SetOpt_Nobody(opt)
```

BESCHREIBUNG

Ein numerischer Parameter, der auf 1 gesetzt ist, weist libcurl an, den Body nicht in die Ausgabe aufzunehmen, wenn er etwas tut, was sonst ein Download wäre. Für HTTP(S) führt dies dazu, dass libcurl eine HEAD-Anfrage ausführt. Für die meisten anderen Protokolle bedeutet dies, dass einfach nicht aufgefordert wird, die Body-Daten zu übertragen.

Das Aktivieren dieser Option bedeutet, dass Sie einen Download ohne Body anfragen.

EINGABEN

`opt` Eingabewert

5.155 easy:SetOpt_NoProgress

BEZEICHNUNG

easy:SetOpt_NoProgress – schaltet die Fortschrittsanzeige aus

ÜBERSICHT

```
easy:SetOpt_NoProgress(onoff)
```

BESCHREIBUNG

Wenn `onoff` auf 1 gesetzt ist, sagt es der Bibliothek, dass sie die Fortschrittsanzeige für Anfragen mit diesem `Handle` vollständig abschalten soll. Es verhindert auch, dass `#CURLOPT_PROGRESSFUNCTION` aufgerufen wird.

EINGABEN

`onoff` Eingabewert

5.156 easy:SetOpt_NoProxy

BEZEICHNUNG

easy:SetOpt_NoProxy – deaktiviert die Proxy-Nutzung für bestimmte Hosts

ÜBERSICHT

```
easy:SetOpt_NoProxy(noproxy)
```

BESCHREIBUNG

Übergibt eine Zeichenkette. Die Zeichenkette besteht aus einer kommagetrennten Liste von Hostnamen, die nicht erfordern, dass ein Proxy erreicht wird, selbst wenn einer angegeben ist. Der einzige verfügbare Platzhalter ist ein einzelnes *-Zeichen, der auf alle Hosts zutrifft und den Proxy effektiv deaktiviert. Jeder Name in dieser Liste wird entweder als Domäne, die den Hostnamen enthält, oder als Domäne Hostname selbst abgeglichen. Zum Beispiel würde example.com mit example.com,example.com:80 übereinstimmen und www.example.com, aber nicht mit www.notanexample.com oder example.com.othertld.

Wenn der Name in der `noproxy`-Liste einen führenden Punkt hat, ist es ein Domainabgleich mit dem angegebenen Hostnamen. Auf diese Weise schaltet `".example.com"` die Proxy-Nutzung sowohl für `"www.example.com"` als auch für `"foo.example.com"` aus.

Das Setzen der noproxy-Zeichenkette auf `""` (eine leere Zeichenkette) aktiviert den Proxy explizit für alle Hostnamen, auch wenn eine Umgebungsvariable dafür gesetzt ist.

Geben Sie numerische IPv6-Adressen in die Liste der Hostnamen ein, ohne sie in Klammern zu setzen:

```
"example.com,::1,localhost"
```

EINGABEN

`noproxy` Eingabewert

5.157 easy:SetOpt_NoSignal

BEZEICHNUNG

easy:SetOpt_NoSignal – überspringt die gesamte Signalverarbeitung

ÜBERSICHT

```
easy:SetOpt_NoSignal(onoff)
```

BESCHREIBUNG

Wenn `onoff` 1 ist, verwendet libcurl keine Funktionen, die Signal-Handler installieren oder Funktionen, die dazu führen, dass Signale an den Prozess gesendet werden. Mit dieser Option können Multithread-Unix-Anwendungen weiterhin alle Timeout-Optionen usw. festlegen/verwenden, ohne das Risiko einzugehen, Signale zu erhalten.

Wenn diese Option aktiviert ist und libcurl mit dem Standardnamensauflöser erstellt wurde, treten während der Namensauflösung keine Zeitüberschreitungen auf. Erwägen Sie, libcurl mit den c-ares oder Threaded-Resolver-Backends zu erstellen, um asynchrone DNS-Lookups zu ermöglichen und um Zeitüberschreitungen für Namensauflösungen ohne die Verwendung von Signalen zu ermöglichen.

Wenn Sie `#CURLOPT_NOSIGNAL` auf 1 setzen, fordert libcurl das System NICHT auf, SIGPIPE-Signale zu ignorieren, die andernfalls vom System gesendet werden, wenn versucht wird, Daten an einen Socket zu senden, der am anderen Ende geschlossen ist. libcurl bemüht sich, solche SIGPIPEs niemals auszulösen, aber einige Betriebssysteme können sie nicht vermeiden, und selbst bei solchen gibt es einige Fälle, in denen sie entgegen unserem Wunsch möglicherweise noch auftreten. Darüber hinaus kann die Verwendung der `CURLAUTH_NTLM_WB-Authentifizierung` dazu führen, dass ein SIGCHLD-Signal ausgelöst wird.

EINGABEN

`onoff` Eingabewert

5.158 easy:SetOpt_Password

BEZEICHNUNG

easy:SetOpt_Password – setzt das Passwort zur Verwendung bei der Authentifizierung

ÜBERSICHT

```
easy:SetOpt_Password(pwd)
```

BESCHREIBUNG

Übergibt eine Zeichenkette als Parameter, die auf das Passwort für die Übertragung zeigt.

Die Option `#CURLOPT_PASSWORD` sollte in Verbindung mit der Option `#CURLOPT_USERNAME` verwendet werden.

EINGABEN

`pwd` Eingabewert

5.159 easy:SetOpt_Path_As_Is

BEZEICHNUNG

easy:SetOpt_Path_As_Is – verwendet keine Punkt-Punkt-Sequenzen

ÜBERSICHT

```
easy:SetOpt_Path_As_Is(leaveit)
```

BESCHREIBUNG

Setzen Sie den Parameter `leaveit` auf 1, um libcurl explizit anzuweisen, den angegebenen Pfad nicht zu ändern, bevor er an den Server weitergeleitet wird.

Dies weist libcurl an, NICHT die Sequenzen von "/../" oder "/./" zu verwerfen, die im Pfadteil der URL vorhanden sein können und die gemäß RFC 3986 Abschnitt 5.2.4 entfernt werden sollen.

Es ist bekannt, dass einige Serverimplementierungen (fälschlicherweise) erfordern, dass die Punkt-Punkt-Sequenzen im Pfad verbleiben und einige Clients möchten diese Umleitungen, um Serverimplementierungen auszuprobieren.

Standardmäßig führt libcurl solche Sequenzen zusammen, bevor der Pfad verwendet wird.

EINGABEN

`leaveit` Eingabewert

5.160 easy:SetOpt_PinnedPublicKey

BEZEICHNUNG

easy:SetOpt_PinnedPublicKey – legt das Public Key Pinning fest

ÜBERSICHT

```
easy:SetOpt_PinnedPublicKey(pinnedpubkey)
```

BESCHREIBUNG

Übergibt eine Zeichenkette als Parameter. Die Zeichenkette kann der Dateiname Ihres Public Key Pinning sein. Als Dateiformat wird "PEM" oder "DER" erwartet. Die Zeichenkette kann auch eine beliebige Anzahl von base64-kodierten sha256-Hashes sein, denen "sha256//" vorangestellt und durch ";" getrennt ist.

Beim vermitteln einer TLS- oder SSL-Verbindung sendet der Server ein Zertifikat, das seine Identität angibt. Ein öffentlicher Schlüssel wird aus diesem Zertifikat ausgelesen. Wenn er nicht genau mit dem für diese Option bereitgestellten öffentlichen Schlüssel übereinstimmt, bricht curl die Verbindung ab, bevor Daten gesendet oder empfangen werden.

Bei Nichtübereinstimmung wird `#CURLE_SSL_PINNEDPUBKEYNOTMATCH` zurückgegeben.

EINGABEN

`pinnedpubkey`
Eingabewert

5.161 easy:SetOpt_PipeWait

BEZEICHNUNG

easy:SetOpt_PipeWait – wartet auf Pipelining/Multiplexing

ÜBERSICHT

```
easy:SetOpt_PipeWait(wait)
```

BESCHREIBUNG

Mit setzen von `wait` auf 1 weisen Sie libcurl an, lieber auf eine Verbindung zu warten, um zu bestätigen oder zu verweigern, dass Pipelining oder Multiplexing möglich ist, bevor Sie fortfahren.

Wenn eine neue Übertragung ausgeführt werden soll, die Pipelining oder Multiplexing ermöglicht, prüft libcurl, ob vorhandene Verbindungen wiederverwendet werden können und leitet sie weiter. Wenn keine solche Verbindung besteht, wird diese sofort fortgesetzt und eine neue Verbindung erstellt, die verwendet werden kann.

Wenn Sie diese Option auf 1 setzen - und `CURLMOPT_PIPELINING` für das Multi-Handle, mit dem diese Übertragung verbunden ist, aktiviert haben - wartet libcurl stattdessen darauf, dass die Verbindung meldet, ob es möglich ist, weiter zu pipelinieren/multiplexen, bevor sie fortgesetzt wird. Dadurch kann libcurl die Anzahl der Verbindungen bei Verwendung von Pipeline- oder Multiplex-Protokollen viel besser auf ein Minimum beschränken.

Dies hat zur Folge, dass libcurl mit dieser Option lieber wartet und eine vorhandene Verbindung für das Pipelining wiederverwendet, als das Gegenteil: Lieber eine neue Verbindung öffnen als warten.

Die Wartezeit ist so lang, wie es dauert, bis die Verbindung hergestellt ist und bis libcurl die erforderliche Antwort zurückerhält, die sie über das Protokoll und die Unterstützungsebene informiert.

EINGABEN

`wait` Eingabewert

5.162 easy:SetOpt_Port

BEZEICHNUNG

easy:SetOpt_Port – stellt die Nummer des Remote-Ports ein, mit dem gearbeitet werden soll

ÜBERSICHT

```
easy:SetOpt_Port(number)
```

BESCHREIBUNG

Diese Option setzt `number` als die Nummer des Remote-Ports, mit dem eine Verbindung hergestellt werden soll, anstelle der in der URL angegebenen Nummer oder des Standardports für das verwendete Protokoll.

Normalerweise lässt man die URL einfach entscheiden, welchen Port man verwenden möchte, aber das erlaubt der Anwendung, diesen zu überschreiben.

Eine Portnummer ist in der Regel eine 16-Bit-Nummer und daher führt die Verwendung einer Portnummer über 65535 zu einem Laufzeitfehler.

EINGABEN

`number` Eingabewert

5.163 easy:SetOpt_Post

BEZEICHNUNG

easy:SetOpt_Post – fordert einen HTTP-POST an

ÜBERSICHT

```
easy:SetOpt_Post(post)
```

BESCHREIBUNG

Wird der Parameter `post` auf 1 gesetzt, weist libcurl an, einen regulären HTTP-Post zu schreiben. Dadurch verwendet die Bibliothek auch den Header "Content-Type: application/x-wwww-form- urlencoded". (Dies ist bei weitem die am häufigsten verwendete POST-Methode).

Verwenden Sie `#CURLOPT_POSTFIELDS`, um anzugeben, welche Daten gesendet werden sollen.

Optional können Sie POST-Daten mit den Optionen `#CURLOPT_READFUNCTION` und `#CURLOPT_READDATA` zur Verfügung stellen, aber dann müssen Sie sicherstellen, dass Sie `#CURLOPT_POSTFIELDS` nicht auf etwas anderes als Null setzen. Wenn Sie Daten mit einem Callback zur Verfügung stellen, müssen Sie sie mit gebündelter Übertragungscodierung übertragen oder die Größe der Daten mit den Optionen `#CURLOPT_POSTFIELDSIZE` oder `#CURLOPT_POSTFIELDSIZE_LARGE` einstellen. Um die Chunked-Codierung zu aktivieren, übergeben Sie einfach den entsprechenden Transfer-Encoding-Header (siehe Beispiel post-callback.c).

Sie können den standardmäßigen POST Content-Type: Header überschreiben, indem Sie Ihren eigenen mit `#CURLOPT_HTTPHEADER` einstellen.

Die Verwendung von POST mit HTTP 1.1 impliziert die Verwendung eines "Expect: 100-continue" Headers. Sie können diesen Header mit `#CURLOPT_HTTPHEADER` wie gewohnt deaktivieren.

Wenn Sie POST an einen HTTP 1.1-Server verwenden, können Sie Daten senden, ohne die Größe zu kennen, bevor Sie den POST starten, wenn Sie eine gebündelte Verschlüsselung verwenden. Sie aktivieren dies, indem Sie einen Header wie "Transfer-Encoding: chunked" mit `#CURLOPT_HTTPHEADER` hinzufügen. Bei HTTP 1.0 oder ohne gebündeltem Transfer müssen Sie die Größe in der Anfrage angeben.

Wenn `#CURLOPT_POST` auf 1 gesetzt wird, setzt libcurl automatisch `#CURLOPT_NOBODY` und `#CURLOPT_HTTPGET` auf 0.

Wenn Sie eine POST-Anfrage stellen und dann einen HEAD oder GET mit dem gleichen wiederverwendeten Handle erstellen möchten, müssen Sie den neuen Anforderungstyp explizit mit `#CURLOPT_NOBODY` oder `#CURLOPT_HTTPGET` oder ähnlich einstellen.

EINGABEN

`post`	Eingabewert

5.164 easy:SetOpt_PostFields

BEZEICHNUNG

easy:SetOpt_PostFields – setzt die Daten, welche an den Server gesendet werden

ÜBERSICHT

`easy:SetOpt_PostFields(postdata)`

BESCHREIBUNG

Übergeben Sie eine Zeichenkette als Parameter, die auf die vollständigen Daten zeigt, um eine HTTP-POST-Operation zu senden. Sie müssen sicherstellen, dass die Daten so formatiert sind, wie Sie sie vom Server erhalten möchten. libcurl wird sie in keiner Weise für Sie konvertieren oder kodieren. So kann beispielsweise der Webserver davon ausgehen, dass diese Daten url-codiert sind.

Dieser POST ist eine normale application/x-www-form-url-encodierte Art (und libcurl setzt diesen Content-Typ standardmäßig, wenn diese Option verwendet wird), die häufig von HTML-Formularen verwendet wird. Ändern Sie den Content-Typ mit `#CURLOPT_HTTPHEADER`.

Sie können `easy:Escape()` verwenden, um Ihre Daten bei Bedarf in die URL-Kodierung zu konvertieren. Es wird eine kodierte Zeichenkette zurückgegeben, die in `postdata` übergeben werden kann.

Die Verwendung von `#CURLOPT_POSTFIELDS` impliziert die Einstellung von `#CURLOPT_POST` auf 1.

Wenn `#CURLOPT_POSTFIELDS` explizit auf `Nil` gesetzt ist, dann holt libcurl die POST-Daten aus dem Lese-Callback. Wenn Sie einen Nullbyte-POST senden möchten, setzen Sie `#CURLOPT_POSTFIELDS` auf `""` (eine leere Zeichenkette) oder setzen Sie `#CURLOPT_POST` auf 1 und `#CURLOPT_POSTFIELDSIZE` auf 0.

Die Verwendung von POST mit HTTP 1.1 impliziert die Verwendung eines "Expect: 100-continue" Headers und libcurl fügt diesen Header automatisch hinzu, wenn bekannt ist, dass der POST entweder größer als 1024 Byte oder wenn die erwartete Größe unbekannt ist. Sie können diesen Header mit `#CURLOPT_HTTPHEADER` wie gewohnt deaktivieren.

Um Multipart/Formdata POST (auch bekannt als RFC2388-Posts) zu erstellen, überprüfen Sie die Option `#CURLOPT_HTTPPOST` in Kombination mit `form:AddContent()`.

EINGABEN

`postdata` Eingabewert

5.165 easy:SetOpt_PostQuote

BEZEICHNUNG

easy:SetOpt_PostQuote – setzt die (S)FTP-Befehle zur Ausführung nach der Übertragung

ÜBERSICHT

`easy:SetOpt_PostQuote(ccmds)`

BESCHREIBUNG

Übergeben Sie eine Tabelle mit einer Liste von FTP- oder SFTP-Befehlen, die nach Ihrer FTP-Übertragungsanforderung an den Server übergeben werden sollen. Die Befehle werden nur ausgeführt, wenn kein Fehler aufgetreten ist. Die Tabelle sollte eine vollständig

gültige Liste der ordnungsgemäß ausgefüllten Felder enthalten, wie für `#CURLOPT_QUOTE` beschrieben ist.

Deaktivieren Sie diesen Vorgang, indem Sie erneut diese Option auf `Nil` setzen.

EINGABEN

`cmds` Eingabewert

5.166 easy:SetOpt_PostRedir

BEZEICHNUNG

easy:SetOpt_PostRedir – setzt die Vorgehensweise bei einer HTTP-POST-Umleitung

ÜBERSICHT

```
easy:SetOpt_PostRedir(bitmask)
```

BESCHREIBUNG

Übergeben Sie eine Bitmaske, um zu steuern, wie libcurl auf Umleitungungen nach POSTs reagiert, die eine 301-, 302- oder 303-Antwort zurückerhalten.

Ein Parameter mit gesetztem Bit 0 (Wert `#CURL_REDIR_POST_301`) weist die Bibliothek an, RFC 7231 (Abschnitt 6.4.2 bis 6.4.4) zu beachten und POST-Anforderungen bei einer 301-Umleitung nicht in GET-Anforderungen zu konvertieren. Das Setzen von Bit 1 (Wert `#CURL_REDIR_POST_302`) bewirkt, dass libcurl die Anforderungsmethode nach einer Umleitung von 302 beibehält, während das Setzen von Bit 2 (Wert `#CURL_REDIR_POST_303`) bewirkt, dass libcurl die Anforderungsmethode nach einer Umleitung von 303 beibehält. Der Wert `#CURL_REDIR_POST_ALL` ist eine Komfortfunktion, die alle drei Bits setzt.

Das Nicht-RFC-Verhalten ist in Webbrowsern allgegenwärtig, daher führt die Bibliothek die Konvertierung standardmäßig aus, um die Durchgängigkeit zu gewährleisten. Ein Server kann jedoch verlangen, dass ein POST nach einer solchen Umleitung ein POST bleibt. Diese Option ist nur beim Festlegen von `#CURLOPT_FOLLOWLOCATION` von Bedeutung.

EINGABEN

`bitmask` Eingabewert

5.167 easy:SetOpt_Pre_Proxy

BEZEICHNUNG

easy:SetOpt_Pre_Proxy – stellt den Prä-Proxy für die Verwendung ein

ÜBERSICHT

```
easy:SetOpt_Pre_Proxy(preproxy)
```

BESCHREIBUNG

Legen Sie im Parameter `preproxy` den Prä-Proxy fest, der für die bevorstehende Anforderung verwendet werden soll. Der Parameter sollte eine Zeichenkette sein, die den

Hostnamen oder die punktierte numerische IP-Adresse enthält. Eine numerische IPv6-Adresse muss in [Klammern] angegeben werden.

Fügen Sie zum Angeben der Portnummer in dieser Zeichenkette :[port] am Ende des Hostnamens an. Die Portnummer des Proxys kann optional mit der separaten Option `#CURLOPT_PROXYPORT` angegeben werden. Wenn nicht angegeben, verwendet libcurl standardmäßig Port 1080 für Proxys.

Ein Prä-Proxy ist ein SOCKS-Proxy, mit dem curl eine Verbindung herstellt, bevor eine Verbindung zu dem in der Option `#CURLOPT_PROXY` angegebenen HTTP(S)-Proxy hergestellt wird. Der Prä-Proxy kann nur ein SOCKS-Proxy sein.

Die Prä-Proxy-Zeichenkette sollte das Präfix [Schema]:// vorangestellt werden, um anzugeben, welche Art von Socks verwendet wird. Verwenden Sie socks4://, socks4a://, socks5:// oder socks5h:// (der letzte, der socks5 aktiviert und den Proxy zum Auflösen auffordert, auch bekannt als Typ `#CURLPROXY_SOCKS5_HOSTNAME`), um die jeweilige SOCKS-Version anzufordern die verwendet werden soll. Andernfalls wird standardmäßig SOCKS4 verwendet.

Wenn Sie die Prä-Proxy-Zeichenkette auf `""` (eine leere Zeichenkette) setzen, wird die Verwendung eines Prä-Proxy explizit deaktiviert.

EINGABEN

`preproxy` Eingabewert

5.168 easy:SetOpt_Prequote

BEZEICHNUNG

easy:SetOpt_Prequote – setzt die Befehle, die vor einer FTP-Übertragung ausgeführt werden sollen

ÜBERSICHT

```
easy:SetOpt_Prequote(cmds)
```

BESCHREIBUNG

Übergeben Sie eine Tabelle mit einer Liste von FTP-Befehlen, die nach dem Festlegen des Übertragungstyps an den Server übergeben werden sollen. Deaktivieren Sie diesen Vorgang erneut, indem Sie diese Option auf Null setzen.

Während `#CURLOPT_QUOTE` und `#CURLOPT_POSTQUOTE` für SFTP funktionieren, funktioniert diese Option nicht bei FTP.

EINGABEN

`cmds` Eingabewert

5.169 easy:SetOpt_ProgressFunction

BEZEICHNUNG

easy:SetOpt_ProgressFunction – bestimmt den Callback zur Fortschrittsanzeige-Funktion

ÜBERSICHT

```
easy:SetOpt_ProgressFunction(progress_callback[, userdata])
```

BESCHREIBUNG

Übergeben Sie eine Callback-Funktion. Diese Funktion wird von libcurl anstelle der internen Funktion mit häufigem Intervall aufgerufen. Während der Datenübertragung wird diese Funktion sehr häufig aufgerufen und in langen Zeiträumen, in denen nichts übertragen wird, kann der Aufruf auf etwa einen pro Sekunde verlangsamt werden.

Der Callback erhält vier Parameter: Der erste Parameter gibt die Gesamtzahl der Bytes an, die libcurl bei dieser Übertragung herunterladen soll. Der zweite Parameter ist die Anzahl der bisher heruntergeladenen Bytes. Der dritte Parameter gibt die Gesamtzahl der Bytes an, die libcurl bei dieser Übertragung erwartet und der vierte Parameter gibt die Anzahl der bisher hochgeladenen Bytes an. Wenn Sie das optionale Argument `userdata` übergeben, wird der in `userdata` übergebene Wert als fünfter Parameter an Ihre Callback-Funktion übergeben. Der Parameter `userdata` kann einen beliebigen Typ haben.

An den Callback übergebene unbekannte/nicht verwendete Argumentwerte werden auf Null gesetzt (wenn Sie nur Daten herunterladen, bleibt die Upload-Größe 0). Oft wird der Callback zuerst einmal oder mehrmals aufgerufen, bevor er die Datengrößen kennt, so dass das Programm dementsprechend erstellt werden muss, um das zu berücksichtigen.

Wenn Sie von diesem Callback einen Wert ungleich Null zurückgeben, bricht libcurl die Übertragung ab und gibt `#CURLE_ABORTED_BY_CALLBACK` zurück.

Wenn Sie Daten mit der Multi-Schnittstelle übertragen, wird diese Funktion nicht während Leerlaufzeiten aufgerufen, es sei denn, Sie rufen die entsprechende libcurl-Funktion auf, die die Übertragungen durchführt.

`#CURLOPT_NOPROGRESS` muss auf 0 gesetzt sein, damit diese Funktion tatsächlich aufgerufen wird.

EINGABEN

`progress_callback`
: Eingabewert

`userdata`
: optional: Benutzerdaten, die an die Callback-Funktion übergeben werden sollen

5.170 easy:SetOpt_Protocols

BEZEICHNUNG

easy:SetOpt_Protocols – stellt die erlaubten Protokolle ein

ÜBERSICHT

```
easy:SetOpt_Protocols(bitmask)
```

BESCHREIBUNG

Übergeben Sie einen Wert, der eine Bitmaske mit `#CURLPROTO_XXX` Definitionen enthält. Bei Verwendung dieser Bitmaske wird begrenzt, welche Protokolle libcurl bei der Übertragung verwenden darf. Auf diese Weise können Sie eine libcurl verwenden,

die eine Vielzahl von Protokollen unterstützt, bestimmte Übertragungen jedoch so einschränken, dass nur eine Teilmenge davon benutzt werden darf. Standardmäßig akzeptiert libcurl alle unterstützten Protokolle (`#CURLPROTO_ALL`). Siehe auch `#CURLOPT_REDIR_PROTOCOLS`.

Dies sind die verfügbaren Protokolldefinitionen:

```
#CURLPROTO_DICT
#CURLPROTO_FILE
#CURLPROTO_FTP
#CURLPROTO_FTPS
#CURLPROTO_GOPHER
#CURLPROTO_HTTP
#CURLPROTO_HTTPS
#CURLPROTO_IMAP
#CURLPROTO_IMAPS
#CURLPROTO_LDAP
#CURLPROTO_LDAPS
#CURLPROTO_POP3
#CURLPROTO_POP3S
#CURLPROTO_RTMP
#CURLPROTO_RTMPE
#CURLPROTO_RTMPS
#CURLPROTO_RTMPT
#CURLPROTO_RTMPTE
#CURLPROTO_RTMPTS
#CURLPROTO_RTSP
#CURLPROTO_SCP
#CURLPROTO_SFTP
#CURLPROTO_SMB
#CURLPROTO_SMBS
#CURLPROTO_SMTP
#CURLPROTO_SMTPS
#CURLPROTO_TELNET
#CURLPROTO_TFTP
```

EINGABEN

`bitmask` Eingabewert

5.171 easy:SetOpt_Proxy

BEZEICHNUNG

easy:SetOpt_Proxy – stellt den Proxy für die Verwendung ein

ÜBERSICHT

```
easy:SetOpt_Proxy(proxy)
```

BESCHREIBUNG

Stellen Sie den `proxy` ein, der für die anstehende Anforderung verwendet werden soll. Der Parameter sollte eine Zeichenkette sein, die den Hostnamen oder die punktierte numerische IP-Adresse enthält. Eine numerische IPv6-Adresse muss in [Klammern] angegeben werden.

Fügen Sie zum Angeben der Portnummer in dieser Zeichenkette :[port] am Ende des Hostnamens an. Die Portnummer des Proxys kann optional mit der separaten Option `#CURLOPT_PROXYPORT` angegeben werden. Wenn nicht angegeben, verwendet libcurl standardmäßig Port 1080 für Proxys.

Die Proxy-Zeichenkette kann das Präfix [Schema]:// vorangestellt werden, um anzugeben, welche Art von Proxy verwendet wird.

`http://` HTTP-Proxy. Standard, wenn kein Schema oder Proxy-Typ angegeben ist.

`https://` HTTPS-Proxy. (Hinzugefügt in 7.52.0 für OpenSSL, GnuTLS und NSS)

`socks4://`
SOCKS4 Proxy.

`socks4a://`
SOCKS4a Proxy. Proxy löst URL-Hostnamen auf.

`socks5://`
SOCKS5 Proxy.

`socks5h://`
SOCKS5-Proxy. Proxy löst URL-Hostnamen auf.

Ohne ein Schema-Präfix kann mit `#CURLOPT_PROXYTYPE` angegeben werden, welche Art von Proxy die Zeichenkette identifiziert.

Wenn Sie die Bibliothek anweisen, einen HTTP-Proxy zu verwenden, konvertiert libcurl Vorgänge transparent in HTTP, auch wenn Sie eine FTP-URL usw. angeben. Dies kann sich auf die anderen Funktionen der Bibliothek auswirken, die Sie verwenden können, z.B. `#CURLOPT_QUOTE` und ähnliche FTP Besonderheiten, die nur funktionieren, wenn Sie über den HTTP-Proxy tunneln. Ein solches Tunneln wird mit `#CURLOPT_HTTPPROXYTUNNEL` aktiviert.

Wenn Sie die Proxy-Zeichenkette auf `""` (eine leere Zeichenfolge) setzen, wird die Verwendung eines Proxys explizit deaktiviert, auch wenn eine Umgebungsvariable dafür festgelegt ist.

Eine Proxy-Host-Zeichenkette kann auch ein Protokollschema (http://) und einen eingebetteten Benutzer plus Passwort enthalten.

EINGABEN

`proxy` Eingabewert

5.172 easy:SetOpt_Proxy_CAInfo

BEZEICHNUNG

easy:SetOpt_Proxy_CAInfo – setzt den Pfad zum Proxy Certificate Authority (CA)-Paket

ÜBERSICHT

```
easy:SetOpt_Proxy_CAInfo(path)
```

BESCHREIBUNG

Diese Option dient zum Herstellen einer Verbindung zu einem HTTPS-Proxy und nicht zu einem HTTPS-Server.

Übergeben Sie eine Zeichenkette mit dem Namen einer Datei, die ein oder mehrere Zertifikate enthält, um den HTTPS-Proxy mit zu überprüfen.

Wenn `#CURLOPT_PROXY_SSL_VERIFYPEER` Null ist und Sie die Überprüfung des Serverzertifikats vermeiden, muss `#CURLOPT_PROXY_CAINFO` nicht einmal eine zugreifbare Datei angeben.

Diese Option ist standardmäßig auf den Systempfad festgelegt, in dem das CAcert-Paket von libcurl gespeichert werden soll, wie dies zum Zeitpunkt der Erstellung festgelegt wurde.

Wenn curl für die NSS SSL-Bibliothek erstellt wird, muss das NSS PEM PKCS#11-Modul (libnsspem.so) verfügbar sein, damit diese Option ordnungsgemäß funktioniert.

(nur iOS und MacOS) Wenn curl für Sichere Übertragung erstellt wurde, wird diese Option aus Gründen der Abwärtskompatibilität mit anderen SSL-Systemen unterstützt, sollte jedoch nicht festgelegt werden. Wenn die Option nicht aktiviert ist, verwendet curl die Zertifikate im System und die Schlüsselkette des Benutzers, um den Peer zu überprüfen. Dies ist die bevorzugte Methode zum Überprüfen der Zertifikatkette des Peers.

EINGABEN

`path` Eingabewert

5.173 easy:SetOpt_Proxy_CAPath

BEZEICHNUNG

easy:SetOpt_Proxy_CAPath – gibt ein Verzeichnis mit Proxy-CA-Zertifikaten an

ÜBERSICHT

```
easy:SetOpt_Proxy_CAPath(capath)
```

BESCHREIBUNG

Übergeben Sie eine Zeichenkette, die ein Verzeichnis mit mehreren CA-Zertifikaten benennt, um den HTTPS-Proxy mit zu überprüfen. Wenn libcurl für OpenSSL erstellt wird, muss das Zertifikatverzeichnis mit dem Dienstprogramm openssl c_rehash vorbereitet werden. Dies ist nur dann sinnvoll, wenn `#CURLOPT_PROXY_SSL_VERIFYPEER` aktiviert ist (was standardmäßig der Fall ist).

EINGABEN

`capath` Eingabewert

5.174 easy:SetOpt_Proxy_CRLFile

BEZEICHNUNG

easy:SetOpt_Proxy_CRLFile – gibt eine Datei für Proxy-Zertifikatssperrlisten an

ÜBERSICHT

```
easy:SetOpt_Proxy_CRLFile(file)
```

BESCHREIBUNG

Diese Option dient zum Herstellen einer Verbindung zu einem HTTPS-Proxy und nicht zu einem HTTPS-Server.

Übergeben Sie im Parameter `file` eine Zeichenkette, die eine Datei mit der Verkettung der Zertifikatsperrliste (im PEM-Format) benennt und bei der Zertifikatüberprüfung verwendet wird, die während des SSL-Austauschs erfolgt.

Wenn curl für die Verwendung von NSS oder GnuTLS erstellt wurde, kann die Verwendung der CRL, die zur Unterstützung des Überprüfungsprozesses übergeben wurde, nicht beeinflusst werden. Wenn libcurl mit OpenSSL-Unterstützung erstellt wird, sind `X509_V_FLAG_CRL_CHECK` und `X509_V_FLAG_CRL_CHECK_ALL` festgelegt, sodass eine CRL-Prüfung für alle Elemente der Zertifikatkette erforderlich ist, wenn eine CRL-Datei übergeben wird.

Diese Option ist nur in Kombination mit der Option `#CURLOPT_PROXY_SSL_VERIFYPEER` sinnvoll.

Mit der Option wird ein bestimmter Fehlercode (`#CURLE_SSL_CRL_BADFILE`) definiert. Er wird zurückgegeben, wenn der SSL-Austausch fehlschlägt, da die CRL-Datei nicht geladen werden kann. Ein Fehler bei der Zertifikatüberprüfung aufgrund von Sperrinformationen in der CRL löst diesen angegebenen Fehler nicht aus.

EINGABEN

`file` Eingabewert

5.175 easy:SetOpt_Proxy_KeyPasswd

BEZEICHNUNG

easy:SetOpt_Proxy_KeyPasswd – setzt die Passphrase auf privaten Proxy-Schlüssel

ÜBERSICHT

```
easy:SetOpt_Proxy_KeyPasswd(pwd)
```

BESCHREIBUNG

Diese Option dient zum Herstellen einer Verbindung zu einem HTTPS-Proxy und nicht zu einem HTTPS-Server.

Übergeben Sie eine Zeichenkette als Parameter. Es wird als Passphrase verwendet, das für die Verwendung des privaten Schlüssels `#CURLOPT_PROXY_SSLKEY` erforderlich ist. Sie haben nie eine Passphrase gebraucht, um ein Zertifikat zu laden, aber Sie brauchen eine, um Ihren privaten Schlüssel zu laden.

EINGABEN

`pwd` Eingabewert

5.176 easy:SetOpt_Proxy_PinnedPublicKey

BEZEICHNUNG

easy:SetOpt_Proxy_PinnedPublicKey – legt das Public Key Pinning für https-Proxy fest

ÜBERSICHT

```
easy:SetOpt_Proxy_PinnedPublicKey(pinnedpubkey)
```

BESCHREIBUNG

Übergeben Sie eine Zeichenkette als Parameter. Die Zeichenkette kann der Dateiname Ihres Public Key Pinning sein. Das erwartete Dateiformat ist "PEM" oder "DER". Die Zeichenkette kann auch eine beliebige Anzahl von Base64-codierten sha256-Hashes sein, denen "sha256//" vorangestellt und durch ";" getrennt wird.

Beim Übertragung einer TLS- oder SSL-Verbindung sendet der https-Proxy ein Zertifikat, das seine Identität angibt. Ein öffentlicher Schlüssel (Public Key) wird aus diesem Zertifikat ausgelesen. Wenn er nicht genau mit dem für diese Option bereitgestellten öffentlichen Schlüssel übereinstimmt, bricht curl die Verbindung ab, bevor Daten gesendet oder empfangen werden.

Bei Nichtübereinstimmung wird `#CURLE_SSL_PINNEDPUBKEYNOTMATCH` zurückgegeben.

EINGABEN

`pinnedpubkey`
: Eingabewert

5.177 easy:SetOpt_Proxy_Service_Name

BEZEICHNUNG

easy:SetOpt_Proxy_Service_Name – setzt den Namen des Proxy-Authentifizierungsdienstes

ÜBERSICHT

```
easy:SetOpt_Proxy_Service_Name(name)
```

BESCHREIBUNG

Übergeben Sie eine Zeichenkette als Parameter in `name` an eine Zeichenkette, die den des Dienstes enthält. Der Standarddienstname lautet "HTTP" für HTTP-basierte Proxys und "rcmd" für SOCKS5. Mit dieser Option können Sie das ändern.

EINGABEN

`name` Eingabewert

5.178 easy:SetOpt_Proxy_SSL_Cipher_List

BEZEICHNUNG

easy:SetOpt_Proxy_SSL_Cipher_List – legt die für Proxy-TLS zu verwendenden Verschlüsselungsart fest

ÜBERSICHT

```
easy:SetOpt_Proxy_SSL_Cipher_List(list)
```

BESCHREIBUNG

Übergeben Sie eine Zeichenkette mit der Liste der Verschlüsselungsarten, die für die Verbindung zum HTTPS-Proxy verwendet werden sollen. Die Liste muss syntaktisch korrekt sein und aus einer oder mehreren durch Doppelpunkte getrennten Zeichenketten bestehen. Kommas oder Leerzeichen sind ebenfalls akzeptierte Trennzeichen, aber normalerweise werden Doppelpunkte verwendet. Als Operatoren können ! und - sowie + verwendet werden.

Zu den gültigen Beispielen für OpenSSL- und GnuTLS-Verschlüsselungslisten gehören RC4-SHA, SHA1 + DES, TLSv1 und DEFAULT. Die Standardliste wird normalerweise festgelegt, wenn Sie OpenSSL kompilieren.

Weitere Informationen zu Verschlüsselungslisten finden Sie unter dieser URL: https://www.openssl.org/docs/apps/ciphers.html

Gültige Beispiele für Verschlüsselungslisten für NSS sind "rsa_rc4_128_md5", "rsa_aes_128_sha" usw. Mit NSS können Sie keine Verschlüsselungsart hinzufügen/entfernen. Wenn man diese Option verwendet, werden alle bekannten Verschlüsselungsart deaktiviert und nur die übergebenen aktiviert.

Weitere Details zu den NSS-Verschlüsselungslisten finden Sie unter dieser URL: http://git.fedorahosted.org/cgit/mod_nss.git/plain/docs/mod_nss.html#Directives

EINGABEN

`list` Eingabewert

5.179 easy:SetOpt_Proxy_SSL_Options

BEZEICHNUNG

easy:SetOpt_Proxy_SSL_Options – legt die Proxy-SSL-Verhaltensoptionen fest

ÜBERSICHT

```
easy:SetOpt_Proxy_SSL_Options(bitmask)
```

BESCHREIBUNG

Übergeben Sie einen Wert mit einer Bitmaske, um libcurl über bestimmte SSL-Verhaltensweisen zu informieren.

`#CURLSSLOPT_ALLOW_BEAST`

Weist libcurl an, keine Problemumgehungen für Sicherheitslücken in den Protokollen SSL3 und TLS1.0 zu verwenden. Wenn diese Option nicht verwendet wird oder dieses Bit auf 0 gesetzt ist, verwendet die von libcurl verwendete SSL-Ebene möglicherweise eine Problemumgehung für diesen Fehler, obwohl dies bei einigen (älteren) SSL-Implementierungen zu Interoperabilitätsproblemen führen kann. WARNUNG: Wenn Sie dieses Umgehen vermeiden, wird die Sicherheit beeinträchtigt. Wenn Sie diese Option auf 1 setzen, werden Sie genau danach gefragt. Diese Option wird nur für DarwinSSL, NSS und OpenSSL unterstützt.

`#CURLSSLOPT_NO_REVOKE`

Weist libcurl an, die Zertifikatsperrüberprüfung für die SSL-Backends zu deaktivieren, in denen ein solches Verhalten vorliegt. Derzeit wird diese Op-

tion nur für Kanäle (die native Windows-SSL-Bibliothek) unterstützt, mit Ausnahme der Blacklist "Untrusted Publishers" von Windows, die anscheinend nicht umgangen werden kann. Diese Option bietet möglicherweise eine umfassendere Unterstützung für künftige andere SSL-Backends. https://curl.haxx.se/docs/ssl-compared.html

EINGABEN

`bitmask` Eingabewert

5.180 easy:SetOpt_Proxy_SSL_VerifyHost

BEZEICHNUNG

easy:SetOpt_Proxy_SSL_VerifyHost – überprüft den Namen des Proxy-Zertifikats anhand des Hosts

ÜBERSICHT

```
easy:SetOpt_Proxy_SSL_VerifyHost(verify)
```

BESCHREIBUNG

Übergeben Sie einen Wert auf 2, um curl anzuweisen, in den Zertifikatsnamenfeldern des HTTPS-Proxys den Proxy-Namen zu überprüfen.

Diese Option bestimmt, ob libcurl überprüft, ob das Proxy-Zertifikat den korrekten Namen für den Namen enthält, unter dem es bekannt ist.

Wenn `#CURLOPT_PROXY_SSL_VERIFYHOST` 2 ist, muss das Proxy-Zertifikat angeben, dass der Server der Proxy ist, zu dem Sie eine Verbindung herstellen möchten, oder die Verbindung schlägt fehl.

Curl betrachtet den Proxy als richtigen, wenn das Feld "Common Name" oder ein Feld "Subject Alternate Name" im Zertifikat mit dem Hostnamen in der Proxy-Zeichenkette übereinstimmt, die Sie Curl zugewiesen haben.

Wenn der Wert von `verify` 1 ist, gibt `easy:SetOpt()` einen Fehler zurück und der Optionswert wird aus alten historischen Gründen nicht geändert.

Wenn der Wert für `verify` 0 ist, ist die Verbindung unabhängig von den im Zertifikat verwendeten Namen erfolgreich. Verwenden Sie diese Fähigkeit mit Vorsicht!

Siehe auch `#CURLOPT_PROXY_SSL_VERIFYPEER`, um die digitale Signatur des Proxy-Zertifikats zu überprüfen. Wenn libcurl gegen NSS erstellt wird und `#CURLOPT_PROXY_SSL_VERIFYPEER` Null ist, wird `#CURLOPT_PROXY_SSL_VERIFYHOST` ebenfalls auf Null gesetzt und kann nicht überschrieben werden.

EINGABEN

`verify` Eingabewert

5.181 easy:SetOpt_Proxy_SSL_VerifyPeer

BEZEICHNUNG

easy:SetOpt_Proxy_SSL_VerifyPeer – aktiviert/deaktiviert die Überprüfung des SSL-Zertifikats des Proxys

ÜBERSICHT

```
easy:SetOpt_Proxy_SSL_VerifyPeer(verify)
```

BESCHREIBUNG

Im Parameter `verify` können Sie entweder 1 zum Aktivieren oder 0 zum Deaktivieren übergeben.

Diese Option weist curl an, die Gültigkeit des HTTPS-Proxy-Zertifikats zu überprüfen. Ein Wert von 1 bedeutet, dass curl überprüft wird. 0 (Null) bedeutet, dass dies nicht der Fall ist.

Dies ist die Proxy-Version von `#CURLOPT_SSL_VERIFYPEER`, die für normale HTTPS-Server verwendet wird.

Bei der Übertragung einer TLS- oder SSL-Verbindung sendet der Server ein Zertifikat, das seine Identität angibt. Curl überprüft, ob das Zertifikat authentisch ist, d.h. dass Sie darauf vertrauen können, dass der Server derjenige ist, von dem das Zertifikat sagt, dass er es ist. Diese Vertrauensstellung basiert auf einer Kette digitaler Signaturen, die auf von Ihnen bereitgestellten Zertifizierungsstellen-Zertifikaten (CA-Zertifikaten) basieren. Curl verwendet ein Standardpaket von CA-Zertifikaten (der Pfad dafür wird zum Zeitpunkt der Erstellung festgelegt). Sie können alternative Zertifikate mit der Option `#CURLOPT_PROXY_CAINFO` oder der Option `#CURLOPT_PROXY_CAPATH` angeben.

Wenn `#CURLOPT_PROXY_SSL_VERIFYPEER` aktiviert ist und bei der Überprüfung nicht nachgewiesen werden kann, dass das Zertifikat echt ist, schlägt die Verbindung fehl. Wenn die Option Null ist, ist die Überprüfung des Peer-Zertifikats unabhängig davon erfolgreich.

Die Authentifizierung des Zertifikats reicht nicht aus, um den Server zu identifizieren. In der Regel möchten Sie auch sicherstellen, dass der Server der Server ist, mit dem Sie kommunizieren möchten. Verwenden Sie dazu `#CURLOPT_PROXY_SSL_VERIFYHOST`. Die Überprüfung, ob der Hostname im Zertifikat für den Hostnamen gültig ist, zu dem Sie eine Verbindung herstellen, erfolgt unabhängig von der Option `#CURLOPT_PROXY_SSL_VERIFYPEER`.

WARNUNG: Wenn Sie die Überprüfung des Zertifikats deaktivieren, können Unbefugte die Kommunikation direkt ausführen, ohne dass Sie es merken. Durch Deaktivieren der Überprüfung wird die Kommunikation unsicher. Die Verschlüsselung einer Übertragung allein reicht nicht aus, da Sie nicht sicher sein können, ob Sie mit dem richtigen Endpunkt kommunizieren.

EINGABEN

`verify` Eingabewert

5.182 easy:SetOpt_Proxy_SSLCert

BEZEICHNUNG

easy:SetOpt_Proxy_SSLCert – legt das SSL-Proxy-Client-Zertifikat fest

ÜBERSICHT

```
easy:SetOpt_Proxy_SSLCert(cert)
```

BESCHREIBUNG

Diese Option dient zum Herstellen einer Verbindung zu einem HTTPS-Proxy und nicht zu einem HTTPS-Server.

Übergeben Sie eine Zeichenkette als Parameter. Die Zeichenkette sollte der Dateiname Ihres Client-Zertifikats sein, das zum Herstellen einer Verbindung zum HTTPS-Proxy verwendet wird. Das Standardformat ist "P12" für Sichere Übertragungen und "PEM" für andere Systeme und kann mit `#CURLOPT_PROXY_SSLCERTTYPE` geändert werden.

Bei NSS oder Sicheren Übertragungen kann dies auch der Kurzname des Zertifikats sein, mit dem Sie sich authentifizieren möchten, wie er in der Sicherheitsdatenbank angegeben ist. Wenn Sie eine Datei aus dem aktuellen Verzeichnis verwenden möchten, müssen Sie ihr das Präfix "./" voranstellen, um Verwechslungen mit einem Kurznamen zu vermeiden.

Wenn Sie ein Client-Zertifikat verwenden, müssen Sie höchstwahrscheinlich auch einen privaten Schlüssel mit `#CURLOPT_PROXY_SSLKEY` bereitstellen.

EINGABEN

`cert` Eingabewert

5.183 easy:SetOpt_Proxy_SSLCertType

BEZEICHNUNG

easy:SetOpt_Proxy_SSLCertType – setzt den Typ des Proxy-Client-SSL-Zertifikats

ÜBERSICHT

```
easy:SetOpt_Proxy_SSLCertType(type)
```

BESCHREIBUNG

Übergeben Sie eine Zeichenkette als Parameter. Die Zeichenkette sollte das Format Ihres Client-Zertifikats sein, das beim Herstellen einer Verbindung zu einem HTTPS-Proxy verwendet wird.

Unterstützte Formate sind "PEM" und "DER", außer bei Sicheren Übertragungen. OpenSSL (Versionen 0.9.3 und höher) und Sicheren Übertragungen (iOS 5 oder höher oder OS X 10.7 oder höher) unterstützen auch "P12" für PKCS#12-codierte Dateien.

EINGABEN

`type` Eingabewert

5.184 easy:SetOpt_Proxy_SSLKey

BEZEICHNUNG

easy:SetOpt_Proxy_SSLKey – setzt eine private Schlüsseldatei für das TLS- und SSL-Proxy-Client-Zertifikat

ÜBERSICHT

```
easy:SetOpt_Proxy_SSLKey(keyfile)
```

BESCHREIBUNG

Übergeben Sie eine Zeichenkette als Parameter. Die Zeichenkette sollte der Dateiname Ihres privaten Schlüssels sein, der für die Verbindung zum HTTPS-Proxy verwendet wird.

Das Standardformat ist "PEM" und kann mit #CURLOPT_PROXY_SSLKEYTYPE geändert werden.

(Nur iOS und Mac OS X) Diese Option wird ignoriert, wenn Curl für Sichere Übertragungen erstellt wurde. Sichere Übertragungen erwarten, dass der private Schlüssel bereits in der Verschlüsselungsdatei oder in der PKCS#12-Datei vorhanden ist, die das Zertifikat enthält.

EINGABEN

`keyfile` Eingabewert

5.185 easy:SetOpt_Proxy_SSLKeyType

BEZEICHNUNG

easy:SetOpt_Proxy_SSLKeyType – legt den Typ der privaten Proxy-Schlüsseldatei fest

ÜBERSICHT

```
easy:SetOpt_Proxy_SSLKeyType(type)
```

BESCHREIBUNG

Diese Option dient zum Herstellen einer Verbindung zu einem HTTPS-Proxy und nicht zu einem HTTPS-Server.

Übergeben Sie eine Zeichenkette als Parameter. Die Zeichenkette sollte das Format Ihres privaten Schlüssels haben. Unterstützte Formate sind "PEM", "DER" und "ENG".

EINGABEN

`type` Eingabewert

5.186 easy:SetOpt_Proxy_SSLVersion

BEZEICHNUNG

easy:SetOpt_Proxy_SSLVersion – legt die bevorzugte Proxy-TLS/SSL-Version fest

ÜBERSICHT

```
easy:SetOpt_Proxy_SSLVersion(version)
```

BESCHREIBUNG

Übergeben Sie einen Wert als Parameter, um zu steuern, welche Version von SSL/TLS verwendet werden soll, wenn eine Verbindung zu einem HTTPS-Proxy hergestellt wird.

Verwenden Sie dazu eine der verfügbaren Definitionen. Die verfügbaren Optionen sind:

`#CURL_SSLVERSION_DEFAULT`
: Die Standardaktion. Dadurch wird versucht, die Remote-SSL-Protokollversion zu ermitteln.

`#CURL_SSLVERSION_TLSv1`
: TLSv1.x

`#CURL_SSLVERSION_TLSv1_0`
: TLSv1.0

`#CURL_SSLVERSION_TLSv1_1`
: TLSv1.1

`#CURL_SSLVERSION_TLSv1_2`
: TLSv1.2

`#CURL_SSLVERSION_TLSv1_3`
: TLSv1.3

Die maximale TLS-Version kann mit Hilfe eines der untenstehenden Makros `#CURL_SSLVERSION_MAX_` eingestellt werden. Es ist auch möglich, mit OR eines der #CURL_SSLVERSION_XXX-Makros mit einem der #CURL_SSLVERSION_MAX_XXX-Makros zu verwenden. Die MAX-Makros werden für WolfSSL nicht unterstützt.

`#CURL_SSLVERSION_MAX_DEFAULT`
: Dieses Flag definiert die maximal unterstützte TLS-Version als TLSv1.2 oder den Standardwert aus der SSL-Bibliothek. (Hinzugefügt in 7.54.0)

`#CURL_SSLVERSION_MAX_TLSv1_0`
: Dieses Flag definiert die maximal unterstützte TLS-Version als TLSv1.0. (Hinzugefügt in 7.54.0)

`#CURL_SSLVERSION_MAX_TLSv1_1`
: Dieses Flag definiert die maximal unterstützte TLS-Version als TLSv1.1. (Hinzugefügt in 7.54.0)

`#CURL_SSLVERSION_MAX_TLSv1_2`
: Dieses Flag definiert die maximal unterstützte TLS-Version als TLSv1.2. (Hinzugefügt in 7.54.0)

`#CURL_SSLVERSION_MAX_TLSv1_3`
: Dieses Flag definiert die maximal unterstützte TLS-Version als TLSv1.3. (Hinzugefügt in 7.54.0)

EINGABEN

`version` Eingabewert

5.187 easy:SetOpt_Proxy_TLSAuth_Password

BEZEICHNUNG

easy:SetOpt_Proxy_TLSAuth_Password – setzt das Passwort für die Proxy-TLS-Authentifizierung

ÜBERSICHT

```
easy:SetOpt_Proxy_TLSAuth_Password(pwd)
```

BESCHREIBUNG

Übergeben Sie eine Zeichenkette als Parameter, die das Passwort für die TLS-Authentifizierungsmethode enthält, die mit der Option `#CURLOPT_PROXY_TLSAUTH_TYPE` angegeben wurde. Dies erfordert auch, dass die Option `#CURLOPT_PROXY_TLSAUTH_USERNAME` festgelegt wird.

EINGABEN

`pwd` Eingabewert

5.188 easy:SetOpt_Proxy_TLSAuth_Type

BEZEICHNUNG

easy:SetOpt_Proxy_TLSAuth_Type – legt die Proxy-TLS-Authentifizierungsmethoden fest

ÜBERSICHT

```
easy:SetOpt_Proxy_TLSAuth_Type(type)
```

BESCHREIBUNG

Übergeben Sie eine Zeichenkette als Parameter. Die Zeichenkette sollte die Methode der TLS-Authentifizierung sein, die für die HTTPS-Verbindung verwendet wird. Unterstützte Methode ist `"SRP"`.

`SRP` TLS-SRP-Authentifizierung. Die sichere Remote-Passwortauthentifizierung für TLS ist in RFC5054 definiert und bietet gegenseitige Authentifizierung, wenn beide Seiten ein gemeinsames Passwort haben. Um TLS-SRP zu verwenden, müssen Sie auch die Werte `#CURLOPT_PROXY_TLSAUTH_USERNAME` und die Optionen für `#CURLOPT_PROXY_TLSAUTH_PASSWORD` festlegen.

EINGABEN

`type` Eingabewert

5.189 easy:SetOpt_Proxy_TLSAuth_UserName

BEZEICHNUNG

easy:SetOpt_Proxy_TLSAuth_UserName – setzt den Benutzername zur Verwendung für die Proxy-TLS-Authentifizierung

ÜBERSICHT

```
easy:SetOpt_Proxy_TLSAuth_UserName(user)
```

BESCHREIBUNG

Übergeben Sie eine Zeichenkette als Parameter mit dem Benutzernamen, der für die mit der Option `#CURLOPT_PROXY_TLSAUTH_TYPE` angegebene HTTPS-Proxy-TLS-Authentifizierungsmethode verwendet werden soll. Dies erfordert auch, dass die Option `#CURLOPT_PROXY_TLSAUTH_PASSWORD` festgelegt wird.

EINGABEN

`user` Eingabewert

5.190 easy:SetOpt_Proxy_Transfer_Mode

BEZEICHNUNG

easy:SetOpt_Proxy_Transfer_Mode – hängt den FTP-Übertragungsmodus an die URL für Proxy an

ÜBERSICHT

```
easy:SetOpt_Proxy_Transfer_Mode(enabled)
```

BESCHREIBUNG

Übergibt einen Wert. Wenn der Wert auf 1 (eins) gesetzt ist, weist er libcurl an, den Übertragungsmodus (binär oder ASCII) für FTP-Übertragungen über einen HTTP-Proxy durch Anhängen von ;type=a oder ;type=i an die URL festzulegen. Ohne diese Einstellung oder ohne den Wert 0 (Standardwert Null) hat `#CURLOPT_TRANSFERTEXT` keine Auswirkung, wenn FTP über einen Proxy ausgeführt wird. Beachten Sie, dass nicht alle Proxys diese Funktion unterstützen.

EINGABEN

`enabled` Eingabewert

5.191 easy:SetOpt_ProxyAuth

BEZEICHNUNG

easy:SetOpt_ProxyAuth – legt die HTTP-Proxy-Authentifizierungsmethoden für den Versuch fest

ÜBERSICHT

```
easy:SetOpt_ProxyAuth(bitmask)
```

BESCHREIBUNG

Übergeben Sie einen Wert als Parameter, der auf eine Bitmaske festgelegt ist, um libcurl anzuweisen, welche HTTP-Authentifizierungsmethode(n) für die Proxy-Authentifizierung verwendet werden sollen. Wenn mehr als ein Bit gesetzt ist, fragt libcurl zuerst die Seite ab, um festzustellen, welche Authentifizierungsmethoden unterstützt werden und wählt dann die beste aus, die Sie verwenden dürfen. Bei einigen Methoden führt dies zu einem zusätzlichen Netzwerk-Durchlauf. Stellen Sie den tatsächlichen Namen und das Passwort mit der Option `#CURLOPT_PROXYUSERPWD` ein.

Die Bitmaske kann erstellt werden, indem die vollständig aufgelisteten und in der Bedienungsanleitung `#CURLOPT_HTTPAUTH` beschriebenen Bits zusammengefügt werden.

EINGABEN

`bitmask` Eingabewert

5.192 easy:SetOpt_ProxyHeader

BEZEICHNUNG

easy:SetOpt_ProxyHeader – setzt die an den Proxy zu übergebenden benutzerdefinierte HTTP-Header

ÜBERSICHT

```
easy:SetOpt_ProxyHeader(headers)
```

BESCHREIBUNG

Übergeben Sie eine Tabelle mit einer Liste von HTTP-Headern, um Ihre an einen Proxy gesendete HTTP-Anfrage zu übergeben. Die Regeln für diese Liste sind identisch mit denen der Option `#CURLOPT_HTTPHEADER`.

Die mit dieser Option gesetzten Headern werden immer nur in Anfragen verwendet, die an einen Proxy gesendet werden - wenn auch eine Anfrage an einen Host gesendet wird.

Die erste Zeile in einer Anfrage (die die Methode enthält, normalerweise ein GET oder POST) ist KEIN Header und kann mit dieser Option nicht ersetzt werden. Nur die Zeilen, die auf die Anforderungszeile folgen, sind Header. Das Hinzufügen dieser Methodenzeile zu dieser Liste von Headern führt nur dazu, dass Ihre Anforderung einen ungültigen Header sendet.

Übergeben Sie einen Nullwert, um keine benutzerdefinierte Headers wiederherzustellen.

EINGABEN

`headers` Eingabewert

5.193 easy:SetOpt_ProxyPassword

BEZEICHNUNG

easy:SetOpt_ProxyPassword – setzt das Passwort für die Proxy-Authentifizierung

ÜBERSICHT

```
easy:SetOpt_ProxyPassword(pwd)
```

BESCHREIBUNG

Übergeben Sie eine Zeichenkette als Parameter, die auf das Passwort verweist, das für die Authentifizierung beim Proxy verwendet werden soll.

Die Option `#CURLOPT_PROXYPASSWORD` sollte in Verbindung mit der Option `#CURLOPT_PROXYUSERNAME` verwendet werden.

EINGABEN

`pwd` Eingabewert

5.194 easy:SetOpt_ProxyPort

BEZEICHNUNG

easy:SetOpt_ProxyPort – setzt die Portnummer für den Proxy

ÜBERSICHT

```
easy:SetOpt_ProxyPort(port)
```

BESCHREIBUNG

Übergeben Sie einen Wert mit dieser Option, um den Proxy-Port für die Verbindung festzulegen, es sei denn, er ist in der Proxy-Zeichenfolge `#CURLOPT_PROXY` angegeben oder verwendet 443 für https-Proxys und 1080 für alle anderen als Standard.

Die Portnummer ist 16 Bit und kann daher nicht größer als 65535 sein.

EINGABEN

`port` Eingabewert

5.195 easy:SetOpt_ProxyType

BEZEICHNUNG

easy:SetOpt_ProxyType – setzt den Proxy-Protokolltyp

ÜBERSICHT

```
easy:SetOpt_ProxyType(type)
```

BESCHREIBUNG

Übergeben Sie einen der folgenden Werte, um den Proxy-Typ festzulegen.

`#CURLPROXY_HTTP`
: HTTP-Proxy. Standard.

`#CURLPROXY_HTTPS`
: HTTPS-Proxy. (Hinzugefügt in 7.52.0 für OpenSSL, GnuTLS und NSS)

`#CURLPROXY_HTTP_1_0`
: HTTP 1.0-Proxy. Dies ist `#CURLPROXY_HTTP` sehr ähnlich, außer dass HTTP/1.0 für alle Verbindungstunnel verwendet wird. Die HTTP-Version der tatsächlichen HTTP-Anforderungen, die von `#CURLOPT_HTTP_VERSION` gesteuert werden, wird nicht geändert.

`#CURLPROXY_SOCKS4`
: SOCKS4 Proxy.

`#CURLPROXY_SOCKS4A`
: SOCKS4a Proxy. Proxy löst URL-Hostnamen auf.

`#CURLPROXY_SOCKS5`
: SOCKS5 Proxy.

`#CURLPROXY_SOCKS5_HOSTNAME`
: SOCKS5 Proxy. Proxy löst URL-Hostnamen auf.

Häufig ist es bequemer, den Proxy-Typ mit dem Schema-Teil der Zeichenkette `#CURLOPT_PROXY` anzugeben.

EINGABEN

`type` Eingabewert

5.196 easy:SetOpt_ProxyUserName

BEZEICHNUNG

easy:SetOpt_ProxyUserName – setzt den Benutzername für die Proxy-Authentifizierung

ÜBERSICHT

```
easy:SetOpt_ProxyUserName(username)
```

BESCHREIBUNG

Übergeben Sie eine Zeichenkette als Parameter, der auf den Benutzernamen zeigen soll, der für die Übertragung verwendet werden soll.

`#CURLOPT_PROXYUSERNAME` legt den Benutzernamen fest, der bei der Protokollauthentifizierung mit dem Proxy verwendet werden soll.

Verwenden Sie zum Angeben des Proxy-Passworts das Passwort `#CURLOPT_PROXYPASSWORD`.

EINGABEN

`username` Eingabewert

5.197 easy:SetOpt_ProxyUserPwd

BEZEICHNUNG

easy:SetOpt_ProxyUserPwd – setzt den Benutzernamen und das Passwort für die Proxy-Authentifizierung

ÜBERSICHT

```
easy:SetOpt_ProxyUserPwd(userpwd)
```

BESCHREIBUNG

Übergeben Sie als Parameter eine Zeichenkette, die [Benutzernamen]:[Passwort] sein sollte, um die Verbindung zum HTTP-Proxy herzustellen. Sowohl der Name als auch das Passwort werden vor der Verwendung per URL dekodiert. Um beispielsweise einen Doppelpunkt in den Benutzernamen aufzunehmen, sollten Sie ihn als %3A kodieren. (Vorsicht- Dies unterscheidet sich von der Verwendung von `#CURLOPT_USERPWD`.)

Verwenden Sie `#CURLOPT_PROXYAUTH`, um die Authentifizierungsmethode anzugeben.

EINGABEN

`userpwd` Eingabewert

5.198 easy:SetOpt_Put

BEZEICHNUNG

easy:SetOpt_Put – stellt eine HTTP-PUT-Anfrage (veraltet)

ÜBERSICHT

```
easy:SetOpt_Put(put)
```

BESCHREIBUNG

Ein auf 1 gesetzter Parameter weist die Bibliothek an, HTTP PUT zum Übertragen von Daten zu verwenden. Die Daten sollten mit `#CURLOPT_READDATA` und `#CURLOPT_INFILESIZE` gesetzt werden.

Diese Option ist seit Version 7.12.1 veraltet. Verwenden Sie #CURLOPT_UPLOAD!

EINGABEN

`put` Eingabewert

5.199 easy:SetOpt_Quote

BEZEICHNUNG

easy:SetOpt_Quote – setzt die (S)FTP-Befehle, die vor der Übertragung ausgeführt werden sollen

ÜBERSICHT

```
easy:SetOpt_Quote(cmds)
```

BESCHREIBUNG

Übergeben Sie eine Tabelle mit einer Liste von FTP- oder SFTP-Befehlen, die vor Ihrer Anforderung an den Server übergeben werden sollen. Dies erfolgt, bevor andere Befehle ausgegeben werden (auch vor dem CWD-Befehl für FTP). Deaktivieren Sie diesen Vorgang erneut, indem Sie diese Option auf Null setzen. Wenn Sie mit einem FTP-Server sprechen, stellen Sie dem Befehl ein Sternchen (*) voran, damit libcurl auch dann fortgesetzt wird, wenn der Befehl fehlschlägt, da libcurl standardmäßig beim ersten Fehlschlagen stoppt.

Der Satz gültiger FTP-Befehle hängt vom Server ab (eine Liste der obligatorischen Befehle finden Sie in RFC959).

Die gültigen SFTP-Befehle sind:

`"chgrp group file"`
: Der Befehl chgrp setzt die Gruppen-ID der vom Dateioperanden benannten Datei auf die vom Gruppenoperanden angegebene Gruppen-ID. Der Gruppenoperand ist eine dezimale ganzzahlige Gruppen-ID.

`"chmod mode file"`
: Der Befehl chmod ändert die Dateimodusbits der angegebenen Datei. Der Modusoperand ist eine oktale Ganzzahl-Modusnummer.

`"chown user file"`
: Der Befehl chown setzt den Eigentümer der vom Dateioperanden benannten Datei auf die vom Benutzeroperanden angegebene Benutzer-ID. Der Benutzeroperand ist eine dezimale Ganzzahl-Benutzer-ID.

`"ln source_file target_file"`
: Mit den Befehlen ln und symlink wird an der Position target_file eine symbolische Verknüpfung erstellt, die auf die Position source_file verweist.

`"mkdir directory_name"`
: Der Befehl mkdir erstellt das vom Operanden directory_name angegebene Verzeichnis.

`"pwd"` Der Befehl pwd gibt den absoluten Pfadnamen des aktuellen Arbeitsverzeichnisses zurück.

`"rename source target"`
: Der Befehl zum Umbenennen benennt die vom Quelloperanden benannte Datei oder das Verzeichnis in den vom Zieloperanden benannten Zielpfad um.

`"rm file"` Der Befehl rm entfernt die vom Dateioperanden angegebene Datei.

`"rmdir directory"`
: Der Befehl rmdir entfernt den vom Verzeichnisoperanden angegebenen Verzeichniseintrag, sofern er leer ist.

`"statvfs file"`
: Der Befehl statvfs gibt Statistiken zum Dateisystem zurück, in dem sich die angegebene Datei befindet. (Hinzugefügt in 7.49.0)

`"symlink source_file target_file"`
: Siehe ln.

EINGABEN

`cmds` Eingabewert

5.200 easy:SetOpt_Random_File

BEZEICHNUNG

easy:SetOpt_Random_File – gibt eine Quelle für zufällige Daten an

ÜBERSICHT

```
easy:SetOpt_Random_File(path)
```

BESCHREIBUNG

Übergeben Sie eine Zeichenkette an einen Dateinamen. Die Datei kann zum Lesen verwendet werden, um Zufallsgenerierungen für SSL und mehr zu verwenden.

EINGABEN

`path` Eingabewert

5.201 easy:SetOpt_Range

BEZEICHNUNG

easy:SetOpt_Range – stellt den anzuforderenden Bytebereich ein

ÜBERSICHT

```
easy:SetOpt_Range(range)
```

BESCHREIBUNG

Übergeben Sie eine Zeichenkette als Parameter, die den angegebenen Bereich enthalten soll, den Sie abrufen möchten. Es sollte das Format "X-Y" haben, wobei entweder X oder Y weggelassen werden können und X und Y Byte-Indizes sind.

HTTP-Übertragungen unterstützen auch mehrere Intervalle, die wie in "X-Y, N-M" durch Kommas getrennt sind. Bei Verwendung dieser Art von mehreren Intervallen sendet der HTTP-Server das Antwortdokument in Teilen (unter Verwendung von Standard-MIME-Trennverfahren). Leider erlaubt der HTTP-Standard (RFC 7233, Abschnitt 3.1) Servern, Bereichsanforderungen zu ignorieren. Selbst wenn Sie `#CURLOPT_RANGE` für eine Anforderung festlegen, wird möglicherweise die vollständige Antwort zurückgesendet.

Für RTSP sollte die Formatierung eines Bereichs gemäß RFC2326, Abschnitt 12.29 erfolgen. Für RTSP sind Bytebereiche nicht zulässig. Stattdessen sollten Bereiche in den Formaten npt, utc oder smpte angegeben werden.

Übergeben Sie eine Null an diese Option, um die Verwendung von Bereichen zu deaktivieren.

EINGABEN

`range` Eingabewert

5.202 easy:SetOpt_ReadFunction

BEZEICHNUNG

easy:SetOpt_ReadFunction – liest den Callback für Daten-Uploads

ÜBERSICHT

```
easy:SetOpt_ReadFunction(read_callback[, userdata])
```

BESCHREIBUNG

Übergeben Sie eine Callback-Funktion. Diese Callback-Funktion wird von libcurl aufgerufen, sobald Daten gelesen werden müssen, um sie an den Peer zu senden - wie wenn Sie ihn auffordern, Daten auf den Server hochzuladen oder zu senden.

Der erste Parameter, der an Ihre Callback-Funktion übergeben wird, ist eine Ganzzahl, die die Anzahl der zu lesenden Bytes enthält. Wenn Sie das optionale Argument `userdata` übergeben, wird der in `userdata` übergebene Wert als zweiter Parameter an Ihre Callback-Funktion übergeben. Der Parameter `userdata` kann einen beliebigen Typ haben.

Ihre Funktion muss eine Zeichenkette zurückgeben, die die gelesenen Daten enthält. Dies kann weniger Bytes als angefordert enthalten, es muss jedoch mindestens ein Byte in der Rückgabezeichenkette enthalten sein, sonst wird die Übertragung abgebrochen.

Wenn Sie die aktuelle Übertragung stoppen, indem Sie eine leere Zeichenkette zurückgeben (d.h. bevor der Server sie erwartet hat, z.B. wenn Sie angegeben haben, dass Sie N Bytes hochladen werden und Sie weniger als N Bytes hochladen), kann es vorkommen, dass der Server "hangs" (hängt) und auf den Rest der Daten wartet, die nicht kommen werden.

Der lesende Callback kann `#CURL_READFUNC_ABORT` zurückgeben, um die aktuelle Operation sofort zu stoppen, was zu einem #CURLE_ABORTED_BY_CALLBACK-Fehlercode der Übertragung führt.

Der Callback kann `#CURL_READFUNC_PAUSE` zurückgeben, um das Lesen von dieser Verbindung anzuhalten. Weitere Informationen finden Sie unter `easy:Pause()`.

Fehler: Wenn Sie TFTP-Uploads durchführen, müssen Sie genau die Datenmenge zurückgeben, die der Callback haben möchte. Andernfalls wird die Übertragung vom Server als endgültiges Paket betrachtet und endet dort.

EINGABEN

`read_callback`
: Eingabewert

`userdata`
: optional: Benutzerdaten, die an die Callback-Funktion übergeben werden sollen

BEISPIEL

```
Function p_ReadData(len)
   If readlen + len > totallen Then len = totallen - readlen
   If len > 0
      readlen = readlen + len
      Return(ReadBytes(1, len))
   Else
      Return("")
   EndIf
EndFunction
readlen = 0
totallen = FileLength(1)
e:SetOpt_ReadFunction(p_ReadData)
```

Der obige Code installiert eine Lesefunktion, die alle Daten aus der Datei mit dem Identifikator 1 liest.

5.203 easy:SetOpt_Redir_Protocols

BEZEICHNUNG

easy:SetOpt_Redir_Protocols – legt die Protokolle fest, zu denen umgeleitet werden darf

ÜBERSICHT

```
easy:SetOpt_Redir_Protocols(bitmask)
```

BESCHREIBUNG

Übergeben Sie einen Wert, der eine Bitmaske von `#CURLPROTO_XXX` enthält. Falls verwendet, begrenzt diese Bitmaske, welche Protokolle libcurl in einer Übertragung verwenden darf, zu der es bei einer Umleitung umleitet, wenn `#CURLOPT_FOLLOWLOCATION` aktiviert ist. Auf diese Weise können Sie bestimmte Übertragungen so beschränken, dass nur eine Teilmenge von Protokollen in Umleitungen verwendet werden darf.

Von `#CURLOPT_PROTOCOLS` abgelehnte Protokolle werden von dieser Option nicht überschrieben.

Standardmäßig lässt libcurl aus Sicherheitsgründen alle Protokolle bei der Umleitung zu, mit Ausnahme einiger deaktivierter Protokolle: Seit 7.19.4 sind FILE und SCP deaktiviert und seit 7.40.0 sind SMB und SMBS ebenfalls deaktiviert. `#CURLPROTO_ALL` aktiviert alle Protokolle bei der Umleitung, einschließlich der aus Sicherheitsgründen deaktivierten.

Dies sind die verfügbaren Protokolldefinitionen:

```
#CURLPROTO_DICT
#CURLPROTO_FILE
#CURLPROTO_FTP
#CURLPROTO_FTPS
#CURLPROTO_GOPHER
#CURLPROTO_HTTP
#CURLPROTO_HTTPS
#CURLPROTO_IMAP
#CURLPROTO_IMAPS
#CURLPROTO_LDAP
#CURLPROTO_LDAPS
#CURLPROTO_POP3
#CURLPROTO_POP3S
#CURLPROTO_RTMP
#CURLPROTO_RTMPE
#CURLPROTO_RTMPS
#CURLPROTO_RTMPT
#CURLPROTO_RTMPTE
#CURLPROTO_RTMPTS
#CURLPROTO_RTSP
#CURLPROTO_SCP
#CURLPROTO_SFTP
#CURLPROTO_SMB
#CURLPROTO_SMBS
#CURLPROTO_SMTP
#CURLPROTO_SMTPS
#CURLPROTO_TELNET
#CURLPROTO_TFTP
```

EINGABEN

`bitmask` Eingabewert

5.204 easy:SetOpt_Referer

BEZEICHNUNG

easy:SetOpt_Referer – setzt den HTTP Referer: Header

ÜBERSICHT

```
easy:SetOpt_Referer(where)
```

BESCHREIBUNG

Übergeben Sie eine Zeichenkette als Parameter. Hiermit wird die Referer: Header in der an den Remote-Server gesendeten http-Anfrage festgelegt. Dies kann verwendet werden, um Server oder Skripte zu täuschen. Sie können auch eine beliebig benutzerdefinierten Header mit `#CURLOPT_HTTPHEADER` festlegen.

EINGABEN

`where` Eingabewert

5.205 easy:SetOpt_Request_Target

BEZEICHNUNG

easy:SetOpt_Request_Target – gibt ein alternatives Ziel für diese Anforderung an

ÜBERSICHT

```
easy:SetOpt_Request_Target(string)
```

BESCHREIBUNG

Übergeben Sie eine Zeichenkette an die Zeichenkette, die libcurl in der anstehenden Anforderung anstelle des aus der URL ausgelesen Pfads verwendet.

EINGABEN

`string` Eingabewert

5.206 easy:SetOpt_Resolve

BEZEICHNUNG

easy:SetOpt_Resolve – gibt einen benutzerdefinierten Hostnamen für IP-Adressauflösungen an

ÜBERSICHT

```
easy:SetOpt_Resolve(hosts)
```

BESCHREIBUNG

Übergeben Sie eine Tabelle mit einer Liste von Zeichenketten mit Informationen zur Hostnamenauflösung, die für Anforderungen mit diesem Handle verwendet werden sollen.

Jede einzelne Zeichenkette für die Namensauflösung sollte im Format HOST:PORT:ADDRESS[,ADDRESS] ... geschrieben werden, wobei HOST der Name ist, den libcurl auflösen möchte. PORT ist die Portnummer des Dienstes, bei dem libcurl eine Verbindung zum HOST herstellen möchte und ADDRESS ist eine oder mehrere numerische IP-Adressen. Wenn Sie mehrere IP-Adressen angeben, müssen diese durch Kommas getrennt werden. Wenn libcurl zur Unterstützung von IPv6 erstellt wurde, kann jeder der ADDRESS-Einträge natürlich eine IPv4- oder eine IPv6-Adressierung aufweisen.

Diese Option füllt den DNS-Cache effektiv mit Einträgen für das Host+Port-Paar vor, sodass Umleitungen und alle Vorgänge mit dem Host+Port stattdessen die von Ihnen angegebene Adresse verwenden. Mit `#CURLOPT_RESOLVE` festgelegte Adressen werden nicht wie normale Einträge aus dem DNS-Cache gelöscht.

Wenn der DNS-Cache bereits einen Eintrag für das angegebene Host+Port-Paar enthält, wird dieser Eintrag entfernt und ein neuer Eintrag erstellt. Dies liegt daran, dass alte Einträge möglicherweise andere Adressen haben oder normale Einträge mit Zeitüberschreitungen sind.

Die von dieser Option festgelegte ADRESSE wird auch dann verwendet, wenn `#CURLOPT_IPRESOLVE` so eingestellt ist, dass libcurl eine andere IP-Version verwendet.

Entfernen Sie die Namen erneut aus dem DNS-Cache, um die Bereitstellung dieser falschen Auflösungen zu beenden, indem Sie eine Zeichenkette in die Liste aufnehmen, die das Format "-HOST:PORT" verwendet. Dem Hostnamen muss ein Bindestrich vorangestellt werden und der Hostname und die Portnummer müssen genau mit dem übereinstimmen, was zuvor bereits hinzugefügt wurde.

Unterstützung für die Angabe von ADRESS in [Klammern] wurde in 7.57.0 hinzugefügt.

Unterstützung für die Bereitstellung mehrerer IP-Adressen pro Eintrag wurde in 7.59.0 hinzugefügt.

EINGABEN

`hosts` Eingabewert

5.207 easy:SetOpt_Resume_From

BEZEICHNUNG

easy:SetOpt_Resume_From – legt die Position fest, von der aus die Übertragung fortgesetzt wird

ÜBERSICHT

```
easy:SetOpt_Resume_From(from)
```

BESCHREIBUNG

Übergeben Sie einen Wert als Parameter. Er enthält den Versatz in der Anzahl der Bytes, ab der die Übertragung beginnen soll. Setzen Sie diese Option auf 0, um die Übertragung von vorne zu beginnen (deaktiviert die Wiederaufnahme). Setzen Sie diese Option für FTP auf -1, damit die Übertragung am Ende der Zieldatei beginnt (nützlich, um einen unterbrochenen Upload fortzusetzen).

Beim Hochladen mit FTP befindet sich die Wiederaufnahmeposition in der Lokal-/Quelldatei, von der aus libcurl versuchen sollte, den Upload fortzusetzen. Anschließend wird die Quelldatei an die Remote-Zieldatei angehängt.

Wenn Sie eine Übertragung über das 2-GB-Limit hinaus fortsetzen müssen, verwenden Sie stattdessen `#CURLOPT_RESUME_FROM_LARGE`.

EINGABEN

`from` Eingabewert

5.208 easy:SetOpt_Resume_From_Large

BEZEICHNUNG

easy:SetOpt_Resume_From_Large – legt die Position fest, von der aus die Übertragung fortgesetzt wird

ÜBERSICHT

```
easy:SetOpt_Resume_From_Large(from)
```

BESCHREIBUNG

Übergeben Sie curl_off_t als Parameter. Es enthält den Versatz in der Anzahl der Bytes, ab der die Übertragung beginnen soll. Setzen Sie diese Option auf 0, um die Übertragung von vorne zu beginnen (deaktiviert die Wiederaufnahme). Setzen Sie diese Option für FTP auf -1, damit die Übertragung am Ende der Zieldatei beginnt (nützlich, um einen unterbrochenen Upload fortzusetzen).

Beim Hochladen mit FTP befindet sich die Wiederaufnahmeposition in der Lokal-/Quelldatei, von der aus libcurl versuchen sollte, den Upload fortzusetzen. Anschließend wird die Quelldatei an die Remote-Zieldatei angehängt.

EINGABEN

`from` Eingabewert

5.209 easy:SetOpt_RTSP_Client_CSeq

BEZEICHNUNG

easy:SetOpt_RTSP_Client_CSeq – legt die RTSP-Client-CSEQ-Nummer fest

ÜBERSICHT

```
easy:SetOpt_RTSP_Client_CSeq(cseq)
```

BESCHREIBUNG

Übergeben Sie einen Wert, um die CSEQ-Nummer festzulegen, die für die nächste RTSP-Anforderung ausgegeben werden soll. Nützlich, wenn die Anwendung eine zuvor unterbrochene Verbindung wiederherstellt. Der CSEQ erhöht sich von nun an von dieser neuen Nummer.

EINGABEN

`cseq` Eingabewert

5.210 easy:SetOpt_RTSP_Request

BEZEICHNUNG

easy:SetOpt_RTSP_Request – gibt die RTSP-Anfrage an

ÜBERSICHT

```
easy:SetOpt_RTSP_Request(request)
```

BESCHREIBUNG

Gibt libcurl an, welche Art von RTSP-Anfrage zu stellen ist. Übergeben Sie einen der folgenden RTSP-Aufzählungswerte als Wert im Argument `request`. Sofern nicht anders angegeben, muss für Befehle die Sitzungs-ID initialisiert werden.

`#CURL_RTSPREQ_OPTIONS`

Wird verwendet, um die verfügbaren Methoden des Servers abzurufen. Die Anwendung ist für das Parsen und Befolgen der Antwort verantwortlich. (Die Sitzungs-ID wird für diese Methode nicht benötigt.)

`#CURL_RTSPREQ_DESCRIBE`
: Wird verwendet, um die Beschreibung eines Streams auf niedriger Ebene abzurufen. Die Anwendung sollte beachten, welches Format im Header `'Accept:'` steht. Sofern nicht manuell festgelegt, füllt libcurl automatisch `'Accept:` application/sdp' aus. Time-condition Header werden zu Beschreibungsanforderungen hinzugefügt, wenn die Option `#CURLOPT_TIMECONDITION` aktiviert ist. (Die Sitzungs-ID wird für diese Methode nicht benötigt.)

`#CURL_RTSPREQ_ANNOUNCE`
: Wenn diese Methode von einem Client gesendet wird, ändert sie die Beschreibung der Sitzung. Wenn ein Client beispielsweise den Server zum Aufzeichnen einer Besprechung verwendet, kann der Client mit ANNOUNCE den Server über alle Metainformationen zur Sitzung informieren. ANNOUNCE verhält sich wie ein HTTP-PUT oder POST wie `#CURL_RTSPREQ_SET_PARAMETER`.

`#CURL_RTSPREQ_SETUP`
: Setup wird verwendet, um die Transportebene für die Sitzung zu initialisieren. Die Anwendung muss die gewünschten Transportoptionen für eine Sitzung mithilfe der Option `#CURLOPT_RTSP_TRANSPORT` festlegen, bevor Setup aufgerufen wird. Wenn derzeit keine Sitzungs-ID mit `#CURLOPT_RTSP_SESSION_ID` festgelegt ist, liest libcurl die Sitzungs-ID aus und verwendet sie in der Antwort für diese Anforderung. (Die Sitzungs-ID wird für diese Methode nicht benötigt.)

`#CURL_RTSPREQ_PLAY`
: Senden Sie einen Wiedergabebefehl an den Server. Verwenden Sie die Option `#CURLOPT_RANGE`, um die Wiedergabezeit zu ändern (z.B. 'npt=10-15').

`#CURL_RTSPREQ_PAUSE`
: Senden Sie einen Pause-Befehl an den Server. Verwenden Sie die Option `#CURLOPT_RANGE` mit einem einzelnen Wert, um anzugeben, wann der Stream angehalten werden soll. (z.B. npt='25 ')

`#CURL_RTSPREQ_TEARDOWN`
: Dieser Befehl beendet eine RTSP-Sitzung. Das einfache Schließen einer Verbindung beendet die RTSP-Sitzung nicht, da eine RTSP-Sitzung über verschiedene Verbindungen gesteuert werden kann.

`#CURL_RTSPREQ_GET_PARAMETER`
: Rufen Sie einen Parameter vom Server ab. Standardmäßig enthält libcurl bei allen nicht leeren Anforderungen automatisch eine Überschrift vom `Content-Typ`: text/parameters, sofern keine benutzerdefinierte festgelegt ist. GET_PARAMETER verhält sich wie ein HTTP PUT oder POST (siehe `#CURL_RTSPREQ_SET_PARAMETER`). Anwendungen, die eine Heartbeat-Nachricht senden möchten (z.B. bei einer vom Server angegebenen Zeitüberschreitung), sollten eine leere GET_PARAMETER-Anforderung senden.

`#CURL_RTSPREQ_SET_PARAMETER`
: Legen Sie einen Parameter auf dem Server fest. Standardmäßig enthält libcurl automatisch einen Header für `Content-Type`: text/parameters, sofern keine benutzerdefinierte festgelegt ist. Die Interaktion mit SET_PARAMETER ähnelt einem HTTP PUT oder POST. Eine Anwendung kann entweder `#CURLOPT_UPLOAD` mit `#CURLOPT_READDATA` wie ein HTTP-PUT oder `#CURLOPT_POSTFIELDS` wie ein HTTP-POST verwenden. Mehrteilige-Übertragungen sind nicht zulässig, daher muss die Anwendung im ersteren den Wert `#CURLOPT_INFILESIZE` und im letzteren den Wert `#CURLOPT_POSTFIELDSIZE` festlegen. Außerdem werden in RTSP keine mehrteiligen POSTs verwendet.

`#CURL_RTSPREQ_RECORD`
: Wird verwendet, um den Server anzuweisen, eine Sitzung aufzuzeichnen. Verwenden Sie die Option `#CURLOPT_RANGE`, um die Aufnahmezeit zu ändern.

`#CURL_RTSPREQ_RECEIVE`
: Dies ist eine spezielle Anforderung, da keine Daten an den Server gesendet werden. Die Anwendung kann diese Funktion aufrufen, um verschachtelte RTP-Daten zu empfangen. Es wird nach der Verarbeitung eines Lesepuffers von Daten zurückgegeben, um der Anwendung die Möglichkeit zu geben, ausgeführt zu werden.

EINGABEN

`request` Eingabewert

5.211 easy:SetOpt_RTSP_Server_CSeq

BEZEICHNUNG

easy:SetOpt_RTSP_Server_CSeq – stellt die CSEQ-Nummer des RTSP-Servers ein

ÜBERSICHT

```
easy:SetOpt_RTSP_Server_CSeq(cseq)
```

BESCHREIBUNG

Übergeben Sie einen Wert, um die CSEQ-Nummer festzulegen, die für die nächste RTSP-Server->Client-Anforderung erwartet wird. HINWEIS: Diese Funktion (Überwachen von Serveranforderungen) ist nicht implementiert.

EINGABEN

`cseq` Eingabewert

5.212 easy:SetOpt_RTSP_Session_ID

BEZEICHNUNG

easy:SetOpt_RTSP_Session_ID – legt die RTSP-Sitzungs-ID fest

ÜBERSICHT

```
easy:SetOpt_RTSP_Session_ID(id)
```

BESCHREIBUNG

Übergeben Sie eine Zeichenfolge als Parameter, um den Wert der aktuellen RTSP-Sitzungs-ID für den Handle festzulegen. Nützlich, um eine laufende Sitzung fortzusetzen. Sobald dieser Wert auf einen Nicht-Null-Wert gesetzt ist, gibt libcurl `#CURLE_RTSP_SESSION_ERROR` zurück, wenn die vom Server empfangene ID nicht übereinstimmt. Wenn nicht gesetzt (oder auf Null gesetzt), setzt libcurl die ID automatisch, wenn der Server sie zum ersten Mal in einer Antwort setzt.

EINGABEN

`id` Eingabewert

5.213 easy:SetOpt_RTSP_Stream_URI

BEZEICHNUNG

easy:SetOpt_RTSP_Stream_URI – stellt die RTSP-Stream-URI ein

ÜBERSICHT

```
easy:SetOpt_RTSP_Stream_URI(URI)
```

BESCHREIBUNG

Stellen Sie den Stream `URI` so ein, dass er bearbeitet wird, indem Sie eine Zeichenkette übergeben. Beispielsweise kann eine einzelne Sitzung rtsp://foo/twister/audio und rtsp://foo/twister/video steuern und die Anwendung kann mit dieser Option zum entsprechenden Stream wechseln. Wenn diese Option deaktiviert ist, verwendet libcurl standardmäßig generische Serveroptionen, indem anstelle des RTSP-Stream-URI '*' übergeben wird. Diese Option unterscheidet sich von `#CURLOPT_URL`. Bei der Arbeit mit RTSP gibt `#CURLOPT_RTSP_STREAM_URI` an, welche URL in dem Anforderungs-Header an den Server gesendet werden soll, während `#CURLOPT_URL` angibt, wo die Verbindung hergestellt werden soll. (z.B. könnte die `#CURLOPT_URL` für die obigen Beispiele auf rtsp://foo/twister gesetzt sein.)

EINGABEN

`URI` Eingabewert

5.214 easy:SetOpt_RTSP_Transport

BEZEICHNUNG

easy:SetOpt_RTSP_Transport – setzt den RTSP Transport: Header

ÜBERSICHT

```
easy:SetOpt_RTSP_Transport(transport)
```

BESCHREIBUNG

Übergeben Sie eine Zeichenkette, um libcurl mitzuteilen, was für den Transport: Header für diese RTSP-Sitzung übergeben werden soll. Dies ist hauptsächlich eine bequeme Methode, um zu vermeiden, dass für jede SETUP-Anforderung einen benutzerdefinierten Transport: Header festgelegt werden muss. Die Anwendung muss einen Transport: Header festlegen, bevor eine SETUP-Anforderung ausgegeben wird.

EINGABEN

`transport`
Eingabewert

5.215 easy:SetOpt_SASL_IR

BEZEICHNUNG

easy:SetOpt_SASL_IR – aktiviert das Senden der ersten Antwort im ersten Paket

ÜBERSICHT

```
easy:SetOpt_SASL_IR(enable)
```

BESCHREIBUNG

Übergibt einen Wert. Wenn der Wert 1 ist, sendet curl die erste Antwort im ersten Authentifizierungspaket an den Server, um die Anzahl der Ping-Pong-Anforderungen zu verringern. Gilt nur für die folgenden unterstützenden SASL-Authentifizierungsmechanismen:

```
* Login
* Plain
* GSSAPI
* NTLM
* OAuth 2.0
```

Hinweis: Obwohl IMAP diese Option unterstützt, muss sie nicht explizit festgelegt werden, da libcurl die Funktion selbst bestimmen kann, wenn der Server SASL-IR CAPABILITY unterstützt.

EINGABEN

`enable` Eingabewert

5.216 easy:SetOpt_SeekFunction

BEZEICHNUNG

easy:SetOpt_SeekFunction – legt den Benutzer-Callback zum Suchen im Eingabedatenstrom fest

ÜBERSICHT

```
easy:SetOpt_SeekFunction(seek_callback[, userdata])
```

BESCHREIBUNG

Übergeben Sie eine Callback-Funktion. Diese Funktion wird von libcurl aufgerufen, um nach einer bestimmten Position im Eingabedatenstrom zu suchen und kann verwendet werden, um eine Datei in einem erneuten Upload vorwärtszuspulen (anstatt alle hochgeladenen Bytes mit der normalen Lesefunktion/Callback zu lesen). Sie wird auch aufgerufen, um einen Datenstrom zurückzuspulen, wenn bereits Daten an den Server gesendet wurden und erneut gesendet werden müssen. Dies kann passieren, wenn ein HTTP-PUT oder -POST mit einer Authentifizierungsmethode mit mehreren Durchläufen ausgeführt

wird oder wenn eine vorhandene HTTP-Verbindung zu spät wiederverwendet wird und der Server die Verbindung schließt.

Die Funktion empfängt zwei Argumente: Das erste Argument gibt den zu suchenden Versatz an, das zweite Argument den Ursprung des im ersten Argument übergebenen Versatzes. Dies wird eine der folgenden speziellen Zeichenketten sein:

`set` Der Versatz ist relativ zum Anfang.

`cur` Der Versatz ist relativ zur aktuellen Position.

`end` Der Versatz ist relativ zum Ende.

Wenn Sie das optionale Argument `userdata` übergeben, wird der in `userdata` übergebene Wert als dritter Parameter an Ihre Callback-Funktion übergeben. Der Parameter `userdata` kann einen beliebigen Typ haben.

Die Callback-Funktion muss bei Erfolg `#CURL_SEEKFUNC_OK` (oder nichts) zurückgeben, `#CURL_SEEKFUNC_FAIL`, damit der Upload-Vorgang fehlschlägt, oder `#CURL_SEEKFUNC_CANTSEEK`, um anzuzeigen, dass libcurl das Problem nach Möglichkeit umgeht. Letzteres kann manchmal durch Lesen von der Eingabe oder ähnlichem erfolgen.

EINGABEN

`seek_callback`
Eingabewert

`userdata` optional: Benutzerdaten, die an die Callback-Funktion übergeben werden sollen

5.217 easy:SetOpt_Service_Name

BEZEICHNUNG

easy:SetOpt_Service_Name – setzt den Namen des Authentifizierungsdienstes

ÜBERSICHT

```
easy:SetOpt_Service_Name(name)
```

BESCHREIBUNG

Übergeben Sie eine Zeichenkette als Parameter an eine Zeichenkette, die den Namen des Dienstes für die Authentifizierungsmechanismen DIGEST-MD5, SPNEGO und Kerberos 5 enthält. Die Standarddienstnamen sind "ftp", "HTTP", "imap", "pop" und "smtp". Mit dieser Option können Sie ihn ändern.

EINGABEN

`name` Eingabewert

5.218 easy:SetOpt_Share

BEZEICHNUNG

easy:SetOpt_Share – gibt den Share-Handle an

ÜBERSICHT

```
easy:SetOpt_Share(share)
```

BESCHREIBUNG

Übergeben Sie einen Share-Handle als Parameter. Der Share-Handle muss durch einen vorherigen Aufruf von `hurl.Share()` erstellt worden sein. Wenn Sie diese Option aktivieren, verwendet dieser Curl-Handle die Daten aus dem Share-Handle, anstatt die Daten für sich zu behalten. Auf diese Weise können Sie mehrere Curl-Handle gemeinsam nutzen. Wenn die Curl gleichzeitig in mehreren Threads verwendet wird, MÜSSEN Sie die Sperrmethoden im Share-Handle verwenden. Siehe `share:SetOpt()` für Details.

Wenn Sie einen Share hinzufügen, der Cookies freigeben soll, verwendet Ihr Easy-Handle diesen Cookie-Cache und aktiviert das Cookie-System. Wenn Sie die Freigabe eines Objekts aufheben, das Cookies verwendet hat (oder zu einem anderen Objekt wechseln, das keine Cookies verwendet), wird das Cookie-System des Easy-Handles deaktiviert.

Daten, für die das Share-Objekt nicht auf Freigabe eingestellt ist, werden auf die übliche Weise behandelt, als ob keine Freigabe verwendet worden wäre.

Setzen Sie diese Option erneut auf Null, um die Verwendung dieses Share-Oobjekts zu beenden.

EINGABEN

`share` Eingabewert

5.219 easy:SetOpt_Socks5_Auth

BEZEICHNUNG

easy:SetOpt_Socks5_Auth – legt die zulässigen Methoden für die SOCKS5-Proxyauthentifizierung fest

ÜBERSICHT

```
easy:SetOpt_Socks5_Auth(bitmask)
```

BESCHREIBUNG

Übergeben Sie einen Wert als Parameter, der auf eine Bitmaske festgelegt ist, um libcurl mitzuteilen, welche Authentifizierungsmethoden für die SOCKS5-Proxyauthentifizierung zulässig sind. Die einzigen unterstützten Flags sind `#CURLAUTH_BASIC`, mit dem eine Benutzer-/Passwortauthentifizierung möglich ist, `#CURLAUTH_GSSAPI`, mit dem eine GSS-API-Authentifizierung möglich ist und `#CURLAUTH_keine`, mit dem keine Authentifizierung möglich ist. Stellen Sie den tatsächlichen Benutzernamen und das Passwort mit der Option `#CURLOPT_PROXYUSERPWD` ein.

EINGABEN

`bitmask` Eingabewert

5.220 easy:SetOpt_Socks5_GSSAPI_NEC

BEZEICHNUNG

easy:SetOpt_Socks5_GSSAPI_NEC – setzt den Socks Proxy gssapi Übertragungsschutz

ÜBERSICHT

```
easy:SetOpt_Socks5_GSSAPI_NEC(nec)
```

BESCHREIBUNG

Übergeben Sie einen Wert von 1 zum Aktivieren oder 0 zum Deaktivieren. Im Rahmen der gssapi-Übertragung wird ein Schutzmodus ausgehandelt. Der RFC1961 sagt in Abschnitt 4.3/4.4, dass er geschützt werden sollte, die NEC-Referenzimplementierung jedoch nicht. Wenn diese Option aktiviert ist, kann die Schutzmodusübertragung ungeschützt ausgetauscht werden.

EINGABEN

`nec` Eingabewert

5.221 easy:SetOpt_Socks5_GSSAPI_Service

BEZEICHNUNG

easy:SetOpt_Socks5_GSSAPI_Service – setzt den SOCKS5-Name des Proxy-Authentifizierungsdienstes (veraltet)

ÜBERSICHT

```
easy:SetOpt_Socks5_GSSAPI_Service(name)
```

BESCHREIBUNG

Veraltet seit 7.49.0. Verwenden Sie stattdessen `#CURLOPT_PROXY_SERVICE_NAME`.

Übergeben Sie eine Zeichenkette als Parameter an eine Zeichenkete, die den Namen des Dienstes enthält. Der Standarddienstname für einen SOCKS5-Server lautet "rcmd". Mit dieser Option können Sie das ändern.

EINGABEN

`name` Eingabewert

5.222 easy:SetOpt_SSH_Auth_Types

BEZEICHNUNG

easy:SetOpt_SSH_Auth_Types – stellt den gewünschten Authentifizierungstypen für SFTP und SCP ein

ÜBERSICHT

```
easy:SetOpt_SSH_Auth_Types(bitmask)
```

BESCHREIBUNG

Übergeben Sie einen Wertesatz an eine Bitmaske, die aus einem oder mehreren `#CURLSSH_AUTH_PUBLICKEY`, `#CURLSSH_AUTH_PASSWORD`, `#CURLSSH_AUTH_HOST`, `#CURLSSH_AUTH_KEYBOARD` und `#CURLSSH_AUTH_AGENT` besteht.

Setzen Sie `#CURLSSH_AUTH_ANY`, damit libcurl einen passenden auswählt. Derzeit hat `#CURLSSH_AUTH_HOST` keine Auswirkung. Wenn `#CURLSSH_AUTH_AGENT` verwendet wird, versucht libcurl, eine Verbindung zu ssh-agent oder pageant herzustellen und lässt den Agenten die Authentifizierung versuchen.

EINGABEN

`bitmask` Eingabewert

5.223 easy:SetOpt_SSH_Host_Public_Key_MD5

BEZEICHNUNG

easy:SetOpt_SSH_Host_Public_Key_MD5 – setzt die Prüfsumme des öffentlichen Schlüssels des SSH-Servers

ÜBERSICHT

```
easy:SetOpt_SSH_Host_Public_Key_MD5(md5)
```

BESCHREIBUNG

Übergeben Sie eine Zeichenkette, die auf eine Zeichenkette mit 32 hexadezimalen Ziffern zeigt. Die Zeichenkette sollte die 128-Bit-MD5-Prüfsumme des öffentlichen Schlüssels des Remote-Hosts sein und libcurl lehnt die Verbindung zum Host ab, sofern die md5-Summen nicht übereinstimmen.

EINGABEN

`md5` Eingabewert

5.224 easy:SetOpt_SSH_KnownHosts

BEZEICHNUNG

easy:SetOpt_SSH_KnownHosts – setzt den Dateiname mit den bekannten SSH-Hosts

ÜBERSICHT

```
easy:SetOpt_SSH_KnownHosts(fname)
```

BESCHREIBUNG

Übergeben Sie eine Zeichenkette mit dem Dateinamen der zu verwendenden Datei known_host. Die Datei known_hosts sollte das von libssh2 unterstützte OpenSSH-Dateiformat verwenden. Wenn diese Datei angegeben ist, akzeptiert libcurl nur Verbindungen mit Hosts, die bekannt sind und in dieser Datei vorhanden sind, mit einem passenden Public Key. Verwenden Sie `#CURLOPT_SSH_KEYFUNCTION`, um die (Fehl-)Übereinstimmung beim Host- und Schlüsselabgleich zu ändern.

EINGABEN

`fname` Eingabewert

5.225 easy:SetOpt_SSH_Private_KeyFile

BEZEICHNUNG

easy:SetOpt_SSH_Private_KeyFile – legt die private Schlüsseldatei für SSH-Authentifizierung fest

ÜBERSICHT

```
easy:SetOpt_SSH_Private_KeyFile(filename)
```

BESCHREIBUNG

Übergeben Sie eine Zeichenkette, die auf einen Dateinamen für Ihren privaten Schlüssel verweist. Wird libcurl nicht verwendet, wird standardmäßig $HOME/.ssh/id_dsa verwendet, wenn die Umgebungsvariable HOME festgelegt ist und nur "id_dsa" im aktuellen Verzeichnis, wenn HOME nicht festgelegt ist.

Wenn die Datei passwortgeschützt ist, legen Sie das Passwort mit `#CURLOPT_KEYPASSWD` fest.

EINGABEN

`filename` Eingabewert

5.226 easy:SetOpt_SSH_Public_KeyFile

BEZEICHNUNG

easy:SetOpt_SSH_Public_KeyFile – legt die öffentliche Schlüsseldatei für die SSH-Authentifizierung fest

ÜBERSICHT

```
easy:SetOpt_SSH_Public_KeyFile(filename)
```

BESCHREIBUNG

Übergeben Sie eine Zeichenkette, die auf einen Dateinamen für Ihren öffentlichen Schlüssel verweist. Wenn sie nicht verwendet wird und die Umgebungsvariable HOME gesetzt ist, ist libcurl standardmäßig auf $HOME/.ssh/id_dsa.pub eingestellt, und nur "id_dsa.pub" im aktuellen Verzeichnis, wenn HOME nicht gesetzt wurde.

Wenn `Nil` (oder eine leere Zeichenkette) übergeben wird, übergibt libcurl keinen öffentlichen Schlüssel an libssh2. Somit wird dann versucht, ihn aus dem privaten Schlüssel zu berechnen. Es ist bekannt, dass dies mit libssh2 1.4.0+ funktioniert, wenn sie mit OpenSSL verknüpft ist.

EINGABEN

`filename` Eingabewert

5.227 easy:SetOpt_SSL_Cipher_List

BEZEICHNUNG

easy:SetOpt_SSL_Cipher_List – gibt die Verschlüsselung an, die für TLS verwendet werden soll

ÜBERSICHT

```
easy:SetOpt_SSL_Cipher_List(list)
```

BESCHREIBUNG

Übergeben Sie eine Zeichenkette, die auf eine Zeichenkette verweist, die die Liste der für die SSL-Verbindung zu verwendende Verschlüsselung enthält. Die Liste muss syntaktisch korrekt sein und aus einer oder mehreren durch Doppelpunkte getrennten Zeichenketten bestehen. Kommas oder Leerzeichen sind ebenfalls akzeptierte Trennzeichen, aber

normalerweise werden Doppelpunkte verwendet. Als Operatoren können ! und - sowie + verwendet werden.

Zu den gültigen Beispielen für OpenSSL- und GnuTLS-Verschlüsselungslisten gehören 'RC4-SHA', SHA1+DES, 'TLSv1' und 'DEFAULT'. Die Standardliste wird normalerweise festgelegt, wenn Sie OpenSSL kompilieren.

Weitere Informationen zu Verschlüsselungslisten finden Sie unter dieser URL: https://curl.haxx.se/docs/ssl-ciphers.html

Gültige Beispiele für Verschlüsselungslisten für NSS sind "rsa_rc4_128_md5", "rsa_aes_128_sha" usw. Mit NSS können Sie keine Verschlüsselung hinzufügen/entfernen. Wenn man diese Option verwendet, werden alle bekannten Verschlüsselungen deaktiviert und nur die übergebenen aktiviert.

Gültige Beispiele für Verschlüsselungslisten für WolfSSL sind ECDHE-RSA-RC4-SHA, 'AES256-SHA:AES256-SHA256', usw.

EINGABEN

`list` Eingabewert

5.228 easy:SetOpt_SSL_Enable_Alpn

BEZEICHNUNG

easy:SetOpt_SSL_Enable_Alpn – aktiviert/deaktiviert ALPN

ÜBERSICHT

```
easy:SetOpt_SSL_Enable_Alpn(npn)
```

BESCHREIBUNG

Übergeben Sie einen Wert als Parameter, 0 oder 1, wobei 1 für die Aktivierung und 0 für die Deaktivierung steht. Diese Option aktiviert/deaktiviert ALPN im SSL-Übergabeverfahren (sofern das libcurl SSL-Backend dies unterstützt), das für die Übertragung von http2 verwendet werden kann.

EINGABEN

`npn` Eingabewert

5.229 easy:SetOpt_SSL_Enable_Npn

BEZEICHNUNG

easy:SetOpt_SSL_Enable_Npn – aktiviert/deaktiviert NPN

ÜBERSICHT

```
easy:SetOpt_SSL_Enable_Npn(npn)
```

BESCHREIBUNG

Übergeben Sie einen Wert als Parameter, 0 oder 1, wobei 1 für die Aktivierung und 0 für die Deaktivierung steht. Diese Option aktiviert/deaktiviert NPN im SSL-Übergabeverfahren (sofern das libcurl SSL-Backend dies unterstützt), für die Übertragung von http2 verwendet werden kann.

EINGABEN

`npn` Eingabewert

5.230 easy:SetOpt_SSL_FalseStart

BEZEICHNUNG

easy:SetOpt_SSL_FalseStart – aktiviert/deaktiviert TLS-Fehlstart

ÜBERSICHT

```
easy:SetOpt_SSL_FalseStart(enable)
```

BESCHREIBUNG

Übergeben Sie einen Wert als Parametersatz 1 zum Aktivieren oder 0 zum Deaktivieren.

Diese Option legt fest, ob libcurl während des TLS-Übergabeverfahren einen Fehlstart verwenden soll. Fehlstart ist ein Modus, in dem ein TLS-Client anfängt Anwendungsdaten zu senden, bevor er die Fertigmeldung des Servers überprüft, wodurch eine Hin- und Herschleife bei einem vollständigen Übergabeverfahren vermieden wird.

EINGABEN

`enable` Eingabewert

5.231 easy:SetOpt_SSL_Options

BEZEICHNUNG

easy:SetOpt_SSL_Options – legt SSL-Verhaltensoptionen fest

ÜBERSICHT

```
easy:SetOpt_SSL_Options(bitmask)
```

BESCHREIBUNG

Übergeben Sie einen Wert mit einer Bitmaske, um libcurl über bestimmte SSL-Verhaltensweisen zu informieren.

`#CURLSSLOPT_ALLOW_BEAST`
: Weist libcurl an, keine Problemumgehungen für Sicherheitslücken in den Protokollen SSL3 und TLS1.0 zu verwenden. Wenn diese Option nicht verwendet wird oder dieses Bit auf 0 gesetzt ist, verwendet die von libcurl verwendete SSL-Ebene möglicherweise eine Problemumgehung für diesen Fehler, obwohl dies bei einigen (älteren) SSL-Implementierungen zu Interoperabilitätsproblemen führen kann. WARNUNG: Wenn Sie dieses Umgehen vermeiden, wird die Sicherheit beeinträchtigt. Wenn Sie diese Option auf 1 setzen, werden Sie genau danach gefragt. Diese Option wird nur für DarwinSSL, NSS und OpenSSL unterstützt.

`#CURLSSLOPT_NO_REVOKE`
: Weist libcurl an, die Zertifikatsperrüberprüfung für die SSL-Backends zu deaktivieren, in denen ein solches Verhalten vorliegt. Derzeit wird diese Option

nur für Kanäle (die native Windows-SSL-Bibliothek) unterstützt, mit Ausnahme der Blacklist "Untrusted Publishers" von Windows, die anscheinend nicht umgangen werden kann. Diese Option bietet möglicherweise eine umfassendere Unterstützung für künftig andere SSL-Backends. https://curl.haxx.se/docs/ssl-compared.html

EINGABEN

`bitmask` Eingabewert

5.232 easy:SetOpt_SSL_SessionID_Cache

BEZEICHNUNG

easy:SetOpt_SSL_SessionID_Cache – aktiviert/deaktiviert die Verwendung des SSL-Sitzungs-ID-Cache

ÜBERSICHT

```
easy:SetOpt_SSL_SessionID_Cache(enabled)
```

BESCHREIBUNG

Übergeben Sie den Wert 0, um die Verwendung von SSL-Sitzungs-ID-Caching durch libcurl zu deaktivieren. Setzen Sie dies auf 1, um es zu aktivieren. Standardmäßig werden alle Übertragungen mit dem aktivierten Cache durchgeführt. Obwohl der Versuch, SSL-Sitzungs-IDs wiederzuverwenden, niemals Schaden anrichten sollte, scheint es in der Natur fehlerhafte SSL-Implementierungen zu geben oder gegeben zu haben, bei denen Sie diese möglicherweise deaktivieren müssen, um erfolgreich zu sein.

EINGABEN

`enabled` Eingabewert

5.233 easy:SetOpt_SSL_VerifyHost

BEZEICHNUNG

easy:SetOpt_SSL_VerifyHost – überprüft den Namen des Zertifikats anhand des Hosts

ÜBERSICHT

```
easy:SetOpt_SSL_VerifyHost(verify)
```

BESCHREIBUNG

Übergeben Sie einen Wert als Parameter, der angibt, was überprüft werden soll.

Diese Option legt fest, ob libcurl überprüft, ob das Serverzertifikat für den Server bestimmt ist, als den es bezeichnet wird.

Bei der Übertragung von TLS- und SSL-Verbindungen sendet der Server ein Zertifikat, das seine Identität angibt.

Wenn `#CURLOPT_SSL_VERIFYHOST` 2 ist, muss dieses Zertifikat angeben, dass der Server der Server ist, zu dem Sie eine Verbindung herstellen möchten oder die Verbindung schlägt fehl. Einfach ausgedrückt bedeutet dies, dass das Zertifikat denselben Namen haben muss wie die URL, mit der Sie arbeiten.

Curl betrachtet den Server als den beabsichtigten Server, wenn das Feld Allgemeiner Name oder das Feld Alternativer Antragstellername im Zertifikat mit dem Hostnamen in der URL übereinstimmt, zu der Sie Curl aufgefordert haben, eine Verbindung herzustellen.

Wenn der Wert von `verify` 1 ist, gibt `easy:SetOpt()` einen Fehler zurück und der Optionswert wird nicht geändert. Es war früher (in 7.28.0 und älter) eine Debug-Option, wird aber nicht mehr unterstützt, da es häufig zu Programmierfehlern kommt. Zukünftige Versionen werden keinen Fehler mehr für 1 zurückgeben und 1 und 2 gleich behandeln.

Wenn der Wert für `verify` 0 ist, ist die Verbindung unabhängig von den Namen im Zertifikat erfolgreich. Verwenden Sie diese Fähigkeit mit Vorsicht!

Der Standardwert für diese Option ist 2.

Diese Option steuert die Überprüfung der beanspruchten Identität des Serverzertifikats. Der Server könnte lügen. Um das Lügen zu überprüfen, siehe `#CURLOPT_SSL_VERIFYPEER`.

EINGABEN

`verify` Eingabewert

5.234 easy:SetOpt_SSL_VerifyPeer

BEZEICHNUNG

easy:SetOpt_SSL_VerifyPeer – überprüft das SSL-Zertifikat des Peers

ÜBERSICHT

```
easy:SetOpt_SSL_VerifyPeer(verify)
```

BESCHREIBUNG

Übergeben Sie einen Wert als Parameter zum Aktivieren oder Deaktivieren.

Diese Option bestimmt, ob Curl die Authentizität des Peer-Zertifikats überprüft. Ein Wert von 1 bedeutet, dass curl das SSL-Zertifikat überprüft, 0 (Null) hingegen bedeutet, dass dies nicht der Fall ist.

Bei der Vermittlung einer TLS- oder SSL-Verbindung sendet der Server ein Zertifikat, das seine Identität angibt. Curl überprüft, ob das Zertifikat authentisch ist, d.h. dass Sie darauf vertrauen können, dass der Server derjenige ist, von dem das Zertifikat sagt, dass er es ist. Diese Vertrauensstellung basiert auf einer Kette digitaler Signaturen, die auf von Ihnen bereitgestellten Zertifizierungsstellen-Zertifikaten (CA-Zertifikaten) basieren. Curl verwendet ein Standardpaket von CA-Zertifikaten (der Pfad dafür wird zum Zeitpunkt der Erstellung festgelegt). Sie können alternative Zertifikate mit der Option `#CURLOPT_CAINFO` oder der Option `#CURLOPT_CAPATH` angeben.

Wenn `#CURLOPT_SSL_VERIFYPEER` aktiviert ist und bei der Überprüfung nicht nachgewiesen werden kann, dass das Zertifikat authentisch ist, schlägt die Verbindung fehl. Wenn die Option Null ist, ist die Überprüfung des Peer-Zertifikats unabhängig davon erfolgreich.

Die Authentifizierung des Zertifikats reicht nicht aus, um den Server zu identifizieren. In der Regel möchten Sie auch sicherstellen, dass der Server der Server ist, mit dem Sie kommunizieren möchten. Verwenden Sie dazu `#CURLOPT_SSL_VERIFYHOST`. Die Überprüfung,

ob der Hostname im Zertifikat für den Hostnamen gültig ist, zu dem Sie eine Verbindung herstellen, erfolgt unabhängig von der Option `#CURLOPT_SSL_VERIFYPEER`.

WARNUNG: Wenn Sie die Überprüfung des Zertifikats deaktivieren, können Unbefugte die Kommunikation direkt ausführen, ohne dass Sie es merken. Durch Deaktivieren der Überprüfung wird die Kommunikation unsicher. Die Verschlüsselung einer Übertragung allein reicht nicht aus, da Sie nicht sicher sein können, ob Sie mit dem richtigen Endpunkt kommunizieren.

HINWEIS: Auch wenn diese Option deaktiviert ist, lädt curl je nach verwendetem TLS-Backend möglicherweise die in `#CURLOPT_CAINFO` angegebene Zertifikatdatei. Curl-Standardeinstellungen in einigen Distributionen können eine recht große Datei für `#CURLOPT_CAINFO` verwenden, so dass das Laden der Datei sehr aufwendig sein kann, insbesondere bei vielen Verbindungen. Daher möchten Sie in einigen Situationen die Überprüfung möglicherweise vollständig deaktivieren, um Ressourcen zu sparen, indem Sie `#CURLOPT_CAINFO` auf Null setzen. Beachten Sie jedoch auch die obige Warnung!

EINGABEN

`verify` Eingabewert

5.235 easy:SetOpt_SSL_VerifyStatus

BEZEICHNUNG

easy:SetOpt_SSL_VerifyStatus – überprüft den Status des Zertifikats

ÜBERSICHT

```
easy:SetOpt_SSL_VerifyStatus(verify)
```

BESCHREIBUNG

Übergeben Sie einen Wert als Parametersatz, 1 zum Aktivieren oder 0 zum Deaktivieren.

Diese Option legt fest, ob libcurl den Status des Serverzertifikats mithilfe der TLS-Erweiterung "Certificate Status Request" (auch als OCSP-Stapelung bezeichnet) überprüft.

Beachten Sie, dass die Überprüfung fehlschlägt, wenn diese Option aktiviert ist, der Server jedoch die TLS-Erweiterung nicht unterstützt.

EINGABEN

`verify` Eingabewert

5.236 easy:SetOpt_SSLCert

BEZEICHNUNG

easy:SetOpt_SSLCert – stellt das SSL-Client-Zertifikat ein

ÜBERSICHT

```
easy:SetOpt_SSLCert(cert)
```

BESCHREIBUNG

Übergeben Sie eine Zeichenkette als Parameter. Die Zeichenkette sollte der Dateiname Ihres Client-Zertifikats sein. Das Standardformat ist "P12" für Sichere Übertragungen

sowie "PEM" für andere Systeme und kann mit `#CURLOPT_SSLCERTTYPE` geändert werden.

Bei NSS oder Sichere Übertragungen kann dies auch der Kurzname des Zertifikats sein, mit dem Sie sich authentifizieren möchten, wie er in der Sicherheitsdatenbank angegeben ist. Wenn Sie eine Datei aus dem aktuellen Verzeichnis verwenden möchten, müssen Sie ihr das Präfix "./" voranstellen, um Verwechslungen mit einem Kurznamen zu vermeiden.

(Nur Kanal) Client-Zertifikate müssen durch einen Pfadausdruck zu einem Zertifikatspeicher angegeben werden. (Laden von PFX wird nicht unterstützt; Sie können es zuerst in den Speicher importieren). Sie können "<Standort>\<Speichername>\<Speicherort>" verwenden, um auf ein Zertifikat im Systemzertifikatsspeicher zu verweisen, z.B. "CurrentUser\MY\934a7ac6f8a5d579285a74fa61e19f23ddfe8d7a". Der Speicherort ist normalerweise eine SHA-1-Hex-Zeichenkette, die Sie in den Zertifikatdetails sehen können. Folgende Speicherorte werden unterstützt: CurrentUser, LocalMachine, CurrentService, Services, CurrentUserGroupPolicy, LocalMachineGroupPolicy, LocalMachineEnterprise.

Wenn Sie ein Client-Zertifikat verwenden, müssen Sie höchstwahrscheinlich auch einen privaten Schlüssel mit `#CURLOPT_SSLKEY` bereitstellen.

EINGABEN

`cert` Eingabewert

5.237 easy:SetOpt_SSLCertType

BEZEICHNUNG

easy:SetOpt_SSLCertType – gibt den Typ des Client-SSL-Zertifikats an

ÜBERSICHT

```
easy:SetOpt_SSLCertType(type)
```

BESCHREIBUNG

Übergeben Sie eine Zeichenkette als Parameter. Die Zeichenkette sollte das Format Ihres Zertifikats haben. Unterstützte Formate sind "PEM" und "DER", außer bei Sicherer Übertragung. OpenSSL (Versionen 0.9.3 und höher) und Sicheren Übertragungen (iOS 5 oder höher oder OS X 10.7 oder höher) unterstützen auch "P12" für PKCS#12-codierte Dateien.

EINGABEN

`type` Eingabewert

5.238 easy:SetOpt_SSLEngine

BEZEICHNUNG

easy:SetOpt_SSLEngine – setzt die SSL System ID

ÜBERSICHT

```
easy:SetOpt_SSLEngine(id)
```

BESCHREIBUNG

Übergeben Sie eine Zeichenkette als Parameter. Sie wird als ID für die Crypto-Systeme verwendet, die Sie für Ihren privaten Schlüssel verwenden möchten.

EINGABEN

`id` Eingabewert

5.239 easy:SetOpt_SSLEngine_Default

BEZEICHNUNG

easy:SetOpt_SSLEngine_Default – legt das SSL-System als Standard fest

ÜBERSICHT

```
easy:SetOpt_SSLEngine_Default(val)
```

BESCHREIBUNG

Übergeben Sie den Wert 1, um das bereits angegebene Crypto-System als Standard für (asymmetrische) Crypto-Operationen festzulegen.

Diese Option hat keine Auswirkung, es sei denn, sie wurde nach `#CURLOPT_SSLENGINE` festgelegt.

EINGABEN

`val` Eingabewert

5.240 easy:SetOpt_SSLKey

BEZEICHNUNG

easy:SetOpt_SSLKey – gibt eine private Schlüsseldatei für TLS- und SSL-Client-Zertifikate an

ÜBERSICHT

```
easy:SetOpt_SSLKey(keyfile)
```

BESCHREIBUNG

Übergeben Sie eine Zeichenkette als Parameter. Die Zeichenkette sollte der Dateiname Ihres privaten Schlüssels sein. Das Standardformat ist `"PEM"` und kann mit `#CURLOPT_SSLKEYTYPE` geändert werden.

(Nur iOS und Mac OS X) Diese Option wird ignoriert, wenn Curl für Sichere Übertragung erstellt wurde. Sichere Übertragung erwartet, dass der private Schlüssel bereits im Keychain oder in der PKCS#12-Datei vorhanden ist, die das Zertifikat enthält.

EINGABEN

`keyfile` Eingabewert

5.241 easy:SetOpt_SSLKeyType

BEZEICHNUNG

easy:SetOpt_SSLKeyType – setzt den Typ der privaten Schlüsseldatei

ÜBERSICHT

```
easy:SetOpt_SSLKeyType(type)
```

BESCHREIBUNG

Übergeben Sie eine Zeichenkette als Parameter. Die Zeichenkette sollte das Format Ihres privaten Schlüssels haben. Unterstützte Formate sind "PEM", "DER" und "ENG".

Das Format "ENG" ermöglicht es Ihnen, den privaten Schlüssel von einem Crypto-System zu laden. In diesem Fall wird `#CURLOPT_SSLKEY` die an das System übergebene Passwort verwendet. Sie müssen das Crypto System mit `#CURLOPT_SSLENGINE` einstellen. Die Schlüsseldatei im "DER"-Format funktioniert derzeit aufgrund eines Fehlers in OpenSSL nicht.

EINGABEN

`type` Eingabewert

5.242 easy:SetOpt_SSLVersion

BEZEICHNUNG

easy:SetOpt_SSLVersion – stellt die bevorzugte TLS/SSL-Version ein

ÜBERSICHT

```
easy:SetOpt_SSLVersion(version)
```

BESCHREIBUNG

Übergeben Sie einen Wert als Parameter, um zu steuern, welcher Versionsbereich von SSL/TLS-Versionen verwendet werden soll.

Die SSL- und TLS-Versionen haben sich in der Regel aus der unsichersten Version entwickelt, um in dieser Reihenfolge des Verlaufs immer sicherer zu werden: SSL v2, SSL v3, TLS v1.0, TLS v1.1, TLS v1.2 und die neueste TLS v1.3.

Verwenden Sie dazu eine der folgenden verfügbaren Optionen:

`#CURL_SSLVERSION_DEFAULT`
: Der standardmäßig zulässige Versionsbereich. Die minimale akzeptable Version ist standardmäßig TLS v1.0 seit 7.39.0 (es sei denn, die TLS-Bibliothek hat eine strengere Regel).

`#CURL_SSLVERSION_TLSv1`
: TLS v1.0 oder höher

`#CURL_SSLVERSION_SSLv2`
: SSL v2 (aber nicht SSLv3)

`#CURL_SSLVERSION_SSLv3`
: SSL v3 (aber nicht SSLv2)

`#CURL_SSLVERSION_TLSv1_0`
: TLS v1.0 oder höher (Hinzugefügt in 7.34.0)

`#CURL_SSLVERSION_TLSv1_1`
: TLS v1.1 oder höher (Hinzugefügt in 7.34.0)

`#CURL_SSLVERSION_TLSv1_2`
: TLS v1.2 oder höher (Hinzugefügt in 7.34.0)

`#CURL_SSLVERSION_TLSv1_3`
: TLS v1.3 oder höher (Hinzugefügt in 7.52.0)

Die maximale TLS-Version kann mit `einem` der untenstehenden Makros `#CURL_SSLVERSION_MAX_` eingestellt werden. Es ist auch möglich, eines der `#CURL_SSLVERSION_` Makros mit einem der `#CURL_SSLVERSION_MAX_` Makros mittels OR zu verwenden. Die MAX-Makros unterstützen WolfSSL nicht.

`#CURL_SSLVERSION_MAX_DEFAULT`
: Das Flag definiert die von libcurl maximal unterstützte TLS-Version, oder es wird der Standardwert aus der SSL-Bibliothek verwendet. libcurl verwendet ein sinnvolles Standardmaximum, das TLS v1.2 bis vor 7.61.0 war und seitdem TLS v1.3 ist - vorausgesetzt, die TLS-Bibliothek unterstützt es. (Hinzugefügt in 7.54.0)

`#CURL_SSLVERSION_MAX_TLSv1_0`
: Das Flag definiert die maximal unterstützte TLS-Version als TLS v1.0. (Hinzugefügt in 7.54.0)

`#CURL_SSLVERSION_MAX_TLSv1_1`
: Das Flag definiert die maximal unterstützte TLS-Version als TLS v1.1. (Hinzugefügt in 7.54.0)

`#CURL_SSLVERSION_MAX_TLSv1_2`
: Das Flag definiert die maximal unterstützte TLS-Version als TLS v1.2. (Hinzugefügt in 7.54.0)

`#CURL_SSLVERSION_MAX_TLSv1_3`
: Das Flag definiert die maximal unterstützte TLS-Version als TLS v1.3. (Hinzugefügt in 7.54.0)

EINGABEN

`version` Eingabewert

5.243 easy:SetOpt_Stream_Depends

BEZEICHNUNG

easy:SetOpt_Stream_Depends – stellt den Stream ein, von dem diese Übertragung abhängt

ÜBERSICHT

```
easy:SetOpt_Stream_Depends(dephandle)
```

BESCHREIBUNG

Übergeben Sie in `dephandle` ein Curl-Handle, um den Stream innerhalb derselben Verbindung zu identifizieren, von der dieser Stream abhängig ist. Diese Option löscht das

exklusive Bit und schließt sich gegenseitig für die Option `#CURLOPT_STREAM_DEPENDS_E` aus.

In der Spezifikation heißt es: "Das Einbeziehen einer Abhängigkeit drückt eine Präferenz aus, Ressourcen dem identifizierten Stream zuzuweisen, anstelle des abhängigen Stream."

Diese Option kann während der Übertragung eingestellt werden.

`dephandle` darf nicht mit `handle` identisch sein, da dieser Befehl einen Fehler zurückgibt. Dies muss ein weiterer Easy-Handle sein und es muss sich auch um einen Handle einer Übertragung handeln, die über dieselbe HTTP/2-Verbindung gesendet wird, damit diese Option tatsächlich wirksam wird.

EINGABEN

`dephandle`
: Eingabewert

5.244 easy:SetOpt_Stream_Depends_e

BEZEICHNUNG

easy:SetOpt_Stream_Depends_e – stellt den Stream ein, von dem diese Übertragung ausschließlich abhängt

ÜBERSICHT

```
easy:SetOpt_Stream_Depends_e(dephandle)
```

BESCHREIBUNG

Übergeben Sie in `dephandle` einen Curl-Handle, um den Stream innerhalb derselben Verbindung zu identifizieren, von der dieser Stream ausschließlich abhängt. Das heißt, er hängt davon ab und setzt das Exclusive-Bit.

In der Spezifikation heißt es: "Das Einbeziehen einer Abhängigkeit drückt eine Präferenz aus, Ressourcen dem identifizierten Stream zuzuweisen, anstelle des abhängigen Stream."

Durch das Festlegen einer Abhängigkeit mit dem Exklusiv-Flag für einen repriorisierten Stream werden alle Abhängigkeiten des neuen übergeordneten Streams vom repriorisierten Stream abhängig.

Diese Option kann während der Übertragung eingestellt werden.

`dephandle` darf nicht mit `handle` identisch sein, da dieser Befehl einen Fehler zurückgibt. Dies muss ein weiterer Easy-Handle sein und es muss sich auch um ein Handle einer Übertragung handeln, die über dieselbe HTTP/2-Verbindung gesendet wird, damit diese Option tatsächlich wirksam wird.

EINGABEN

`dephandle`
: Eingabewert

5.245 easy:SetOpt_Stream_Weight

BEZEICHNUNG

easy:SetOpt_Stream_Weight – setzt die Gewichtung des numerischen Datenstroms

ÜBERSICHT

```
easy:SetOpt_Stream_Weight(weight)
```

BESCHREIBUNG

Stellen Sie den Parameter `weight` auf eine Zahl zwischen 1 und 256 ein.

Bei Verwendung von HTTP/2 wird mit dieser Option die individuelle Gewichtung für diesen bestimmten Datenstrom festgelegt, die vom Easy `handle` verwendet wird. Das Festlegen und Verwenden von Gewichtungen ist nur sinnvoll und kann nur verwendet werden, wenn mehrere Datenströme über die gleichen Verbindungen ausgeführt werden. Dies bedeutet, dass Sie `#CURLMOPT_PIPELINING` verwenden.

Diese Option kann während der Übertragung festgelegt werden und bewirkt, dass die aktualisierten Gewichtungsinformationen beim nächsten Senden eines HTTP/2-Frames an den Server gesendet werden.

Weitere Informationen zum Protokoll finden Sie in Abschnitt 5.3 von RFC 7540: https://httpwg.github.io/specs/rfc7540.html#StreamPriority

Datenströme mit demselben übergeordneten Element sollten entsprechend ihrer Gewichtung Ressourcen zugewiesen bekommen. Wenn also zwei Datenströme aktiv sind, Datenstrom A mit Gewichtung 16 und Datenstrom B mit Gewichtung 32, erhält Datenstrom B zwei Drittel (32/48) der verfügbaren Bandbreite (vorausgesetzt, der Server kann die Daten für beide Datenströme gleichermaßen senden).

EINGABEN

`weight` Eingabewert

5.246 easy:SetOpt_Suppress_Connect_Headers

BEZEICHNUNG

easy:SetOpt_Suppress_Connect_Headers – unterdrückt Proxy-CONNECT-Antwort-Header von Benutzer-Callbacks

ÜBERSICHT

```
easy:SetOpt_Suppress_Connect_Headers(onoff)
```

BESCHREIBUNG

Unterdrücken Sie bei Verwendung von `#CURLOPT_HTTPPROXYTUNNEL` und einer CONNECT-Anforderung den Proxy-CONNECT-Antwort-Header aus den Benutzer-Callback-Funktionen `#CURLOPT_HEADERFUNCTION` und `#CURLOPT_WRITEFUNCTION`.

Proxy CONNECT-Antwort-Header können die Header-Verarbeitung erschweren, da es sich im Wesentlichen um einen separaten Satz von Header handelt. Sie können diese Option aktivieren, um diesen Header zu unterdrücken.

Nehmen wir beispielsweise an, dass eine HTTPS-URL über CONNECT abgerufen werden soll. Bei Erfolg würde es normalerweise zwei Sätze von Header geben und jeder Header würde an die Header-Funktion und/oder die Schreibfunktion gesendet. Die Daten für die Callbacks sehen folgendermaßen aus:

```
HTTP / 1.1 200 Verbindung hergestellt
{Header} ...
```

```
HTTP/1.1 200 OK
Inhaltstyp: Anwendung/json
{Header} ...

{body}...
```

Wenn Sie diese Option aktivieren, werden die CONNECT-Antwort-Header jedoch unterdrückt, sodass die an die Callback gesendeten Daten folgendermaßen aussehen:

```
HTTP/1.1 200 OK
Inhaltstyp: Anwendung/json
{Header} ...

{body}...
```

EINGABEN

`onoff` Eingabewert

5.247 easy:SetOpt_TCP_FastOpen

BEZEICHNUNG

easy:SetOpt_TCP_FastOpen – aktiviert/deaktiviert TCP Fast Open

ÜBERSICHT

```
easy:SetOpt_TCP_FastOpen(enable)
```

BESCHREIBUNG

Übergeben Sie einen Wert als Parametersatz 1 zum Aktivieren oder 0 zum Deaktivieren.

TCP Fast Open (RFC7413) ist ein Mechanismus, mit dem Daten in den SYN- und SYN-ACK-Paketen übertragen und vom empfangenden Ende während des anfänglichen Verbindungs-Handshakes verbraucht werden können, wodurch bis zu einem vollständigen Round-Trip-Time (RTT) eingespart wird.

EINGABEN

`enable` Eingabewert

5.248 easy:SetOpt_TCP_KeepAlive

BEZEICHNUNG

easy:SetOpt_TCP_KeepAlive – aktiviert die Tests TCP-Keep-Alive

ÜBERSICHT

```
easy:SetOpt_TCP_KeepAlive(probe)
```

BESCHREIBUNG

Übergeben Sie einen Wert. Wenn dieser Wert auf 1 gesetzt ist, werden TCP-Keep-Alive-Tests gesendet. Die Verzögerung und Frequenz dieser Tests kann mit den Optionen `#CURLOPT_TCP_KEEPIDLE` und `#CURLOPT_TCP_KEEPINTVL` gesteuert werden, sofern das

Betriebssystem sie unterstützt. Setzen Sie den Wert auf 0 (Standardverhalten), um die Keep-Alive-Tests zu deaktivieren.

EINGABEN

`probe` Eingabewert

5.249 easy:SetOpt_TCP_KeepIdle

BEZEICHNUNG

easy:SetOpt_TCP_KeepIdle – setzt die TCP-Keep-Alive Leerlaufzeit

ÜBERSICHT

```
easy:SetOpt_TCP_KeepIdle(delay)
```

BESCHREIBUNG

Übergebe einen Wert. Legt die Verzögerung in Sekunden fest, die das Betriebssystem wartet, während die Verbindung inaktiv ist, bevor Keep-Alive-Tests gesendet werden. Nicht alle Betriebssysteme unterstützen diese Option.

EINGABEN

`delay` Eingabewert

5.250 easy:SetOpt_TCP_KeepIntvl

BEZEICHNUNG

easy:SetOpt_TCP_KeepIntvl – legt den TCP-Keep-Alive-Intervall fest

ÜBERSICHT

```
easy:SetOpt_TCP_KeepIntvl(interval)
```

BESCHREIBUNG

Übergibt einen Wert. Legt den Intervall in Sekunden fest, in dem das Betriebssystem zwischen dem Senden von Keep-Alive-Tests wartet. Nicht alle Betriebssysteme unterstützen diese Option. (Hinzugefügt in 7.25.0)

EINGABEN

`interval` Eingabewert

5.251 easy:SetOpt_TCP_NoDelay

BEZEICHNUNG

easy:SetOpt_TCP_NoDelay – aktiviert/deaktiviert die Option TCP_NODELAY

ÜBERSICHT

```
easy:SetOpt_TCP_NoDelay(nodelay)
```

BESCHREIBUNG

Übergeben Sie einen Wert, der angibt, ob die Option TCP_NODELAY gesetzt oder gelöscht werden soll (1 = gesetzt, 0 = gelöscht). Die Option ist standardmäßig aktiviert. Dies hat nach dem Verbindungsaufbau keine Auswirkung.

Wenn Sie diese Option auf 1 setzen, wird der Nagle-Algorithmus von TCP für diese Verbindung deaktiviert. Mit diesem Algorithmus soll versucht werden, die Anzahl der kleinen Pakete im Netzwerk zu minimieren (wobei "kleine Pakete" TCP-Segmente bedeuten, die kleiner sind als die maximale Segmentgröße Maximum Segment Size (MSS) für das Netzwerk).

Das Maximieren der pro TCP-Segment gesendeten Datenmenge ist gut, da dies den Aufwand des Sendens amortisiert. In einigen Fällen müssen jedoch möglicherweise kleine Segmente unverzüglich gesendet werden. Dies ist weniger effizient als das gleichzeitige Senden größerer Datenmengen und kann zu einer Überlastung des Netzwerks führen.

EINGABEN

`nodelay` Eingabewert

5.252 easy:SetOpt_TelnetOptions

BEZEICHNUNG

easy:SetOpt_TelnetOptions – setzt die benutzerdefinierten Telnet-Optionen

ÜBERSICHT

```
easy:SetOpt_TelnetOptions(cmds)
```

BESCHREIBUNG

Stellen Sie eine Tabelle mit einer Liste von Variablen bereit, die an die Telnet-Übertragungen übergeben werden sollen. Die Variablen sollten im Format <option=value> vorliegen. libcurl unterstützt die Optionen 'TTYPE', 'XDISPLOC' und 'NEW_ENV'. Einzelheiten finden Sie im TELNET-Standard.

EINGABEN

`cmds` Eingabewert

5.253 easy:SetOpt_TFTP_BlkSize

BEZEICHNUNG

easy:SetOpt_TFTP_BlkSize – setzt die TFTP-Blockgröße

ÜBERSICHT

```
easy:SetOpt_TFTP_BlkSize(blocksize)
```

BESCHREIBUNG

Geben Sie die Blockgröße an, die für die TFTP-Datenübertragung verwendet werden soll. Der gültige Bereich gemäß RFC2348 liegt zwischen 8 und 65464 Byte. Der Standardwert von 512 Byte wird verwendet, wenn diese Option nicht angegeben wird. Die angegebene Blockgröße wird nur verwendet, wenn der Remote-Server dies nicht unterstützt.

Wenn der Server keine Optionsbestätigung oder eine Optionsbestätigung ohne Blockgröße zurückgibt, wird der Standardwert von 512 Byte verwendet.

EINGABEN

`blocksize`
: Eingabewert

5.254 easy:SetOpt_TFTP_No_Options

BEZEICHNUNG

easy:SetOpt_TFTP_No_Options – sendet keine TFTP-Optionsanforderungen

ÜBERSICHT

```
easy:SetOpt_TFTP_No_Options(onoff)
```

BESCHREIBUNG

Setzen Sie `onoff` auf 1, um alle in RFC2347, RFC2348 und RFC2349 definierten TFTP-Optionen von Lese- und Schreibanforderungen (RRQs/WRQs) auszuschließen.

Diese Option verbessert die Interoperabilität mit einigen Legacy-Servern, die TFTP-Optionen nicht bestätigen oder nicht ordnungsgemäß implementieren. Wenn diese Option verwendet wird, wird `#CURLOPT_TFTP_BLKSIZE` ignoriert.

EINGABEN

`onoff` Eingabewert

5.255 easy:SetOpt_TimeCondition

BEZEICHNUNG

easy:SetOpt_TimeCondition – wählt die Bedingung für eine Zeitanfrage aus

ÜBERSICHT

```
easy:SetOpt_TimeCondition(cond)
```

BESCHREIBUNG

Übergeben Sie einen Wert als Parameter. Dies definiert, wie der Zeitwert `#CURLOPT_TIMEVALUE` behandelt wird. Sie können diesen Parameter auf `#CURL_TIMECOND_IFMODSINCE` oder `#CURL_TIMECOND_IFUNMODSINCE` setzen.

Der letzte Änderungszeitpunkt einer Datei ist nicht immer bekannt und in solchen Fällen hat dieser Befehl keine Auswirkung, selbst wenn die angegebene Zeitbedingung nicht erfüllt worden wäre. Der Befehl `easy:GetInfo()` mit der Option `#CURLINFO_CONDITION_UNMET` kann nach einer Übertragung verwendet werden, um zu erfahren, ob ein erfolgreicher "transfer" von null Byte aufgrund dieser nicht übereinstimmenden Bedingung erfolgte.

EINGABEN

`cond` Eingabewert

5.256 easy:SetOpt_Timeout

BEZEICHNUNG

easy:SetOpt_Timeout – legt die maximale Zeit in Sekunden fest, die die Anforderung dauern darf

ÜBERSICHT

```
easy:SetOpt_Timeout(timeout)
```

BESCHREIBUNG

Übergeben Sie einen Wert als Parameter in `timeout`. Das ist die maximale Zeit in Sekunden, die Sie für die libcurl-Übertragung benötigen. Normalerweise können Namensnachforschungen eine beträchtliche Zeit in Anspruch nehmen und den Vorgang auf weniger als ein paar Minuten begrenzen. Diese Option kann dazu führen, dass libcurl das SIGALRM-Signal für Zeitüberschreitungen von Systemaufrufen verwendet.

In Unix-ähnlichen Systemen kann dies dazu führen, dass Signale verwendet werden, sofern nicht `#CURLOPT_NOSIGNAL` gesetzt ist.

Wenn sowohl `#CURLOPT_TIMEOUT` als auch `#CURLOPT_TIMEOUT_MS` festgelegt sind, wird der zuletzt festgelegte Wert verwendet.

Da dies die Dauer einer Anforderung stark einschränkt, ist ihre Verwendung in dynamischen Anwendungsfällen mit variierenden Übertragungszeiten eingeschränkt. In diesem Fall sollten Sie `#CURLOPT_LOW_SPEED_LIMIT`, `#CURLOPT_LOW_SPEED_TIME` oder `#CURLOPT_PROGRESSFUNCTION` verwenden, um Ihre eigene Zeitüberschreitungs-Logik zu implementieren.

EINGABEN

`timeout` Eingabewert

5.257 easy:SetOpt_Timeout_MS

BEZEICHNUNG

easy:SetOpt_Timeout_MS – legt die maximale Zeit in Millisekunden fest, die die Anforderung dauern darf

ÜBERSICHT

```
easy:SetOpt_Timeout_MS(timeout)
```

BESCHREIBUNG

Übergeben Sie einen Wert als Parameter in `timeout`. Das ist die maximale Zeit in Millisekunden, die Sie für die libcurl-Übertragung benötigen. Normalerweise können Namensnachforschungen eine beträchtliche Zeit in Anspruch nehmen und den Vorgang auf weniger als ein paar Minuten begrenzen. Diese Option kann dazu führen, dass libcurl das SIGALRM-Signal für Zeitüberschreitungen von Systemaufrufen verwendet.

Wenn libcurl für die Verwendung des Standard-Systemnamensauflösers erstellt wurde, verwendet dieser Teil der Übertragung für Zeitüberschreitungen eine Auflösung von einer vollen Sekunde, wobei eine Zeitüberschreitung von mindestens einer Sekunde zulässig ist.

In Unix-ähnlichen Systemen kann dies dazu führen, dass Signale verwendet werden, sofern nicht `#CURLOPT_NOSIGNAL` gesetzt ist.

Wenn sowohl `#CURLOPT_TIMEOUT` als auch `#CURLOPT_TIMEOUT_MS` festgelegt sind, wird der zuletzt festgelegte Wert verwendet.

Da dies die Dauer einer Anforderung stark einschränkt, ist ihre Verwendung in dynamischen Anwendungsfällen mit variierenden Übertragungszeiten eingeschränkt. In diesem Fall sollten Sie `#CURLOPT_LOW_SPEED_LIMIT`, `#CURLOPT_LOW_SPEED_TIME` oder `#CURLOPT_PROGRESSFUNCTION` verwenden, um Ihre eigene Zeitüberschreitungs-Logik zu implementieren.

EINGABEN

`timeout` Eingabewert

5.258 easy:SetOpt_TimeValue

BEZEICHNUNG

easy:SetOpt_TimeValue – setzt den Zeitwert für bedingtes Verhalten

ÜBERSICHT

```
easy:SetOpt_TimeValue(val)
```

BESCHREIBUNG

Übergeben Sie einen Wert im Parameter `val`. Dies sollte die Zeit sein, die seit dem 1. Januar 1970 als Sekunden gezählt wurde und die Zeit in einer mit `#CURLOPT_TIMECONDITION` angegebenen Bedingung verwendet wird.

Auf Systemen mit 'langen' 32-Bit-Variablen kann diese Option keine Daten nach dem Jahr 2038 festlegen. Berücksichtigen Sie stattdessen `#CURLOPT_TIMEVALUE_LARGE`.

EINGABEN

`val` Eingabewert

5.259 easy:SetOpt_TLSAuth_Password

BEZEICHNUNG

easy:SetOpt_TLSAuth_Password – setzt das Passwort für die TLS-Authentifizierung

ÜBERSICHT

```
easy:SetOpt_TLSAuth_Password(pwd)
```

BESCHREIBUNG

Übergeben Sie eine Zeichenkette als Parameter mit dem Passwort, das für die mit der Option `#CURLOPT_TLSAUTH_TYPE` angegebene TLS-Authentifizierungsmethode verwendet werden soll. Erfordert, dass auch die Option `#CURLOPT_TLSAUTH_USERNAME` festgelegt wird.

EINGABEN

`pwd` Eingabewert

5.260 easy:SetOpt_TLSAuth_Type

BEZEICHNUNG

easy:SetOpt_TLSAuth_Type – legt die TLS-Authentifizierungsmethoden fest

ÜBERSICHT

```
easy:SetOpt_TLSAuth_Type(type)
```

BESCHREIBUNG

Übergeben Sie eine Zeichenkette als Parameter. Die Zeichenkette sollte die Methode der TLS-Authentifizierung sein. Unterstützte Methode ist "SRP".

`SRP` TLS-SRP-Authentifizierung. Die sichere Remote-Passwortauthentifizierung für TLS ist in RFC5054 definiert und bietet gegenseitige Authentifizierung, wenn beide Seiten ein gemeinsames Passwort haben. Um TLS-SRP zu verwenden, müssen Sie auch die Optionen `#CURLOPT_TLSAUTH_USERNAME` und `#CURLOPT_TLSAUTH_PASSWORD` festlegen.

EINGABEN

`type` Eingabewert

5.261 easy:SetOpt_TLSAuth_UserName

BEZEICHNUNG

easy:SetOpt_TLSAuth_UserName – legt den Benutzernamen fest, der für die TLS-Authentifizierung verwendet wird

ÜBERSICHT

```
easy:SetOpt_TLSAuth_UserName(user)
```

BESCHREIBUNG

Übergeben Sie eine Zeichenkette als Parameter mit dem Benutzernamen, der für die mit der Option `#CURLOPT_TLSAUTH_TYPE` angegebene TLS-Authentifizierungsmethode verwendet werden soll. Erfordert, dass auch die Option `#CURLOPT_TLSAUTH_PASSWORD` festgelegt wird.

EINGABEN

`user` Eingabewert

5.262 easy:SetOpt_Transfer_Encoding

BEZEICHNUNG

easy:SetOpt_Transfer_Encoding – fordert die Übertragungscodierung an

ÜBERSICHT

```
easy:SetOpt_Transfer_Encoding(enable)
```

BESCHREIBUNG

Übergeben Sie im Parameter `enable` einen Wert von 1 zum Aktivieren oder 0 zum Deaktivieren.

Fügt der ausgehenden HTTP-Anforderung eine Anforderung für die komprimierte Übertragungscodierung hinzu. Wenn der Server dies unterstützt und dies wünscht, kann er mit der HTTP-Antwort antworten, die mit einer komprimierten Übertragungscodierung gesendet wird, die von libcurl beim Empfang automatisch dekomprimiert wird.

Die Übertragungscodierung unterscheidet sich geringfügig von der Inhaltscodierung, die Sie mit `#CURLOPT_ACCEPT_ENCODING` anfordern, da eine Übertragungscodierung ausschließlich für die Übertragung vorgesehen ist und daher vor dem Eintreffen der Daten im Client decodiert werden MUSS. Üblicherweise wird die Übertragungscodierung von HTTP-Clients und HTTP-Servern weitaus weniger verwendet und unterstützt.

EINGABEN

`enable` Eingabewert

5.263 easy:SetOpt_TransferText

BEZEICHNUNG

easy:SetOpt_TransferText – fordet eine textbasierte Übertragung für FTP an

ÜBERSICHT

```
easy:SetOpt_TransferText(text)
```

BESCHREIBUNG

Ein auf 1 gesetzter Parameter `text` weist die Bibliothek an, anstelle der Standard-Binärübertragung den ASCII-Modus für FTP-Übertragungen zu verwenden. Für Win32-Systeme wird die Standardausgabe nicht auf den Binärmodus gesetzt. Diese Option kann beim Übertragen von Textdaten zwischen Systemen mit unterschiedlichen Ansichten für bestimmte Zeichen, z.B. Zeilenumbrüche oder ähnlichem, verwendet werden.

libcurl führt bei ASCII-Übertragungen über FTP keine vollständige ASCII-Konvertierung durch. Dies ist eine bekannte Einschränkung/ein bekannter Fehler, den niemand behoben hat. libcurl setzt den Modus einfach auf ASCII und führt eine Standardübertragung durch.

EINGABEN

`text` Eingabewert

5.264 easy:SetOpt_Unix_Socket_Path

BEZEICHNUNG

easy:SetOpt_Unix_Socket_Path – setzt den Unix Domain Socket

ÜBERSICHT

```
easy:SetOpt_Unix_Socket_Path(path)
```

BESCHREIBUNG

Aktiviert die Verwendung von Unix-Domain-Sockets als Verbindungsendpunkt und setzt den Pfad auf `path`. Wenn `path` gleich Null ist, sind Unix-Domain-Sockets deaktiviert.

Eine leere Zeichenkette führt irgendwann zu einem Fehler. Die Verwendung von Unix-Domain-Sockets wird nicht deaktiviert.

Wenn diese Option aktiviert ist, stellt curl eine Verbindung zum Unix-Domain-Socket her, anstatt eine TCP-Verbindung zu einem Host herzustellen. Da keine TCP-Verbindung erstellt wird, muss curl den DNS-Hostnamen in der URL nicht auflösen.

Die maximale Pfadlänge unter Cygwin, Linux und Solaris beträgt 107. Auf anderen Plattformen ist sie möglicherweise noch geringer.

Proxy- und TCP-Optionen wie

`#CURLOPT_TCP_NODELAY`
: werden nicht unterstützt. Proxy-Optionen wie

`#CURLOPT_PROXY`
: haben auch keine Auswirkung, da diese TCP-orientiert sind und es nicht möglich ist, einen Proxyserver aufzufordern, eine Verbindung zu einem bestimmten Unix-Domain-Socket herzustellen.

EINGABEN

`path` Eingabewert

5.265 easy:SetOpt_Unrestricted_Auth

BEZEICHNUNG

easy:SetOpt_Unrestricted_Auth – sendet Authentifizierungsdaten auch an andere Hosts

ÜBERSICHT

```
easy:SetOpt_Unrestricted_Auth(goahead)
```

BESCHREIBUNG

Setzen Sie den Parameter `gohead` auf 1, damit libcurl weiterhin Authentifizierungsdaten (Benutzer+Passwort) sendet, wenn nachfolgende Speicherorte verfolgt werden, auch wenn der Hostname geändert wird. Diese Option ist nur beim Festlegen von `#CURLOPT_FOLLOWLOCATION` von Bedeutung.

Standardmäßig sendet libcurl die angegebenen Authentifizierungsdaten nur an den ursprünglichen Hostnamen, der in der ursprünglichen URL angegeben ist, um zu verhindern, dass Benutzername und Passwort an andere Seiten weitergegeben werden.

EINGABEN

`goahead` Eingabewert

5.266 easy:SetOpt_Upload

BEZEICHNUNG

easy:SetOpt_Upload – aktiviert das Hochladen von Daten

ÜBERSICHT

```
easy:SetOpt_Upload(upload)
```

BESCHREIBUNG

Der auf 1 gesetzte Parameter `upload` weist die Bibliothek an, einen Upload vorzubereiten und durchzuführen. Die Optionen `#CURLOPT_READDATA` und `#CURLOPT_INFILESIZE` oder `#CURLOPT_INFILESIZE_LARGE` sind auch für Uploads interessant. Wenn das Protokoll HTTP ist, bedeutet Hochladen die Verwendung der PUT-Anforderung, sofern Sie libcurl nichts anderes mitteilen.

Die Verwendung von PUT mit HTTP 1.1 impliziert die Verwendung eines "Expect: 100-continue" Headers. Sie können diesen Header wie gewohnt mit `#CURLOPT_HTTPHEADER` deaktivieren.

Wenn Sie PUT auf einen HTTP 1.1-Server verwenden, können Sie Daten hochladen, ohne die Größe vor dem Start der Übertragung zu kennen, wenn Sie Chunked Encoding verwenden. Sie aktivieren dies, indem Sie einen Header wie "Transfer-Encoding: chunked" mit `#CURLOPT_HTTPHEADER` hinzufügen. Bei HTTP 1.0 oder ohne Chunked-Transfer müssen Sie die Größe angeben.

EINGABEN

`upload` Eingabewert

5.267 easy:SetOpt_URL

BEZEICHNUNG

easy:SetOpt_URL – gibt die URL an, die in der Anfrage verwendet werden soll

ÜBERSICHT

```
easy:SetOpt_URL(URL)
```

BESCHREIBUNG

Übergeben Sie eine Zeichenkette mit der `URL`, mit der gearbeitet werden soll. Der Parameter sollte eine Zeichenkette sein, die im folgenden Format URL-codiert sein muss:

```
scheme://host:port/path
```

Eine ausführlichere Erläuterung des Formats finden Sie in RFC3986.

libcurl überprüft die Syntax nicht und verwendet diese Variable erst, wenn die Übertragung erfolgt ist. Auch wenn Sie hier einen verrückten Wert festlegen, gibt `easy:SetOpt()` immer noch `#CURLE_OK` zurück.

Wenn der angegebenen URL ein Schemaname fehlt (z.B. "http://" oder "ftp://" usw.), wird libcurl basierend auf dem Host eine Vermutung anstellen. Wenn der äußerste Subdomänenname mit DICT, FTP, IMAP, LDAP, POP3 oder SMTP übereinstimmt, wird dieses Protokoll verwendet, andernfalls wird HTTP benutzt. Seit 7.45.0 kann das Raten durch Setzen eines Standardprotokolls deaktiviert werden. Siehe `#CURLOPT_DEFAULT_PROTOCOL` für Einzelheiten.

Sollte das vom Schema angegebene oder von libcurl aus dem Hostnamen abgeleitete Protokoll von libcurl nicht unterstützt werden, wird `#CURLE_UNSUPPORTED_PROTOCOL` entweder vom Befehl `easy:Perform()` oder `multi:Perform( )` aufgerufen. Verwenden Sie `curl_version_info`, um detaillierte Informationen darüber zu erhalten, welche Protokolle von der von Ihnen verwendeten libcurl-Version unterstützt werden.

#CURLOPT_PROTOCOLS kann verwendet werden, um zu begrenzen, welche Protokolle libcurl für diese Übertragung verwendet, unabhängig davon, für welche Unterstützung libcurl kompiliert wurde. Dies kann hilfreich sein, wenn Sie die URL von einer externen Quelle akzeptieren und die Zugänglichkeit einschränken möchten.

Die Zeichenkette #CURLOPT_URL wird ignoriert, wenn #CURLOPT_CURLU festgelegt ist.

#CURLOPT_URL oder #CURLOPT_CURLU müssen gesetzt sein, bevor eine Übertragung gestartet wird.

Der Host-Teil der URL enthält die Adresse des Servers, zu dem Sie eine Verbindung herstellen möchten. Dies kann der vollständig qualifizierte Domänenname des Servers, der lokale Netzwerkname des Computers in Ihrem Netzwerk oder die IP-Adresse des Servers oder Computers sein, die entweder durch eine IPv4- oder eine IPv6-Adresse dargestellt wird. Zum Beispiel:

```
http://www.example.com/
http://hostname/
http://192.168.0.1/
http://[2001:1890:1112:1::20]/
```

Es ist auch möglich, den Benutzernamen, das Passwort und alle unterstützten Anmeldeoptionen als Teil des Hosts für die folgenden Protokolle anzugeben, wenn eine Verbindung zu Servern hergestellt wird, die eine Authentifizierung erfordern:

```
http://user:password@www.example.com
ftp://user:password@ftp.example.com
smb://domain%2fuser:password@server.example.com
imap://user:password;options@mail.example.com
pop3://user:password;options@mail.example.com
smtp://user:password;options@mail.example.com
```

Derzeit unterstützen nur IMAP, POP3 und SMTP Anmeldeoptionen als Teil des Hosts. Weitere Informationen zu den Anmeldeoptionen in der URL-Syntax finden Sie unter RFC2384, RFC5092 und IETF-Entwurf "draft-earhart-url-smtp-00.txt" (Hinzugefügt in 7.31.0).

Der Port ist optional und wenn nicht angegeben, verwendet libcurl den Standardport basierend auf dem festgelegten oder angegebenen Protokoll: 80 für HTTP, 21 für FTP und 25 für SMTP usw. Die folgenden Beispiele zeigen, wie der Port angegeben wird:

`http://www.example.com:8080/`
: Dadurch wird eine Verbindung zu einem Webserver über Port 8080 anstelle von 80 hergestellt.

`smtp://mail.example.com:587/`
: Dadurch wird eine Verbindung zu einem SMTP-Server am alternativen Mail-Port hergestellt.

Der Pfadteil der URL ist protokollspezifisch und obwohl einige Beispiele unter dieser Liste aufgeführt sind, ist dies nicht schlüssig:

`HTTP`
: Der Pfadteil einer HTTP-Anforderung gibt die abzurufende Datei an und aus welchem Verzeichnis. Wenn das Verzeichnis nicht angegeben ist, wird das Stammverzeichnis des Webservers verwendet. Wenn die Datei weggelassen

wird, wird das Standarddokument entweder für das angegebene Verzeichnis oder das Stammverzeichnis abgerufen. Die genaue Ressource, die für jede URL zurückgegeben wird, hängt vollständig von der Serverkonfiguration ab.

`http://www.example.com`
: Dies ruft die Hauptseite vom Webserver ab.

`http://www.example.com/index.html`
: Dies gibt die Hauptseite zurück, indem sie explizit angefordert wird.

`http://www.example.com/contactus/`
: Dies gibt das Standarddokument aus dem Kontaktverzeichnis zurück.

`FTP`
: Der Pfadteil einer FTP-Anfrage gibt die abzurufende Datei an und aus welchem Verzeichnis. Wenn der Dateiteil weggelassen wird, lädt libcurl die Verzeichnisliste für das angegebene Verzeichnis herunter. Wenn das Verzeichnis weggelassen wird, wird die Verzeichnisliste für das Stammverzeichnis / Basisverzeichnis zurückgegeben.

  `ftp://ftp.example.com`
  : Dadurch wird die Verzeichnisliste für das Stammverzeichnis abgerufen.

  `ftp://ftp.example.com/readme.txt`
  : Dadurch wird die Datei readme.txt aus dem Stammverzeichnis heruntergeladen.

  `ftp://ftp.example.com/libcurl/readme.txt`
  : Dadurch wird readme.txt aus dem libcurl-Verzeichnis heruntergeladen.

  `ftp://user:password@ftp.example.com/readme.txt`
  : Das ruft die readme.txt-Datei aus dem Basisverzeichnis des Benutzers ab. Wenn ein Benutzername und ein Passwort angegeben werden, bezieht sich alles, was im Pfad angegeben wird, auf das Basisverzeichnis des Benutzers. Um Dateien aus dem Stammverzeichnis oder einem Verzeichnis unterhalb des Stammverzeichnisses abzurufen, muss der absolute Pfad angegeben werden, indem ein zusätzlicher Schrägstrich an den Anfang des Pfads angehängt wird.

  `ftp://user:password@ftp.example.com//readme.txt`
  : Dadurch wird die Datei readme.txt aus dem Stammverzeichnis abgerufen, wenn Sie sich als angegebener Benutzer anmelden.

`SMTP`
: Der Pfadteil einer SMTP-Anforderung gibt den Hostnamen an, der während der Kommunikation mit dem Mailserver angezeigt werden soll. Wenn der Pfad weggelassen wird, versucht libcurl, den Hostnamen des lokalen Computers aufzulösen. Dies gibt jedoch möglicherweise nicht den vollständig qualifizierten Domänennamen zurück, der von einigen Mailservern benötigt wird.

Wenn Sie diesen Pfad angeben, können Sie einen alternativen Namen festlegen, z.B. den vollständig qualifizierten Domänennamen Ihres Computers, den Sie möglicherweise von einer externen Funktion erhalten haben, wie gethostname oder getaddrinfo.

`smtp://mail.example.com`
: Dadurch wird eine Verbindung zum Mailserver unter example.com hergestellt und der Hostname Ihres lokalen Computers im Befehl HELO/EHLO gesendet.

`smtp://mail.example.com/client.example.com`
: Dadurch wird client.example.com im HELO/EHLO-Befehl an den Mailserver unter example.com gesendet.

`POP3` Der Pfadteil einer POP3-Anforderung gibt die abzurufende Nachrichten-ID an. Wenn die ID nicht angegeben ist, wird stattdessen eine Liste anstehender Nachrichten zurückgegeben.

`pop3://user:password@mail.example.com`
: Hier werden die verfügbaren Nachrichten für den Benutzer aufgelistet.

`pop3://user:password@mail.example.com/1`
: Dadurch wird die erste Nachricht für den Benutzer abgerufen.

`IMAP` Der Pfadteil einer IMAP-Anfrage gibt nicht nur die Mailbox an, die aufgelistet (hinzugefügt in 7.30.0) oder ausgewählt werden soll, sondern kann auch dazu verwendet werden, die UIDVALIDITY der Mailbox zu überprüfen, die UID, SECTION (hinzugefügt in 7.30.0) und PARTIELLE Oktette (hinzugefügt in 7.37.0) der zu holenden Nachricht anzugeben und zu spezifizieren, nach welchen Nachrichten gesucht werden soll (hinzugefügt in 7.37.0).

`imap://user:password@mail.example.com`
: Führt eine Ordnerliste der obersten Ebene aus.

`imap://user:password@mail.example.com/INBOX`
: Führt eine Ordnerliste im Posteingang des Benutzers durch.

`imap://user:password@mail.example.com/INBOX/;UID=1`
: Wählt den Posteingang des Benutzers aus und ruft die Nachricht mit uid = 1 ab.

`imap://user:password@mail.example.com/INBOX/;MAILINDEX=1`
: Wählt den Posteingang des Benutzers aus und ruft die erste Nachricht in der Mailbox ab.

`imap://user:password@mail.example.com/INBOX;UIDVALIDITY=50/;UID=2`
: Wählt den Posteingang des Benutzers aus, überprüft, ob die UIDVALIDITY der Mailbox 50 beträgt und ruft die Nachricht 2 ab, falls dies der Fall ist.

`imap://user:password@mail.example.com/INBOX/;UID=3/;SECTION=TEXT`
: Wählt den Posteingang des Benutzers aus und ruft den Textteil von Nachricht 3 ab.

`imap://user:password@mail.example.com/INBOX/;UID=4/;PARTIAL=0.1024`
: Wählt den Posteingang des Benutzers aus und ruft die ersten 1024 Oktette der Nachricht 4 ab.

`imap://user:password@mail.example.com/INBOX?NEW`
: Wählt den Posteingang des Benutzers aus und sucht nach NEUE Nachrichten.

`imap://user:password@mail.example.com/INBOX?SUBJECT%20shadows`
: Wählt den Posteingang des Benutzers aus und sucht in der Betreffzeile nach Nachrichten, die "shadows" enthalten.

Weitere Informationen zu den einzelnen Komponenten einer IMAP-URL finden Sie in RFC5092.

`SCP` Der Pfadteil einer SCP-Anforderung gibt die abzurufende Datei sowie aus welchem Verzeichnis an. Der Dateiteil darf nicht weggelassen werden. Die Datei wird als absoluter Pfad aus dem Stammverzeichnis auf dem Server übernommen. Wenn Sie einen Pfad relativ zum Basisverzeichnis des Benutzers auf dem Server angeben möchten, stellen Sie `~/` vor den Pfadabschnitt. Wenn der Benutzername nicht in die URL eingebettet ist, kann er mit der Option `#CURLOPT_USERPWD` oder `#CURLOPT_USERNAME` festgelegt werden.

`scp://user@example.com/etc/issue`
: Dies gibt die Datei /etc/issue an.

`scp://example.com/~/my-file`
: Dies gibt die Datei my-file im Home-Verzeichnis des Benutzers auf dem Server an.

`SFTP` Der Pfadteil einer SFTP-Anforderung gibt die abzurufende Datei sowie aus welchem Verzeichnis an. Wenn der Dateiteil weggelassen wird, lädt libcurl die Verzeichnisliste für das angegebene Verzeichnis herunter. Wenn der Pfad mit einem / endet, wird anstelle einer Datei eine Verzeichnisliste zurückgegeben. Wenn der Pfad vollständig weggelassen wird, wird die Verzeichnisliste für das Stammverzeichnis / Basisverzeichnis zurückgegeben. Wenn der Benutzername nicht in die URL eingebettet ist, kann er mit der Option `#CURLOPT_USERPWD` oder `#CURLOPT_USERNAME` festgelegt werden.

`sftp://user:password@example.com/etc/issue`
: Dies gibt die Datei /etc/issue an.

`sftp://user@example.com/~/my-file`
: Dies gibt die Datei my-file im Home-Verzeichnis des Benutzers an.

`sftp://ssh.example.com/~/Documents/`
: Dadurch wird eine Verzeichnisliste des Verzeichniss "Dokuments" im Ausgangsverzeichnis des Benutzers angefordert.

`SMB` Der Pfadteil einer SMB-Anforderung gibt die abzurufende Datei an (und darf nicht ausgelassen werden), aus welchem Share und Verzeichnis oder in

welches Share hochgeladen werden soll. Wenn der Benutzername nicht in die URL eingebettet ist, kann er mit der Option `#CURLOPT_USERPWD` oder `#CURLOPT_USERNAME` festgelegt werden. Wenn der Benutzername in die URL eingebettet ist, muss er den Domänennamen enthalten und als solcher muss der umgekehrte Schrägstrich als %2f URL codiert sein.

`smb://server.example.com/files/issue`
: Dies gibt die Datei "issue" an, die sich im Stammverzeichnis der Freigabe "files" befindet.

`smb://server.example.com/files/ -T issue`
: Dies legt fest, dass die Datei "issue" in das Stammverzeichnis der Freigabe "files" hochgeladen wird.

`LDAP`
: Der Pfadteil einer LDAP-Anfrage kann verwendet werden, um Folgendes anzugeben: Distinguished Name, Attributes, Scope, Filter and Extension für eine LDAP-Suche. Jedes Feld ist durch ein Fragezeichen getrennt. Wenn dieses Feld nicht erforderlich ist, muss eine leere Zeichenkette mit dem Fragezeichen-Trennzeichen eingefügt werden.

  `ldap://ldap.example.com/o=My%20Organisation`
  : Dadurch wird eine LDAP-Suche mit dem DN My Organisation durchgeführt.

  `ldap://ldap.example.com/o=My%20Organisation?postalAddress`
  : Dadurch wird dieselbe Suche durchgeführt, es werden jedoch nur postalAddress-Attribute zurückgegeben.

  `ldap://ldap.example.com/?rootDomainNamingContext`
  : Dies gibt einen leeren DN an und fordert Informationen zum Attribut rootDomainNamingContext für einen aktiven Directory-Server an.

  Weitere Informationen zu den einzelnen Komponenten einer LDAP-URL finden Sie in RFC4516.

`RTMP`
: Es gibt keine offizielle URL-Angaben für RTMP, daher verwendet libcurl die URL-Syntax, die von der zugrunde liegenden Bibliothek librtmp unterstützt wird. Sie hat eine Syntax, in der eine herkömmliche URL verwendet wird, gefolgt von einem Leerzeichen und einer Reihe von durch Leerzeichen getrennten name=value-Paaren.

  Während Leerzeichen normalerweise kein "legal"er Buchstabe ist, akzeptiert libcurl sie. Wenn ein Benutzer ein '#' (Hash-Zeichen) übergeben möchte, wird es als Fragment behandelt und von libcurl abgeschnitten, wenn es wörtlich angegeben wird. Sie müssen es stattdessen mit einem Backslash und dem ASCII-Wert in hexadezimaler Schreibweise "\23" angeben.

EINGABEN

`URL`
: Eingabewert

5.268 easy:SetOpt_Use_SSL

BEZEICHNUNG

easy:SetOpt_Use_SSL – fordert für die Übertragung SSL/TLS an

ÜBERSICHT

```
easy:SetOpt_Use_SSL(level)
```

BESCHREIBUNG

Übergeben Sie einen Wert mit einem der folgenden Werte, damit libcurl die gewünschte SSL-Stufe für die Übertragung verwendet.

Dies sind alle Protokolle, die mit Klartext beginnen und mit dem Befehl STARTTLS auf SSL "aktualisiert" werden.

Dies dient zum Aktivieren von SSL/TLS bei Verwendung von FTP, SMTP, POP3, IMAP usw.

`#CURLUSESSL_keine`
: Versucht nicht, SSL zu verwenden.

`#CURLUSESSL_TRY`
: Versucht es mit SSL, geht ansonsten wie gewohnt vor.

`#CURLUSESSL_CONTROL`
: Erfordert SSL für die Steuerverbindung oder schlägt mit `#CURLE_USE_SSL_FAILED` fehl.

`#CURLUSESSL_ALL`
: Erfordert SSL für die gesamte Kommunikation oder schlägt mit `#CURLE_USE_SSL_FAILED` fehl.

EINGABEN

`level` Eingabewert

5.269 easy:SetOpt_UserAgent

BEZEICHNUNG

easy:SetOpt_UserAgent – setzt den HTTP-User-Agent-Header

ÜBERSICHT

```
easy:SetOpt_UserAgent(ua)
```

BESCHREIBUNG

Übergeben Sie eine Zeichenkette als Parameter. Hiermit wird der User-Agent: header in der HTTP-Anforderung festgelegt, die an den Remote-Server gesendet wird. Dies kann verwendet werden, um Server oder Skripte zu täuschen. Sie können auch einen beliebigen benutzerdefinierten Header mit `#CURLOPT_HTTPHEADER` festlegen.

EINGABEN

`ua` Eingabewert

5.270 easy:SetOpt_UserName

BEZEICHNUNG

easy:SetOpt_UserName – legt den Benutzername für die Authentifizierung fest

ÜBERSICHT

```
easy:SetOpt_UserName(username)
```

BESCHREIBUNG

Übergeben Sie eine Zeichenkette als Parameter, der auf den Benutzernamen zeigt, der für die Übertragung verwendet werden soll.

`#CURLOPT_USERNAME` legt den Benutzernamen fest, der für die Protokollauthentifizierung verwendet werden soll. Sie sollten diese Option nicht zusammen mit der (älteren) Option `#CURLOPT_USERPWD` verwenden.

Wenn Sie die Authentifizierung Kerberos V5 mit einem Windows-basierten Server verwenden, sollten Sie den Domänennamen angeben, damit der Server ein Kerberos-Ticket erfolgreich abrufen kann. Andernfalls schlägt der erste Teil des Authentifizierungs-Handshakes möglicherweise fehl.

Bei Verwendung von NTLM kann der Benutzername einfach als Benutzername ohne den Domänennamen angegeben werden, wenn der Server Teil einer einzelnen Domäne und Gesamtstruktur ist.

Verwenden Sie zum Einschließen des Domänennamens entweder das Down-Level-Anmeldungsnamen- oder das UPN-Format (User Principal Name). Beispiel: EXAMPLE\user und user@example.com

Einige HTTP-Server (unter Windows) unterstützen auch die Einbeziehung der Domäne für die Standardauthentifizierung.

Verwenden Sie zum Angeben der Passwort- und Anmeldeoptionen zusammen mit dem Benutzernamen die Optionen `#CURLOPT_PASSWORD` und `#CURLOPT_LOGIN_OPTIONS`.

EINGABEN

`username` Eingabewert

5.271 easy:SetOpt_UserPwd

BEZEICHNUNG

easy:SetOpt_UserPwd – legt den Benutzername und das Passwort für die Authentifizierung fest

ÜBERSICHT

```
easy:SetOpt_UserPwd(userpwd)
```

BESCHREIBUNG

Übergeben Sie eine Zeichenkette als Parameter, der die Anmeldedaten für die Verbindung enthält. Das Format lautet: [user name]:[password].

Wenn Sie die Kerberos V5-Authentifizierung mit einem Windows-basierten Server verwenden, müssen Sie den Benutzernamensabschnitt mit dem Domänennamen angeben, damit der Server ein Kerberos-Ticket erfolgreich abrufen kann. Andernfalls schlägt der erste Teil des Authentifizierungs-Handshakes möglicherweise fehl.

Bei Verwendung von NTLM kann der Benutzername einfach als Benutzername ohne den Domänennamen angegeben werden, wenn der Server Teil einer einzelnen Domäne und Gesamtstruktur ist.

Verwenden Sie zum Angeben des Domänennamens die Formate Down-Level-Anmeldename oder UPN (User Principal Name). Beispiel: EXAMPLE\user beziehungsweise user@example.com.

Einige HTTP-Server (unter Windows) unterstützen auch die Einbeziehung der Domäne für die Standardauthentifizierung.

Bei Verwendung von HTTP und `#CURLOPT_FOLLOWLOCATION` führt libcurl möglicherweise mehrere Anforderungen an möglicherweise verschiedene Hosts aus. libcurl sendet diese Benutzer- und Passwortinformationen nur unter Verwendung des ursprünglichen Hostnamens an Hosts (es sei denn, `#CURLOPT_UNRESTRICTED_AUTH` ist festgelegt). Wenn libcurl also Speicherorten an andere Hosts folgt, werden Benutzer und Passwort nicht an diese Hosts gesendet. Dies wird erzwungen, um versehentliche Informationslöcher zu verhindern.

Verwenden Sie `#CURLOPT_HTTPAUTH`, um die Authentifizierungsmethode für HTTP-basierte Verbindungen anzugeben, oder `#CURLOPT_LOGIN_OPTIONS`, um IMAP-, POP3- und SMTP-Optionen zu steuern.

Die Benutzer- und Passwortzeichenketten sind nicht URL-dekodiert, daher kann mit dieser Option kein Benutzername mit Doppelpunkt gesendet werden. Verwenden Sie hierfür `#CURLOPT_USERNAME` oder fügen Sie ihn in die URL ein.

EINGABEN

`userpwd` Eingabewert

5.272 easy:SetOpt_Verbose

BEZEICHNUNG

easy:SetOpt_Verbose – schaltet den ausführlichen Modus ein/aus

ÜBERSICHT

```
easy:SetOpt_Verbose(onoff)
```

BESCHREIBUNG

Setzen Sie den Parameter `onoff` auf 1, damit die Bibliothek viele ausführliche Informationen zu ihren Vorgängen in diesem `handle` anzeigt. Sehr nützlich für das Debuggen und Verstehen von libcurl und/oder Protokollen. Die ausführlichen Informationen werden an stderr oder den mit `#CURLOPT_STDERR` festgelegten Datenstrom gesendet.

Sie werden diesen Modus kaum jemals in der Ausführung verwenden wollen, Sie werden ihn aber fast immer benutzen, wenn Sie Probleme debuggen oder Fehlermeldungen auftauchen.

Verwenden Sie die Funktion `#CURLOPT_DEBUGFUNCTION`, um auch alle gesendeten und empfangenen Protokolldaten abzurufen.

EINGABEN

`onoff` Eingabewert

5.273 easy:SetOpt_WildcardMatch

BEZEICHNUNG

easy:SetOpt_WildcardMatch – aktiviert die Übertragung von Verzeichnis-Platzhaltern

ÜBERSICHT

```
easy:SetOpt_WildcardMatch(onoff)
```

BESCHREIBUNG

Setzen Sie `onoff` auf 1, wenn Sie mehrere Dateien nach einem Dateinamenmuster übertragen möchten. Das Muster kann als Teil der Option `#CURLOPT_URL` unter Verwendung eines fnmatch-ähnlichen Musters (Shell Pattern Matching) im letzten Teil der URL (Dateiname) angegeben werden.

Standardmäßig verwendet libcurl die interne Wildcard-Matching-Implementierung. Mit der Option `#CURLOPT_FNMATCH_FUNCTION` können Sie Ihre eigene Abgleichfunktion bereitstellen.

Es folgt eine kurze Einführung in die Syntax:

`"* - ASTERISK"`

ftp://example.com/some/path/*.txt (für alle txts aus dem Stammverzeichnis). Innerhalb einer Musterzeichenkette sind nur zwei Sternchen zulässig.

`"? - QUESTION MARK"`

Das Fragezeichen entspricht einem beliebigen (genau einem) Zeichen.

ftp://example.com/some/path/photo?.jpeg

`"[ - BRACKET EXPRESSION"`

Die linke Klammer öffnet einen Klammerausdruck. Das Fragezeichen und das Sternchen haben in einem Klammerausdruck keine besondere Bedeutung. Jeder Klammerausdruck endet mit der rechten Klammer und entspricht genau einem Zeichen. Es folgen einige Beispiele:

`[a-zA-Z0\-9] or [f\-gF\-G]`
Zeichenintervall

`[abc]` Zeichenaufzählung

`[^abc] or [!abc]`
Verneinung

`[[:§name:]]`
Klassenausdruck. Unterstützte Klassen sind alnum,lower, space, alpha, digit, print, upper, blank, graph, xdigit.

`[][-!^]` Sonderfall \- entspricht nur '\ -', ']', '[', '!' oder '^'. Diese Zeichen haben keinen besonderen Zweck.

`[\[\]\\]` Escape-Syntax. Entspricht '[', ']' oder '\.

Unter Verwendung der obigen Regeln kann ein Dateinamenmuster erstellt werden:

ftp://example.com/some/path/[a-z[:upper:]\\].jpeg

EINGABEN

`onoff` Eingabewert

5.274 easy:SetOpt_WriteFunction

BEZEICHNUNG

easy:SetOpt_WriteFunction – setzt den Callback zum Schreiben empfangener Daten

ÜBERSICHT

```
easy:SetOpt_WriteFunction(write_callback[, userdata])
```

BESCHREIBUNG

Übergeben Sie eine Callbackfunktion. Diese Callbackfunktion wird von libcurl aufgerufen, sobald Daten empfangen werden, die gespeichert werden müssen. Bei den meisten Übertragungen wird dieser Callback mehrmals aufgerufen und jeder Aufruf liefert einen weiteren Datenblock.

Der erste Parameter, der an Ihre Callbackfunktion übergeben wird, ist eine Zeichenkette, die die soeben empfangenen binären Rohdaten enthält. Wenn Sie das optionale Argument `userdata` übergeben, wird der in `userdata` übergebene Wert als zweiter Parameter an Ihre Callbackfunktion übergeben. Der Parameter `userdata` kann einen beliebigen Typ haben.

Die Callback-Funktion wird in allen Aufrufen so viele Daten wie möglich übergeben, aber Sie dürfen keine Annahmen treffen. Es kann ein Byte sein, es können Tausende sein. Die maximale Menge an Stammdaten, die an den Schreibcallback übergeben wird, ist wie folgt definiert: `#CURL_MAX_WRITE_SIZE` (der übliche Standard ist 16K). Wenn `#CURLOPT_HEADER` aktiviert ist, wodurch die Header an den Schreibcallback übergeben werden, können Sie bis zu `#CURL_MAX_HTTP_HEADER` Bytes von Header erhalten, die an sie übergeben werden. Das bedeutet in der Regel 100K.

Diese Funktion kann mit Null-Byte-Daten aufgerufen werden, wenn die übertragene Datei leer ist.

Ihr Callback sollte die Anzahl der tatsächlich bearbeiteten Bytes zurückgeben. Wenn dieser Betrag von dem Betrag abweicht, der an Ihre Callbackfunktion übergeben wurde, wird der Bibliothek ein Fehler gemeldet. Dadurch wird die Übertragung abgebrochen und die verwendete libcurl-Funktion gibt `#CURLE_WRITE_ERROR` zurück.

Wenn Ihre Schreibfunktion nichts zurückgibt, signalisiert dies den Erfolg und die Übertragung wird fortgesetzt.

Wenn Ihre Callbackfunktion `#CURL_WRITEFUNC_PAUSE` zurückgibt, wird diese Übertragung angehalten. Weitere Informationen finden Sie unter dem Befehl `easy:Pause()`.

EINGABEN

`write_callback`
: Eingabewert

`userdata`
: optional: Benutzerdaten, die an die Callback-Funktion übergeben werden sollen

BEISPIEL

```
Function p_WriteData(data$)
   WriteBytes(1, data$)
EndFunction
e:SetOpt_WriteFunction(p_WriteData)
```

Der obige Code installiert eine Schreibfunktion, die alle empfangenen Daten in die Datei mit dem Identifikator 1 schreibt.

5.275 easy:SetOpt_XOAuth2_Bearer

BEZEICHNUNG

easy:SetOpt_XOAuth2_Bearer – gibt den OAuth 2.0 Access Token an

ÜBERSICHT

```
easy:SetOpt_XOAuth2_Bearer(token)
```

BESCHREIBUNG

Übergeben Sie eine Zeichenkette als Parameter mit dem OAuth 2.0 Access Token zur Verwendung mit HTTP-, IMAP-, POP3- und SMTP-Servern, die die Rahmenbedingungen der OAuth 2.0-Autorisierung unterstützen.

Hinweis: Für IMAP, POP3 und SMTP sollte der Benutzername, der zum Generieren des Tokens verwendet wird, über die Option `#CURLOPT_USERNAME` angegeben werden.

EINGABEN

`token` Eingabewert

5.276 easy:Unescape

BEZEICHNUNG

easy:Unescape – dekodiert die Zeichenkette aus der URL-Prozentdarstellung

ÜBERSICHT

```
e$ = easy:Unescape(s$)
```

BESCHREIBUNG

Dieser Befehl konvertiert die angegebene URL-kodierte Eingabezeichenkette in eine "einfache Zeichenkette" und gibt diese zurück. Alle Eingabezeichen, die URL-kodiert sind (%XX, wobei XX eine zweistellige hexadezimale Zahl ist), werden in ihre binäre Version umgewandelt.

EINGABEN

`s$` Zeichenkette zur Dekodierung

RÜCKGABEWERTE

`e$` dekodierte Zeichenkette

5.277 easy:UnsetOpt

BEZEICHNUNG

easy:UnsetOpt – stellt die Option für einen Curl-Easy-Handle auf den Standardwert zurück

ÜBERSICHT

```
easy:UnsetOpt(option)
```

BESCHREIBUNG

Diese Methode kann verwendet werden, um eine Option auf einem curl-Handle zu deaktivieren, d.h. die Option wird auf ihren Standardwert zurückgesetzt.

Die folgenden Optionstypen werden derzeit unterstützt:

`#CURLOPT_ABSTRACT_UNIX_SOCKET`
: Setzt einen abstrakten Unix-Domänen-Socket zurück. Siehe Abschnitt 5.278 [easy:UnsetOpt_Abstract_Unix_Socket], Seite 208, für Details.

`#CURLOPT_ACCEPT_ENCODING`
: Setzt die automatische Dekompression von HTTP-Downloads zurück. Siehe Abschnitt 5.279 [easy:UnsetOpt_Accept_Encoding], Seite 208, für Details.

`#CURLOPT_ACCEPTTIMEOUT_MS`
: Setzt die Zeitüberschreitung beim Warten auf die erneute Verbindung des FTP-Servers zurück. Siehe Abschnitt 5.280 [easy:UnsetOpt_AcceptTimeout_MS], Seite 208, für Details.

`#CURLOPT_ADDRESS_SCOPE`
: Setzt den Bereich für lokale IPv6-Adressen zurück. Siehe Abschnitt 5.281 [easy:UnsetOpt_Address_Scope], Seite 209, für Details.

`#CURLOPT_APPEND`
: Deaktiviert das Anhängen an die Remote-Datei. Siehe Abschnitt 5.282 [easy:UnsetOpt_Append], Seite 209, für Details.

`#CURLOPT_AUTOREFERER`
: Dektiviert die automatische Aktualisierung des Referer-Header. Siehe Abschnitt 5.283 [easy:UnsetOpt_AutoReferer], Seite 209, für Details.

`#CURLOPT_BUFFERSIZE`
: Stellt die bevorzugte Empfangspuffergröße zurück. Siehe Abschnitt 5.284 [easy:UnsetOpt_BufferSize], Seite 210, für Details.

`#CURLOPT_CAINFO`
: Setzt den Pfad zum Paket der Zertifizierungsstelle (CA) zurück. Siehe Abschnitt 5.285 [easy:UnsetOpt_CAInfo], Seite 210, für Details.

`#CURLOPT_CAPATH`
: Setzt das Verzeichnis mit CA-Zertifikaten zurück. Siehe Abschnitt 5.286 [easy:UnsetOpt_CAPath], Seite 210, für Details.

`#CURLOPT_CERTINFO`
: Deaktiviert die Anforderung von SSL-Zertifikat-Informationen. Siehe Abschnitt 5.287 [easy:UnsetOpt_CertInfo], Seite 210, für Details.

`#CURLOPT_CHUNK_BGN_FUNCTION`
: Deaktiviert den Callback vor einer Übertragung mit FTP Platzhalter Übereinstimmung. Siehe Abschnitt 5.288 [`easy:UnsetOpt_Chunk_BGN_Function`], Seite 211, für Details.

`#CURLOPT_CHUNK_END_FUNCTION`
: Deaktiviert den Callback nach einer Übertragung mit FTP Platzhalter Übereinstimmung. Siehe Abschnitt 5.289 [`easy:UnsetOpt_Chunk_End_Function`], Seite 211, für Details.

`#CURLOPT_CONNECT_ONLY`
: Stoppt nicht mehr, wenn eine Verbindung zum Zielserver besteht. Siehe Abschnitt 5.290 [`easy:UnsetOpt_Connect_Only`], Seite 211, für Details.

`#CURLOPT_CONNECT_TO`
: Verbindet sich wieder mit dem Host und Port der URL. Siehe Abschnitt 5.291 [`easy:UnsetOpt_Connect_To`], Seite 212, für Details.

`#CURLOPT_CONNECTTIMEOUT`
: Setzt die Zeitüberschreitung für die Verbindungsphase zurück. Siehe Abschnitt 5.292 [`easy:UnsetOpt_ConnectTimeout`], Seite 212, für Details.

`#CURLOPT_CONNECTTIMEOUT_MS`
: Setzt die Zeitüberschreitung für die Verbindungsphase zurück. Siehe Abschnitt 5.293 [`easy:UnsetOpt_ConnectTimeout_MS`], Seite 212, für Details.

`#CURLOPT_COOKIE`
: Setzt den Inhalt des HTTP-Cookie-Headers zurück. Siehe Abschnitt 5.294 [`easy:UnsetOpt_Cookie`], Seite 212, für Details.

`#CURLOPT_COOKIEFILE`
: Setzt den Cookie-Dateinamen zurück. Siehe Abschnitt 5.295 [`easy:UnsetOpt_CookieFile`], Seite 213, für Details.

`#CURLOPT_COOKIEJAR`
: Setzt den Cookie-Dateinamen zum Speichern zurück. Siehe Abschnitt 5.296 [`easy:UnsetOpt_CookieJar`], Seite 213, für Details.

`#CURLOPT_COOKIELIST`
: Setzt den internen Cookie-Speicher zurück. Siehe Abschnitt 5.297 [`easy:UnsetOpt_CookieList`], Seite 213, für Details.

`#CURLOPT_COOKIESESSION`
: Beendet eine neue Cookie-Sitzung. Siehe Abschnitt 5.298 [`easy:UnsetOpt_CookieSession`], Seite 214, für Details.

`#CURLOPT_CRLF`
: Deaktiviert die CRLF-Konvertierung. Siehe Abschnitt 5.299 [`easy:UnsetOpt_CRLF`], Seite 214, für Details.

`#CURLOPT_CRLFILE`
: Setzt die Datei für Zertifikatssperrlisten zurück. Siehe Abschnitt 5.300 [`easy:UnsetOpt_CRLFile`], Seite 214, für Details.

`#CURLOPT_CUSTOMREQUEST`
: Stellt den internen Standard für die Anforderung wieder her. Siehe Abschnitt 5.301 [`easy:UnsetOpt_CustomRequest`], Seite 214, für Details.

`#CURLOPT_DEBUGFUNCTION`
: Setzt wieder die Standard-Debug-Funktion ein. Siehe Abschnitt 5.302 [`easy:UnsetOpt_DebugFunction`], Seite 215, für Details.

`#CURLOPT_DEFAULT_PROTOCOL`
: Setzt das Standardprotokoll bei fehlendem Schemanamen zurück. Siehe Abschnitt 5.303 [`easy:UnsetOpt_Default_Protocol`], Seite 215, für Details.

`#CURLOPT_DIRLISTONLY`
: Fragt wieder nach der gesammten Verzeichnisliste. Siehe Abschnitt 5.304 [`easy:UnsetOpt_DirListOnly`], Seite 215, für Details.

`#CURLOPT_DNS_CACHE_TIMEOUT`
: Setzt die Lebensdauer für DNS-Cache-Einträge wieder zurück. Siehe Abschnitt 5.305 [`easy:UnsetOpt_DNS_Cache_Timeout`], Seite 216, für Details.

`#CURLOPT_DNS_INTERFACE`
: Bindet nicht mehr an eine bestimmte Schnittstelle. Siehe Abschnitt 5.306 [`easy:UnsetOpt_DNS_Interface`], Seite 216, für Details.

`#CURLOPT_DNS_LOCAL_IP4`
: Bindet nicht mehr an eine bestimmte IP-Adresse. Siehe Abschnitt 5.307 [`easy:UnsetOpt_DNS_Local_IP4`], Seite 216, für Details.

`#CURLOPT_DNS_LOCAL_IP6`
: Bindet nicht mehr an eine bestimmte IP-Adresse. Siehe Abschnitt 5.308 [`easy:UnsetOpt_DNS_Local_IP6`], Seite 216, für Details.

`#CURLOPT_DNS_SERVERS`
: Setzt die DNS-Server-Liste auf die Systemvoreinstellung zurück. Siehe Abschnitt 5.309 [`easy:UnsetOpt_DNS_Servers`], Seite 217, für Details.

`#CURLOPT_DNS_USE_GLOBAL_CACHE`
: VERALTET: Deaktiviert den globalen DNS-Cache. Siehe Abschnitt 5.310 [`easy:UnsetOpt_DNS_Use_Global_Cache`], Seite 217, für Details.

`#CURLOPT_EGDSOCKET`
: Setzt den EGD-Socketpfad zurück. Siehe Abschnitt 5.311 [`easy:UnsetOpt_EGDSocket`], Seite 217, für Details.

`#CURLOPT_EXPECT_100_TIMEOUT_MS`
: Setzt die Zeitüberschreitung bei der Antwort Expect: 100-continue zurück. Siehe Abschnitt 5.312 [`easy:UnsetOpt_Expect_100_Timeout_MS`], Seite 218, für Details.

`#CURLOPT_FAILONERROR`
: Setzt den Anforderungsfehler bei HTTP-Antwort >= 400 zurück. Siehe Abschnitt 5.313 [`easy:UnsetOpt_FailOnError`], Seite 218, für Details.

`#CURLOPT_FILETIME`
: Fordert keine Änderungszeit der Remote-Datenquelle an. Siehe Abschnitt 5.314 [`easy:UnsetOpt_FileTime`], Seite 218, für Details.

`#CURLOPT_FNMATCH_FUNCTION`
: Setzt die Callback-Platzhalterabgleich-Funktion zurück. Siehe Abschnitt 5.315 [`easy:UnsetOpt_FNMatch_Function`], Seite 218, für Details.

`#CURLOPT_FOLLOWLOCATION`
: Stellt wieder die HTTP 3xx Standortverfolgung ein. Siehe Abschnitt 5.316 [`easy:UnsetOpt_FollowLocation`], Seite 219, für Details.

`#CURLOPT_FORBID_REUSE`
: Lässt die Verbindung stehen, nachdem die Übertragung beendet ist. Siehe Abschnitt 5.317 [`easy:UnsetOpt_Forbid_Reuse`], Seite 219, für Details.

`#CURLOPT_FRESH_CONNECT`
: Erzwingt keine neue Verbindung mehr. Siehe Abschnitt 5.318 [`easy:UnsetOpt_Fresh_Connect`], Seite 219, für Details.

`#CURLOPT_FTP_ACCOUNT`
: Sendet keine Kontoinformationen an den FTP-Server. Siehe Abschnitt 5.319 [`easy:UnsetOpt_FTP_Account`], Seite 220, für Details.

`#CURLOPT_FTP_ALTERNATIVE_TO_USER`
: FTP wird nicht mehr anstelle von USER verwendet. Siehe Abschnitt 5.320 [`easy:UnsetOpt_FTP_Alternative_To_User`], Seite 220, für Details.

`#CURLOPT_FTP_CREATE_MISSING_DIRS`
: Erstellt keine fehlende Verzeichnisse mehr für FTP und SFTP. Siehe Abschnitt 5.321 [`easy:UnsetOpt_FTP_Create_Missing_Dirs`], Seite 220, für Details.

`#CURLOPT_FTP_FILEMETHOD`
: Stellt das Verzeichnisdurchlaufverfahren für FTP wieder auf Standard. Siehe Abschnitt 5.322 [`easy:UnsetOpt_FTP_FileMethod`], Seite 220, für Details.

`#CURLOPT_FTP_RESPONSE_TIMEOUT`
: Setzt die FTP-Antwortzeit wieder zurück. Siehe Abschnitt 5.323 [`easy:UnsetOpt_FTP_Response_Timeout`], Seite 221, für Details.

`#CURLOPT_FTP_SKIP_PASV_IP`
: Verwendet wieder die IP-Adresse in der PASV-Antwort. Siehe Abschnitt 5.324 [`easy:UnsetOpt_FTP_Skip_PASV_IP`], Seite 221, für Details.

`#CURLOPT_FTP_SSL_CCC`
: Lässt SSL mit FTP nach der Authentifizierung eingestellt. Siehe Abschnitt 5.325 [`easy:UnsetOpt_FTP_SSL_CCC`], Seite 221, für Details.

`#CURLOPT_FTP_USE_EPRT`
: Deaktiviert die Nutzung von EPRT mit FTP. Siehe Abschnitt 5.326 [`easy:UnsetOpt_FTP_Use_Eprt`], Seite 222, für Details.

`#CURLOPT_FTP_USE_EPSV`
: Deaktiviert die Nutzung von EPSV mit FTP. Siehe Abschnitt 5.327 [easy:UnsetOpt_FTP_Use_Epsv], Seite 222, für Details.

`#CURLOPT_FTP_USE_PRET`
: Deaktiviert den PRET-Befehl mit FTP. Siehe Abschnitt 5.328 [easy:UnsetOpt_FTP_Use_Pret], Seite 222, für Details.

`#CURLOPT_FTPPORT`
: Deaktiviert die FTP-Übertragung. Siehe Abschnitt 5.329 [easy:UnsetOpt_FTPPort], Seite 223, für Details.

`#CURLOPT_FTPSSLAUTH`
: Setzt die Reihenfolge von TLS/SSL zurück. Siehe Abschnitt 5.330 [easy:UnsetOpt_FTPSSLAuth], Seite 223, für Details.

`#CURLOPT_GSSAPI_DELEGATION`
: Deaktiviert die erlaubte Zuordnung von GSS-APIs. Siehe Abschnitt 5.331 [easy:UnsetOpt_GSSAPI_Delegation], Seite 223, für Details.

`#CURLOPT_HEADER`
: Übergibt keinen Header an den Datenstrom. Siehe Abschnitt 5.332 [easy:UnsetOpt_Header], Seite 223, für Details.

`#CURLOPT_HEADERFUNCTION`
: Setzt die Callback-Funktion für Header-Daten zurück. Siehe Abschnitt 5.333 [easy:UnsetOpt_HeaderFunction], Seite 224, für Details.

`#CURLOPT_HEADEROPT`
: Setzt das Senden von HTTP-Header zurück. Siehe Abschnitt 5.334 [easy:UnsetOpt_HeaderOpt], Seite 224, für Details.

`#CURLOPT_HTTP200ALIASES`
: Setzt alternative Übereinstimmungen für HTTP 200 OK zurück. Siehe Abschnitt 5.335 [easy:UnsetOpt_HTTP200Aliases], Seite 224, für Details.

`#CURLOPT_HTTP_CONTENT_DECODING`
: Deaktiviert die Dekodierung von HTTP-Inhalten. Siehe Abschnitt 5.336 [easy:UnsetOpt_HTTP_Content_Decoding], Seite 225, für Details.

`#CURLOPT_HTTP_TRANSFER_DECODING`
: Deaktiviert die Dekodierung der HTTP-Übertragung. Siehe Abschnitt 5.337 [easy:UnsetOpt_HTTP_Transfer_Decoding], Seite 225, für Details.

`#CURLOPT_HTTP_VERSION`
: Setzt die zu verwendende HTTP-Protokollversion zurück. Siehe Abschnitt 5.338 [easy:UnsetOpt_HTTP_Version], Seite 225, für Details.

`#CURLOPT_HTTPAUTH`
: Setzt die HTTP-Server-Authentifizierungsmethoden zurück. Siehe Abschnitt 5.339 [easy:UnsetOpt_HTTPAuth], Seite 226, für Details.

`#CURLOPT_HTTPGET`
: Setzt die HTTP GET-Anfrage zurück. Siehe Abschnitt 5.340 [easy:UnsetOpt_HTTPGet], Seite 226, für Details.

`#CURLOPT_HTTPHEADER`
: Setzt den benutzerdefinierten HTTP-Header zurück. Siehe Abschnitt 5.341 [`easy:UnsetOpt_HTTPHeader`], Seite 226, für Details.

`#CURLOPT_HTTPPOST`
: Setzt den mehrteiligen Formpost-Inhalt zurück. Siehe Abschnitt 5.342 [`easy:UnsetOpt_HTTPPost`], Seite 226, für Details.

`#CURLOPT_HTTPPROXYTUNNEL`
: Hebt den Tunnel durch einen HTTP-Proxy auf. Siehe Abschnitt 5.343 [`easy:UnsetOpt_HTTPProxyTunnel`], Seite 227, für Details.

`#CURLOPT_IGNORE_CONTENT_LENGTH`
: Bezieht den Content-Length-Header wieder ein. Siehe Abschnitt 5.344 [`easy:UnsetOpt_Ignore_Content_Length`], Seite 227, für Details.

`#CURLOPT_INFILESIZE`
: Setzt die Größe der zu sendenden Eingabedatei zurück. Siehe Abschnitt 5.345 [`easy:UnsetOpt_InFileSize`], Seite 227, für Details.

`#CURLOPT_INFILESIZE_LARGE`
: Setzt die Größe der zu sendenden Eingabedatei zurück. Siehe Abschnitt 5.346 [`easy:UnsetOpt_InFileSize_Large`], Seite 228, für Details.

`#CURLOPT_INTERFACE`
: Setzt die Quellschnittstelle für ausgehenden Datenverkehr zurück. Siehe Abschnitt 5.347 [`easy:UnsetOpt_Interface`], Seite 228, für Details.

`#CURLOPT_IPRESOLVE`
: Setzt die verwendete IP-Protokollversion zurück. Siehe Abschnitt 5.348 [`easy:UnsetOpt_IPResolve`], Seite 228, für Details.

`#CURLOPT_ISSUERCERT`
: Setzt den Dateiname des SSL-Zertifikats des Ausstellers zurück. Siehe Abschnitt 5.349 [`easy:UnsetOpt_IssuerCert`], Seite 228, für Details.

`#CURLOPT_KEEP_SENDING_ON_ERROR`
: Sendet mit einer frühen HTTP-Antwort >= 300 nicht mehr weiter. Siehe Abschnitt 5.350 [`easy:UnsetOpt_Keep_Sending_On_Error`], Seite 229, für Details.

`#CURLOPT_KEYPASSWD`
: Setzt die Passphrase des privaten Schlüssel zurück. Siehe Abschnitt 5.351 [`easy:UnsetOpt_KeyPasswd`], Seite 229, für Details.

`#CURLOPT_KRBLEVEL`
: Deaktiviert die FTP-Kerberos-Sicherheitsstufe. Siehe Abschnitt 5.352 [`easy:UnsetOpt_KRBLevel`], Seite 229, für Details.

`#CURLOPT_LOCALPORT`
: Setzt die lokale Portnummer für Socket zurück. Siehe Abschnitt 5.353 [`easy:UnsetOpt_LocalPort`], Seite 230, für Details.

`#CURLOPT_LOCALPORTRANGE`
: Setzt die Anzahl zusätzlicher lokaler Ports zum Testen zurück. Siehe Abschnitt 5.354 [`easy:UnsetOpt_LocalPortRange`], Seite 230, für Details.

`#CURLOPT_LOGIN_OPTIONS`
: Setzt die Login-Optionen zurück. Siehe Abschnitt 5.355 [`easy:UnsetOpt_Login_Options`], Seite 230, für Details.

`#CURLOPT_LOW_SPEED_LIMIT`
: Setzt die niedrige Geschwindigkeitsbegrenzung zurück. Siehe Abschnitt 5.356 [`easy:UnsetOpt_Low_Speed_Limit`], Seite 230, für Details.

`#CURLOPT_LOW_SPEED_TIME`
: Setzt das Zeitlimit für niedrige Geschwindigkeit zurück. Siehe Abschnitt 5.357 [`easy:UnsetOpt_Low_Speed_Time`], Seite 231, für Details.

`#CURLOPT_MAIL_AUTH`
: Setzt die SMTP-Authentifizierungsadresse zurück. Siehe Abschnitt 5.358 [`easy:UnsetOpt_Mail_Auth`], Seite 231, für Details.

`#CURLOPT_MAIL_FROM`
: Setzt die SMTP-Absenderadresse zurück. Siehe Abschnitt 5.359 [`easy:UnsetOpt_Mail_From`], Seite 231, für Details.

`#CURLOPT_MAIL_RCPT`
: Setzt die Liste der SMTP-Mail-Empfänger zurück. Siehe Abschnitt 5.360 [`easy:UnsetOpt_Mail_RCPT`], Seite 232, für Details.

`#CURLOPT_MAX_RECV_SPEED_LARGE`
: Setzt die Geschwindigkeitslimit für das Herunterladen von Daten zurück. Siehe Abschnitt 5.361 [`easy:UnsetOpt_Max_Recv_Speed_Large`], Seite 232, für Details.

`#CURLOPT_MAX_SEND_SPEED_LARGE`
: Setzt die Geschwindigkeitslimit für das Hochladen von Daten zurück. Siehe Abschnitt 5.362 [`easy:UnsetOpt_Max_Send_Speed_Large`], Seite 232, für Details.

`#CURLOPT_MAXCONNECTS`
: Setzt die maximale Verbindungs-Cache-Größe zurück. Siehe Abschnitt 5.363 [`easy:UnsetOpt_MaxConnects`], Seite 232, für Details.

`#CURLOPT_MAXFILESIZE`
: Setzt die maximal zulässige Dateigröße für das Herunterladen zurück. Siehe Abschnitt 5.364 [`easy:UnsetOpt_MaxFileSize`], Seite 233, für Details.

`#CURLOPT_MAXFILESIZE_LARGE`
: Setzt die maximal zulässige Dateigröße für das Herunterladen zurück. Siehe Abschnitt 5.365 [`easy:UnsetOpt_MaxFileSize_Large`], Seite 233, für Details.

`#CURLOPT_MAXREDIRS`
: Setzt die maximale Anzahl von erlaubten Umleitungen zurück. Siehe Abschnitt 5.366 [`easy:UnsetOpt_MaxRedirs`], Seite 233, für Details.

`#CURLOPT_NETRC`
: Ignoriert wieder die `.netrc`-Informationen. Siehe Abschnitt 5.367 [`easy:UnsetOpt_Netrc`], Seite 234, für Details.

`#CURLOPT_NETRC_FILE`
: Setzt den Dateiname zum Lesen von `.netrc`-Informationen zurück. Siehe Abschnitt 5.368 [`easy:UnsetOpt_Netrc_File`], Seite 234, für Details.

`#CURLOPT_NEW_DIRECTORY_PERMS`
: Setzt die Berechtigungen für neu erstellte Remote-Verzeichnisse zurück. Siehe Abschnitt 5.369 [`easy:UnsetOpt_New_Directory_Perms`], Seite 234, für Details.

`#CURLOPT_NEW_FILE_PERMS`
: Setzt die Berechtigungen für neu erstellte Remote-Dateien zurück. Siehe Abschnitt 5.370 [`easy:UnsetOpt_New_File_Perms`], Seite 235, für Details.

`#CURLOPT_NOBODY`
: Führt die Anfrage wieder mit Body-Download durch. Siehe Abschnitt 5.371 [`easy:UnsetOpt_Nobody`], Seite 235, für Details.

`#CURLOPT_NOPROGRESS`
: Schaltet die Fortschrittsanzeige wieder ein. Siehe Abschnitt 5.372 [`easy:UnsetOpt_NoProgress`], Seite 235, für Details.

`#CURLOPT_NOPROXY`
: Aktiviert wieder die Proxy-Nutzung für bestimmte Hosts. Siehe Abschnitt 5.373 [`easy:UnsetOpt_NoProxy`], Seite 235, für Details.

`#CURLOPT_NOSIGNAL`
: Überspringt die gesamte Signalverarbeitung nicht mehr. Siehe Abschnitt 5.374 [`easy:UnsetOpt_NoSignal`], Seite 236, für Details.

`#CURLOPT_PASSWORD`
: Setzt das Passwort zur Verwendung bei der Authentifizierung zurück. Siehe Abschnitt 5.375 [`easy:UnsetOpt_Password`], Seite 236, für Details.

`#CURLOPT_PATH_AS_IS`
: Verwendet wieder Punkt-Punkt-Sequenzen. Siehe Abschnitt 5.376 [`easy:UnsetOpt_Path_As_Is`], Seite 236, für Details.

`#CURLOPT_PINNEDPUBLICKEY`
: Setzt das Public Key Pinning zurück. Siehe Abschnitt 5.377 [`easy:UnsetOpt_PinnedPublicKey`], Seite 237, für Details.

`#CURLOPT_PIPEWAIT`
: Wartet nicht mehr auf Pipelining/Multiplexing. Siehe Abschnitt 5.378 [`easy:UnsetOpt_PipeWait`], Seite 237, für Details.

`#CURLOPT_PORT`
: Die URL legt wieder fest, welcher Port verwendet wird. Siehe Abschnitt 5.379 [`easy:UnsetOpt_Port`], Seite 237, für Details.

`#CURLOPT_POST`
: Fordert keinen HTTP-POST mehr an. Siehe Abschnitt 5.380 [easy:UnsetOpt_Post], Seite 237, für Details.

`#CURLOPT_POSTFIELDS`
: Setzt die Daten zurück, welche an den Server gesendet werden. Siehe Abschnitt 5.381 [easy:UnsetOpt_PostFields], Seite 238, für Details.

`#CURLOPT_POSTQUOTE`
: Setzt die (S)FTP-Befehle zur Ausführung nach der Übertragung zurück. Siehe Abschnitt 5.382 [easy:UnsetOpt_PostQuote], Seite 238, für Details.

`#CURLOPT_POSTREDIR`
: Setzt die Vorgehensweise bei einer HTTP-POST-Umleitung zurück. Siehe Abschnitt 5.383 [easy:UnsetOpt_PostRedir], Seite 238, für Details.

`#CURLOPT_PRE_PROXY`
: Setzt den Prä-Proxy für die Verwendung zurück. Siehe Abschnitt 5.384 [easy:UnsetOpt_Pre_Proxy], Seite 239, für Details.

`#CURLOPT_PREQUOTE`
: Setzt die Befehle zurück, die vor einer FTP-Übertragung ausgeführt werden sollen. Siehe Abschnitt 5.385 [easy:UnsetOpt_Prequote], Seite 239, für Details.

`#CURLOPT_PROGRESSFUNCTION`
: Setzt den Callback zur Fortschrittsanzeige-Funktion zurück. Siehe Abschnitt 5.386 [easy:UnsetOpt_ProgressFunction], Seite 239, für Details.

`#CURLOPT_PROTOCOLS`
: Erlaubt wieder alle Protokolle zu verwenden. Siehe Abschnitt 5.387 [easy:UnsetOpt_Protocols], Seite 239, für Details.

`#CURLOPT_PROXY`
: Setzt den Proxy für die Verwendung zurück. Siehe Abschnitt 5.388 [easy:UnsetOpt_Proxy], Seite 240, für Details.

`#CURLOPT_PROXY_CAINFO`
: Setzt den Pfad zum Proxy Certificate Authority (CA)-Paket zurück. Siehe Abschnitt 5.389 [easy:UnsetOpt_Proxy_CAInfo], Seite 240, für Details.

`#CURLOPT_PROXY_CAPATH`
: Setzt das Verzeichnis mit Proxy-CA-Zertifikaten zurück. Siehe Abschnitt 5.390 [easy:UnsetOpt_Proxy_CAPath], Seite 240, für Details.

`#CURLOPT_PROXY_CRLFILE`
: Setzt die Datei für Proxy-Zertifikatssperrlisten zurück. Siehe Abschnitt 5.391 [easy:UnsetOpt_Proxy_CRLFile], Seite 241, für Details.

`#CURLOPT_PROXY_KEYPASSWD`
: Setzt die Passphrase auf privaten Proxy-Schlüssel zurück. Siehe Abschnitt 5.392 [easy:UnsetOpt_Proxy_KeyPasswd], Seite 241, für Details.

`#CURLOPT_PROXY_PINNEDPUBLICKEY`
: Setzt das Public Key Pinning für https-Proxy zurück. Siehe Abschnitt 5.393 [easy:UnsetOpt_Proxy_PinnedPublicKey], Seite 241, für Details.

`#CURLOPT_PROXY_SERVICE_NAME`
: Setzt den Namen wieder auf den Standarddienstnamen zurück. Siehe Abschnitt 5.394 [easy:UnsetOpt_Proxy_Service_Name], Seite 241, für Details.

`#CURLOPT_PROXY_SSL_CIPHER_LIST`
: Setzt die für Proxy-TLS zu verwendenden Verschlüsselungsart zurück. Siehe Abschnitt 5.395 [easy:UnsetOpt_Proxy_SSL_Cipher_List], Seite 242, für Details.

`#CURLOPT_PROXY_SSL_OPTIONS`
: Setzt die Proxy-SSL-Verhaltensoptionen zurück. Siehe Abschnitt 5.396 [easy:UnsetOpt_Proxy_SSL_Options], Seite 242, für Details.

`#CURLOPT_PROXY_SSL_VERIFYHOST`
: Setzt die Überprüfung des Namens auf den Standard zurück. Siehe Abschnitt 5.397 [easy:UnsetOpt_Proxy_SSL_VerifyHost], Seite 242, für Details.

`#CURLOPT_PROXY_SSL_VERIFYPEER`
: Deaktiviert die Überprüfung des SSL-Zertifikats des Proxys. Siehe Abschnitt 5.398 [easy:UnsetOpt_Proxy_SSL_VerifyPeer], Seite 243, für Details.

`#CURLOPT_PROXY_SSLCERT`
: Setzt das SSL-Proxy-Client-Zertifikat zurück. Siehe Abschnitt 5.399 [easy:UnsetOpt_Proxy_SSLCert], Seite 243, für Details.

`#CURLOPT_PROXY_SSLCERTTYPE`
: Setzt den Typ des Proxy-Client-SSL-Zertifikats zurück. Siehe Abschnitt 5.400 [easy:UnsetOpt_Proxy_SSLCertType], Seite 243, für Details.

`#CURLOPT_PROXY_SSLKEY`
: Setzt die private Schlüsseldatei für das TLS- und SSL-Proxy-Client-Zertifikat zurück. Siehe Abschnitt 5.401 [easy:UnsetOpt_Proxy_SSLKey], Seite 244, für Details.

`#CURLOPT_PROXY_SSLKEYTYPE`
: Setzt den Typ der privaten Proxy-Schlüsseldatei zurück. Siehe Abschnitt 5.402 [easy:UnsetOpt_Proxy_SSLKeyType], Seite 244, für Details.

`#CURLOPT_PROXY_SSLVERSION`
: Setzt die bevorzugte Proxy-TLS/SSL-Version zurück. Siehe Abschnitt 5.403 [easy:UnsetOpt_Proxy_SSLVersion], Seite 244, für Details.

`#CURLOPT_PROXY_TLSAUTH_PASSWORD`
: Setzt das Passwort für die Proxy-TLS-Authentifizierung zurück. Siehe Abschnitt 5.404 [`easy:UnsetOpt_Proxy_TLSAuth_Password`], Seite 244, für Details.

`#CURLOPT_PROXY_TLSAUTH_TYPE`
: Setzt die Proxy-TLS-Authentifizierungsmethoden zurück. Siehe Abschnitt 5.405 [`easy:UnsetOpt_Proxy_TLSAuth_Type`], Seite 245, für Details.

`#CURLOPT_PROXY_TLSAUTH_USERNAME`
: Setzt den Benutzername zur Verwendung für die Proxy-TLS-Authentifizierung zurück. Siehe Abschnitt 5.406 [`easy:UnsetOpt_Proxy_TLSAuth_UserName`], Seite 245, für Details.

`#CURLOPT_PROXY_TRANSFER_MODE`
: Hängt den FTP-Übertragungsmodus nicht mehr an die URL für Proxy an. Siehe Abschnitt 5.407 [`easy:UnsetOpt_Proxy_Transfer_Mode`], Seite 245, für Details.

`#CURLOPT_PROXYAUTH`
: Setzt die HTTP-Proxy-Authentifizierungsmethoden für den Versuch zurück. Siehe Abschnitt 5.408 [`easy:UnsetOpt_ProxyAuth`], Seite 246, für Details.

`#CURLOPT_PROXYHEADER`
: Setzt die an den Proxy zu übergebenden benutzerdefinierte HTTP-Header zurück. Siehe Abschnitt 5.409 [`easy:UnsetOpt_ProxyHeader`], Seite 246, für Details.

`#CURLOPT_PROXYPASSWORD`
: Setzt das Passwort für die Proxy-Authentifizierung zurück. Siehe Abschnitt 5.410 [`easy:UnsetOpt_ProxyPassword`], Seite 246, für Details.

`#CURLOPT_PROXYPORT`
: Setzt die Portnummer für den Proxy zurück. Siehe Abschnitt 5.411 [`easy:UnsetOpt_ProxyPort`], Seite 247, für Details.

`#CURLOPT_PROXYTYPE`
: Setzt den Proxy-Protokolltyp zurück. Siehe Abschnitt 5.412 [`easy:UnsetOpt_ProxyType`], Seite 247, für Details.

`#CURLOPT_PROXYUSERNAME`
: Setzt den Benutzername für die Proxy-Authentifizierung zurück. Siehe Abschnitt 5.413 [`easy:UnsetOpt_ProxyUserName`], Seite 247, für Details.

`#CURLOPT_PROXYUSERPWD`
: Setzt den Benutzernamen und das Passwort für die Proxy-Authentifizierung zurück. Siehe Abschnitt 5.414 [`easy:UnsetOpt_ProxyUserPwd`], Seite 247, für Details.

`#CURLOPT_PUT`
: Stellt keine HTTP-PUT-Anfrage mehr. Siehe Abschnitt 5.415 [`easy:UnsetOpt_Put`], Seite 248, für Details.

`#CURLOPT_QUOTE`
: Deaktiviert die (S)FTP-Befehle, die vor der Übertragung ausgeführt werden sollen. Siehe Abschnitt 5.416 [`easy:UnsetOpt_Quote`], Seite 248, für Details.

`#CURLOPT_RANDOM_FILE`
: Setzt die Quelle für zufällige Daten zurück. Siehe Abschnitt 5.417 [`easy:UnsetOpt_Random_File`], Seite 248, für Details.

`#CURLOPT_RANGE`
: Setzt den anzuforderenden Bytebereich zurück. Siehe Abschnitt 5.418 [`easy:UnsetOpt_Range`], Seite 249, für Details.

`#CURLOPT_READFUNCTION`
: Setzt den Callback für Daten-Uploads zurück. Siehe Abschnitt 5.419 [`easy:UnsetOpt_ReadFunction`], Seite 249, für Details.

`#CURLOPT_REDIR_PROTOCOLS`
: Setzt die Protokolle zurück, zu denen umgeleitet werden darf. Siehe Abschnitt 5.420 [`easy:UnsetOpt_Redir_Protocols`], Seite 249, für Details.

`#CURLOPT_REFERER`
: Setzt den HTTP Referer: Header zurück. Siehe Abschnitt 5.421 [`easy:UnsetOpt_Referer`], Seite 249, für Details.

`#CURLOPT_REQUEST_TARGET`
: Setzt das alternative Ziel für diese Anforderung zurück. Siehe Abschnitt 5.422 [`easy:UnsetOpt_Request_Target`], Seite 250, für Details.

`#CURLOPT_RESOLVE`
: Setzt den benutzerdefinierten Hostnamen für IP-Adressauflösungen zurück. Siehe Abschnitt 5.423 [`easy:UnsetOpt_Resolve`], Seite 250, für Details.

`#CURLOPT_RESUME_FROM`
: Deaktiviert die Fortsetzung der Übertragung und beginnt von vorne. Siehe Abschnitt 5.424 [`easy:UnsetOpt_Resume_From`], Seite 250, für Details.

`#CURLOPT_RESUME_FROM_LARGE`
: Deaktiviert die Fortsetzung der Übertragung und beginnt von vorne. Siehe Abschnitt 5.425 [`easy:UnsetOpt_Resume_From_Large`], Seite 251, für Details.

`#CURLOPT_RTSP_CLIENT_CSEQ`
: Setzt die RTSP-Client-CSEQ-Nummer zurück. Siehe Abschnitt 5.426 [`easy:UnsetOpt_RTSP_Client_CSeq`], Seite 251, für Details.

`#CURLOPT_RTSP_REQUEST`
: Setzt die RTSP-Anfrage zurück. Siehe Abschnitt 5.427 [`easy:UnsetOpt_RTSP_Request`], Seite 251, für Details.

`#CURLOPT_RTSP_SERVER_CSEQ`
: Setzt die CSEQ-Nummer des RTSP-Servers zurück. Siehe Abschnitt 5.428 [`easy:UnsetOpt_RTSP_Server_CSeq`], Seite 251, für Details.

`#CURLOPT_RTSP_SESSION_ID`
: Setzt die RTSP-Sitzungs-ID zurück. Siehe Abschnitt 5.429 [easy:UnsetOpt_RTSP_Session_ID], Seite 252, für Details.

`#CURLOPT_RTSP_STREAM_URI`
: Setzt die RTSP-Stream-URI zurück. Siehe Abschnitt 5.430 [easy:UnsetOpt_RTSP_Stream_URI], Seite 252, für Details.

`#CURLOPT_RTSP_TRANSPORT`
: Setzt den RTSP Transport: Header zurück. Siehe Abschnitt 5.431 [easy:UnsetOpt_RTSP_Transport], Seite 252, für Details.

`#CURLOPT_SASL_IR`
: Deaktiviert das Senden der ersten Antwort im ersten Paket. Siehe Abschnitt 5.432 [easy:UnsetOpt_SASL_IR], Seite 253, für Details.

`#CURLOPT_SEEKFUNCTION`
: Setzt den Benutzer-Callback zum Suchen im Eingabedatenstrom zurück. Siehe Abschnitt 5.433 [easy:UnsetOpt_SeekFunction], Seite 253, für Details.

`#CURLOPT_SERVICE_NAME`
: Setzt den Namen des Authentifizierungsdienstes zurück. Siehe Abschnitt 5.434 [easy:UnsetOpt_Service_Name], Seite 253, für Details.

`#CURLOPT_SHARE`
: Setzt den Share-Handle zurück. Siehe Abschnitt 5.435 [easy:UnsetOpt_Share], Seite 253, für Details.

`#CURLOPT_SOCKS5_AUTH`
: Setzt die zulässigen Methoden für die SOCKS5-Proxyauthentifizierung zurück. Siehe Abschnitt 5.436 [easy:UnsetOpt_Socks5_Auth], Seite 254, für Details.

`#CURLOPT_SOCKS5_GSSAPI_NEC`
: Setzt den Socks Proxy gssapi Übertragungsschutz zurück. Siehe Abschnitt 5.437 [easy:UnsetOpt_Socks5_GSSAPI_NEC], Seite 254, für Details.

`#CURLOPT_SOCKS5_GSSAPI_SERVICE`
: Setzt den SOCKS5-Name des Proxy-Authentifizierungsdienstes zurück. Siehe Abschnitt 5.438 [easy:UnsetOpt_Socks5_GSSAPI_Service], Seite 254, für Details.

`#CURLOPT_SSH_AUTH_TYPES`
: Setzt den gewünschten Authentifizierungstypen für SFTP und SCP zurück. Siehe Abschnitt 5.439 [easy:UnsetOpt_SSH_Auth_Types], Seite 255, für Details.

`#CURLOPT_SSH_HOST_PUBLIC_KEY_MD5`
: Setzt die Prüfsumme des öffentlichen Schlüssels des SSH-Servers zurück. Siehe Abschnitt 5.440 [easy:UnsetOpt_SSH_Host_Public_Key_MD5], Seite 255, für Details.

`#CURLOPT_SSH_KNOWNHOSTS`
: Setzt den Dateiname mit den bekannten SSH-Hosts zurück. Siehe Abschnitt 5.441 [`easy:UnsetOpt_SSH_KnownHosts`], Seite 255, für Details.

`#CURLOPT_SSH_PRIVATE_KEYFILE`
: Setzt die private Schlüsseldatei für SSH-Authentifizierung zurück. Siehe Abschnitt 5.442 [`easy:UnsetOpt_SSH_Private_KeyFile`], Seite 256, für Details.

`#CURLOPT_SSH_PUBLIC_KEYFILE`
: Setzt die öffentliche Schlüsseldatei für die SSH-Authentifizierung zurück. Siehe Abschnitt 5.443 [`easy:UnsetOpt_SSH_Public_KeyFile`], Seite 256, für Details.

`#CURLOPT_SSL_CIPHER_LIST`
: Setzt die Verschlüsselung zurück, die für TLS verwendet werden soll. Siehe Abschnitt 5.444 [`easy:UnsetOpt_SSL_Cipher_List`], Seite 256, für Details.

`#CURLOPT_SSL_ENABLE_ALPN`
: Deaktiviert ALPN. Siehe Abschnitt 5.445 [`easy:UnsetOpt_SSL_Enable_Alpn`], Seite 257, für Details.

`#CURLOPT_SSL_ENABLE_NPN`
: Deaktiviert NPN. Siehe Abschnitt 5.446 [`easy:UnsetOpt_SSL_Enable_Npn`], Seite 257, für Details.

`#CURLOPT_SSL_FALSESTART`
: Deaktiviert TLS-Fehlstart. Siehe Abschnitt 5.447 [`easy:UnsetOpt_SSL_FalseStart`], Seite 257, für Details.

`#CURLOPT_SSL_OPTIONS`
: Setzt SSL-Verhaltensoptionen zurück. Siehe Abschnitt 5.448 [`easy:UnsetOpt_SSL_Options`], Seite 257, für Details.

`#CURLOPT_SSL_SESSIONID_CACHE`
: Aktiviert wieder die Verwendung des SSL-Sitzungs-ID-Cache. Siehe Abschnitt 5.449 [`easy:UnsetOpt_SSL_SessionID_Cache`], Seite 258, für Details.

`#CURLOPT_SSL_VERIFYHOST`
: Setzt die Überprüfung des Zertifikatsnamen auf den Standard zurück. Siehe Abschnitt 5.450 [`easy:UnsetOpt_SSL_VerifyHost`], Seite 258, für Details.

`#CURLOPT_SSL_VERIFYPEER`
: Deaktiviert die Überprüfung das SSL-Zertifikat des Peers. Siehe Abschnitt 5.451 [`easy:UnsetOpt_SSL_VerifyPeer`], Seite 258, für Details.

`#CURLOPT_SSL_VERIFYSTATUS`
: Überprüft den Status des Zertifikats nicht mehr. Siehe Abschnitt 5.452 [`easy:UnsetOpt_SSL_VerifyStatus`], Seite 259, für Details.

`#CURLOPT_SSLCERT`
: Setzt das SSL-Client-Zertifikat auf Standard zurück. Siehe Abschnitt 5.453 [`easy:UnsetOpt_SSLCert`], Seite 259, für Details.

`#CURLOPT_SSLCERTTYPE`
: Setzt den Typ des Client-SSL-Zertifikats zurück. Siehe Abschnitt 5.454 [easy:UnsetOpt_SSLCertType], Seite 259, für Details.

`#CURLOPT_SSLENGINE`
: Setzt die SSL System ID zurück. Siehe Abschnitt 5.455 [easy:UnsetOpt_SSLEngine], Seite 259, für Details.

`#CURLOPT_SSLENGINE_DEFAULT`
: Setzt das SSL-System zurück. Siehe Abschnitt 5.456 [easy:UnsetOpt_SSLEngine_Default], Seite 260, für Details.

`#CURLOPT_SSLKEY`
: Setzt die private Schlüsseldatei für TLS- und SSL-Client-Zertifikate zurück. Siehe Abschnitt 5.457 [easy:UnsetOpt_SSLKey], Seite 260, für Details.

`#CURLOPT_SSLKEYTYPE`
: Setzt den Typ der privaten Schlüsseldatei zurück. Siehe Abschnitt 5.458 [easy:UnsetOpt_SSLKeyType], Seite 260, für Details.

`#CURLOPT_SSLVERSION`
: Stellt die bevorzugte TLS/SSL-Version zurück. Siehe Abschnitt 5.459 [easy:UnsetOpt_SSLVersion], Seite 261, für Details.

`#CURLOPT_STREAM_DEPENDS`
: Stellt den Stream zurück, von dem diese Übertragung abhängt. Siehe Abschnitt 5.460 [easy:UnsetOpt_Stream_Depends], Seite 261, für Details.

`#CURLOPT_STREAM_DEPENDS_E`
: Stellt den Stream zurück, von dem diese Übertragung ausschließlich abhängt. Siehe Abschnitt 5.461 [easy:UnsetOpt_Stream_Depends_e], Seite 261, für Details.

`#CURLOPT_STREAM_WEIGHT`
: Setzt die Gewichtung des numerischen Datenstroms zurück. Siehe Abschnitt 5.462 [easy:UnsetOpt_Stream_Weight], Seite 261, für Details.

`#CURLOPT_SUPPRESS_CONNECT_HEADERS`
: Unterdrückt Proxy-CONNECT-Antwort-Header von Benutzer-Callbacks nicht mehr. Siehe Abschnitt 5.463 [easy:UnsetOpt_Suppress_Connect_Headers], Seite 262, für Details.

`#CURLOPT_TCP_FASTOPEN`
: Deaktiviert TCP Fast Open. Siehe Abschnitt 5.464 [easy:UnsetOpt_TCP_FastOpen], Seite 262, für Details.

`#CURLOPT_TCP_KEEPALIVE`
: Deaktiviert die TCP-Keep-Alive-Tests. Siehe Abschnitt 5.465 [easy:UnsetOpt_TCP_KeepAlive], Seite 262, für Details.

`#CURLOPT_TCP_KEEPIDLE`
: Setzt die TCP-Keep-Alive Leerlaufzeit zurück. Siehe Abschnitt 5.466 [easy:UnsetOpt_TCP_KeepIdle], Seite 263, für Details.

`#CURLOPT_TCP_KEEPINTVL`
: Setzt den TCP-Keep-Alive-Intervall zurück. Siehe Abschnitt 5.467 [`easy:UnsetOpt_TCP_KeepIntvl`], Seite 263, für Details.

`#CURLOPT_TCP_NODELAY`
: Aktiviert die Option TCP_NODELAY. Siehe Abschnitt 5.468 [`easy:UnsetOpt_TCP_NoDelay`], Seite 263, für Details.

`#CURLOPT_TELNETOPTIONS`
: Setzt die benutzerdefinierten Telnet-Optionen zurück. Siehe Abschnitt 5.469 [`easy:UnsetOpt_TelnetOptions`], Seite 264, für Details.

`#CURLOPT_TFTP_BLKSIZE`
: Setzt die TFTP-Blockgröße auf 512 Byte zurück. Siehe Abschnitt 5.470 [`easy:UnsetOpt_TFTP_BlkSize`], Seite 264, für Details.

`#CURLOPT_TFTP_NO_OPTIONS`
: Sendet wieder TFTP-Optionsanforderungen. Siehe Abschnitt 5.471 [`easy:UnsetOpt_TFTP_No_Options`], Seite 264, für Details.

`#CURLOPT_TIMECONDITION`
: Setzt die Bedingung für eine Zeitanfrage zurück. Siehe Abschnitt 5.472 [`easy:UnsetOpt_TimeCondition`], Seite 264, für Details.

`#CURLOPT_TIMEOUT`
: Setzt die maximale Zeit zurück, die die Anforderung dauern darf. Siehe Abschnitt 5.473 [`easy:UnsetOpt_Timeout`], Seite 265, für Details.

`#CURLOPT_TIMEOUT_MS`
: Setzt die maximale Zeit zurück, die die Anforderung dauern darf. Siehe Abschnitt 5.474 [`easy:UnsetOpt_Timeout_MS`], Seite 265, für Details.

`#CURLOPT_TIMEVALUE`
: Setzt den Zeitwert für bedingtes Verhalten zurück. Siehe Abschnitt 5.475 [`easy:UnsetOpt_TimeValue`], Seite 265, für Details.

`#CURLOPT_TLSAUTH_PASSWORD`
: Setzt das Passwort für die TLS-Authentifizierung zurück. Siehe Abschnitt 5.476 [`easy:UnsetOpt_TLSAuth_Password`], Seite 266, für Details.

`#CURLOPT_TLSAUTH_TYPE`
: Setzt die TLS-Authentifizierungsmethoden zurück. Siehe Abschnitt 5.477 [`easy:UnsetOpt_TLSAuth_Type`], Seite 266, für Details.

`#CURLOPT_TLSAUTH_USERNAME`
: Setzt den Benutzernamen zurück, der für die TLS-Authentifizierung verwendet wird. Siehe Abschnitt 5.478 [`easy:UnsetOpt_TLSAuth_UserName`], Seite 266, für Details.

`#CURLOPT_TRANSFER_ENCODING`
: Fordert die Übertragungscodierung nicht mehr an. Siehe Abschnitt 5.479 [`easy:UnsetOpt_Transfer_Encoding`], Seite 266, für Details.

`#CURLOPT_TRANSFERTEXT`
: Fordet wieder eine Binärübertragung für FTP an. Siehe Abschnitt 5.480 [easy:UnsetOpt_TransferText], Seite 267, für Details.

`#CURLOPT_UNIX_SOCKET_PATH`
: Setzt den Unix Domain Socket zurück. Siehe Abschnitt 5.481 [easy:UnsetOpt_Unix_Socket_Path], Seite 267, für Details.

`#CURLOPT_UNRESTRICTED_AUTH`
: Sendet wieder die Authentifizierungsdaten nur an den ursprünglichen Hostnamen. Siehe Abschnitt 5.482 [easy:UnsetOpt_Unrestricted_Auth], Seite 267, für Details.

`#CURLOPT_UPLOAD`
: Deaktiviert das Hochladen von Daten. Siehe Abschnitt 5.483 [easy:UnsetOpt_Upload], Seite 268, für Details.

`#CURLOPT_URL`
: Setzt die URL zurück, die in der Anfrage verwendet werden soll. Siehe Abschnitt 5.484 [easy:UnsetOpt_URL], Seite 268, für Details.

`#CURLOPT_USE_SSL`
: Fordert für die Übertragung kein SSL/TLS mehr an. Siehe Abschnitt 5.485 [easy:UnsetOpt_Use_SSL], Seite 268, für Details.

`#CURLOPT_USERAGENT`
: Setzt den HTTP-User-Agent-Header zurück. Siehe Abschnitt 5.486 [easy:UnsetOpt_UserAgent], Seite 268, für Details.

`#CURLOPT_USERNAME`
: Setzt den Benutzername für die Authentifizierung zurück. Siehe Abschnitt 5.487 [easy:UnsetOpt_UserName], Seite 269, für Details.

`#CURLOPT_USERPWD`
: Setzt den Benutzername und das Passwort für die Authentifizierung zurück. Siehe Abschnitt 5.488 [easy:UnsetOpt_UserPwd], Seite 269, für Details.

`#CURLOPT_VERBOSE`
: Schaltet den ausführlichen Modus aus. Siehe Abschnitt 5.489 [easy:UnsetOpt_Verbose], Seite 269, für Details.

`#CURLOPT_WILDCARDMATCH`
: Deaktiviert die Übertragung von Verzeichnis-Platzhaltern. Siehe Abschnitt 5.490 [easy:UnsetOpt_WildcardMatch], Seite 270, für Details.

`#CURLOPT_WRITEFUNCTION`
: Setzt den Callback zum Schreiben empfangener Daten zurück. Siehe Abschnitt 5.491 [easy:UnsetOpt_WriteFunction], Seite 270, für Details.

`#CURLOPT_XOAUTH2_BEARER`
: Setzt den OAuth 2.0 Access Token zurück. Siehe Abschnitt 5.492 [easy:UnsetOpt_XOAuth2_Bearer], Seite 270, für Details.

EINGABEN

`option` Optionstyp zum Aufheben der Einstellung

BEISPIEL

```
e:UnsetOpt(#CURLOPT_URL)
e:UnsetOpt(#CURLOPT_VERBOSE)
e:UnsetOpt(#CURLOPT_FOLLOWLOCATION)
```

Der obige Code setzt einige Optionen auf einfache Weise außer Kraft, d.h. er setzt diese Optionen auf ihre Standardwerte zurück.

5.278 easy:UnsetOpt_Abstract_Unix_Socket

BEZEICHNUNG

easy:UnsetOpt_Abstract_Unix_Socket – setzt einen abstrakten Unix-Domänen-Socket zurück

ÜBERSICHT

```
easy:UnsetOpt_Abstract_Unix_Socket()
```

BESCHREIBUNG

Siehe Abschnitt 5.61 [easy:SetOpt_Abstract_Unix_Socket], Seite 64, für Details.

EINGABEN

keine

5.279 easy:UnsetOpt_Accept_Encoding

BEZEICHNUNG

easy:UnsetOpt_Accept_Encoding – setzt die automatische Dekompression von HTTP-Downloads zurück

ÜBERSICHT

```
easy:UnsetOpt_Accept_Encoding()
```

BESCHREIBUNG

Siehe Abschnitt 5.62 [easy:SetOpt_Accept_Encoding], Seite 65, für Details.

EINGABEN

keine

5.280 easy:UnsetOpt_AcceptTimeout_MS

BEZEICHNUNG

easy:UnsetOpt_AcceptTimeout_MS – setzt die Zeitüberschreitung beim Warten auf die erneute Verbindung des FTP-Servers zurück

ÜBERSICHT

```
easy:UnsetOpt_AcceptTimeout_MS()
```

BESCHREIBUNG

Siehe Abschnitt 5.63 [easy:SetOpt_AcceptTimeout_MS], Seite 66, für Details.

EINGABEN
keine

5.281 easy:UnsetOpt_Address_Scope

BEZEICHNUNG
easy:UnsetOpt_Address_Scope – setzt den Bereich für lokale IPv6-Adressen zurück

ÜBERSICHT
`easy:UnsetOpt_Address_Scope()`

BESCHREIBUNG
Siehe Abschnitt 5.64 [`easy:SetOpt_Address_Scope`], Seite 66, für Details.

EINGABEN
keine

5.282 easy:UnsetOpt_Append

BEZEICHNUNG
easy:UnsetOpt_Append – dektiviert das Anhängen an die Remote-Datei

ÜBERSICHT
`easy:UnsetOpt_Append()`

BESCHREIBUNG
Siehe Abschnitt 5.65 [`easy:SetOpt_Append`], Seite 66, für Details.

EINGABEN
keine

5.283 easy:UnsetOpt_AutoReferer

BEZEICHNUNG
easy:UnsetOpt_AutoReferer – dektiviert die automatische Aktualisierung des Referer-Header

ÜBERSICHT
`easy:UnsetOpt_AutoReferer()`

BESCHREIBUNG
Siehe Abschnitt 5.66 [`easy:SetOpt_AutoReferer`], Seite 67, für Details.

EINGABEN
keine

5.284 easy:UnsetOpt_BufferSize

BEZEICHNUNG
easy:UnsetOpt_BufferSize – stellt die bevorzugte Empfangspuffergröße zurück

ÜBERSICHT
`easy:UnsetOpt_BufferSize()`

BESCHREIBUNG
Siehe Abschnitt 5.67 [easy:SetOpt_BufferSize], Seite 67, für Details.

EINGABEN
keine

5.285 easy:UnsetOpt_CAInfo

BEZEICHNUNG
easy:UnsetOpt_CAInfo – setzt den Pfad zum Paket der Zertifizierungsstelle (CA) zurück

ÜBERSICHT
`easy:UnsetOpt_CAInfo()`

BESCHREIBUNG
Siehe Abschnitt 5.68 [easy:SetOpt_CAInfo], Seite 67, für Details.

EINGABEN
keine

5.286 easy:UnsetOpt_CAPath

BEZEICHNUNG
easy:UnsetOpt_CAPath – setzt das Verzeichnis mit CA-Zertifikaten zurück

ÜBERSICHT
`easy:UnsetOpt_CAPath()`

BESCHREIBUNG
Siehe Abschnitt 5.69 [easy:SetOpt_CAPath], Seite 68, für Details.

EINGABEN
keine

5.287 easy:UnsetOpt_CertInfo

BEZEICHNUNG
easy:UnsetOpt_CertInfo – deaktiviert die Anforderung von SSL-Zertifikat-Informationen

ÜBERSICHT
`easy:UnsetOpt_CertInfo()`

BESCHREIBUNG
Siehe Abschnitt 5.70 [easy:SetOpt_CertInfo], Seite 69, für Details.

EINGABEN
keine

5.288 easy:UnsetOpt_Chunk_BGN_Function

BEZEICHNUNG
easy:UnsetOpt_Chunk_BGN_Function – deaktiviert den Callback vor einer Übertragung mit FTP Platzhalter Übereinstimmung

ÜBERSICHT
```
easy:UnsetOpt_Chunk_BGN_Function()
```

BESCHREIBUNG
Siehe Abschnitt 5.71 [easy:SetOpt_Chunk_BGN_Function], Seite 69, für Details.

EINGABEN
keine

5.289 easy:UnsetOpt_Chunk_End_Function

BEZEICHNUNG
easy:UnsetOpt_Chunk_End_Function – deaktiviert den Callback nach einer Übertragung mit FTP Platzhalter Übereinstimmung

ÜBERSICHT
```
easy:UnsetOpt_Chunk_End_Function()
```

BESCHREIBUNG
Siehe Abschnitt 5.72 [easy:SetOpt_Chunk_End_Function], Seite 70, für Details.

EINGABEN
keine

5.290 easy:UnsetOpt_Connect_Only

BEZEICHNUNG
easy:UnsetOpt_Connect_Only – stoppt nicht mehr, wenn eine Verbindung zum Zielserver besteht

ÜBERSICHT
```
easy:UnsetOpt_Connect_Only()
```

BESCHREIBUNG
Siehe Abschnitt 5.73 [easy:SetOpt_Connect_Only], Seite 71, für Details.

EINGABEN
keine

5.291 easy:UnsetOpt_Connect_To

BEZEICHNUNG
easy:UnsetOpt_Connect_To – verbindet sich wieder mit dem Host und Port der URL

ÜBERSICHT
`easy:UnsetOpt_Connect_To()`

BESCHREIBUNG
Siehe Abschnitt 5.74 [easy:SetOpt_Connect_To], Seite 71, für Details.

EINGABEN
keine

5.292 easy:UnsetOpt_ConnectTimeout

BEZEICHNUNG
easy:UnsetOpt_ConnectTimeout – setzt die Zeitüberschreitung für die Verbindungsphase zurück

ÜBERSICHT
`easy:UnsetOpt_ConnectTimeout()`

BESCHREIBUNG
Siehe Abschnitt 5.75 [easy:SetOpt_ConnectTimeout], Seite 72, für Details.

EINGABEN
keine

5.293 easy:UnsetOpt_ConnectTimeout_MS

BEZEICHNUNG
easy:UnsetOpt_ConnectTimeout_MS – setzt die Zeitüberschreitung für die Verbindungsphase zurück

ÜBERSICHT
`easy:UnsetOpt_ConnectTimeout_MS()`

BESCHREIBUNG
Siehe Abschnitt 5.76 [easy:SetOpt_ConnectTimeout_MS], Seite 73, für Details.

EINGABEN
keine

5.294 easy:UnsetOpt_Cookie

BEZEICHNUNG
easy:UnsetOpt_Cookie – setzt den Inhalt des HTTP-Cookie-Headers zurück

ÜBERSICHT
`easy:UnsetOpt_Cookie()`

BESCHREIBUNG
Siehe Abschnitt 5.77 [`easy:SetOpt_Cookie`], Seite 73, für Details.

EINGABEN
keine

5.295 easy:UnsetOpt_CookieFile

BEZEICHNUNG
easy:UnsetOpt_CookieFile – setzt den Cookie-Dateinamen zurück

ÜBERSICHT
`easy:UnsetOpt_CookieFile()`

BESCHREIBUNG
Siehe Abschnitt 5.78 [`easy:SetOpt_CookieFile`], Seite 74, für Details.

EINGABEN
keine

5.296 easy:UnsetOpt_CookieJar

BEZEICHNUNG
easy:UnsetOpt_CookieJar – setzt den Cookie-Dateinamen zum Speichern zurück

ÜBERSICHT
`easy:UnsetOpt_CookieJar()`

BESCHREIBUNG
Siehe Abschnitt 5.79 [`easy:SetOpt_CookieJar`], Seite 75, für Details.

EINGABEN
keine

5.297 easy:UnsetOpt_CookieList

BEZEICHNUNG
easy:UnsetOpt_CookieList – setzt den internen Cookie-Speicher zurück

ÜBERSICHT
`easy:UnsetOpt_CookieList()`

BESCHREIBUNG
Siehe Abschnitt 5.80 [`easy:SetOpt_CookieList`], Seite 75, für Details.

EINGABEN
keine

5.298 easy:UnsetOpt_CookieSession

BEZEICHNUNG
easy:UnsetOpt_CookieSession – beendet eine neue Cookie-Sitzung

ÜBERSICHT
`easy:UnsetOpt_CookieSession()`

BESCHREIBUNG
Siehe Abschnitt 5.81 [`easy:SetOpt_CookieSession`], Seite 76, für Details.

EINGABEN
keine

5.299 easy:UnsetOpt_CRLF

BEZEICHNUNG
easy:UnsetOpt_CRLF – deaktiviert die CRLF-Konvertierung

ÜBERSICHT
`easy:UnsetOpt_CRLF()`

BESCHREIBUNG
Siehe Abschnitt 5.82 [`easy:SetOpt_CRLF`], Seite 77, für Details.

EINGABEN
keine

5.300 easy:UnsetOpt_CRLFile

BEZEICHNUNG
easy:UnsetOpt_CRLFile – setzt die Datei für Zertifikatssperrlisten zurück

ÜBERSICHT
`easy:UnsetOpt_CRLFile()`

BESCHREIBUNG
Siehe Abschnitt 5.83 [`easy:SetOpt_CRLFile`], Seite 77, für Details.

EINGABEN
keine

5.301 easy:UnsetOpt_CustomRequest

BEZEICHNUNG
easy:UnsetOpt_CustomRequest – stellt den internen Standard für die Anforderung wieder her

ÜBERSICHT
`easy:UnsetOpt_CustomRequest()`

BESCHREIBUNG
Siehe Abschnitt 5.84 [easy:SetOpt_CustomRequest], Seite 78, für Details.

EINGABEN
keine

5.302 easy:UnsetOpt_DebugFunction

BEZEICHNUNG
easy:UnsetOpt_DebugFunction – setzt wieder die Standard-Debug-Funktion ein

ÜBERSICHT
```
easy:UnsetOpt_DebugFunction()
```

BESCHREIBUNG
Siehe Abschnitt 5.85 [easy:SetOpt_DebugFunction], Seite 79, für Details.

EINGABEN
keine

5.303 easy:UnsetOpt_Default_Protocol

BEZEICHNUNG
easy:UnsetOpt_Default_Protocol – setzt das Standardprotokoll bei fehlendem Schemanamen zurück

ÜBERSICHT
```
easy:UnsetOpt_Default_Protocol()
```

BESCHREIBUNG
Siehe Abschnitt 5.86 [easy:SetOpt_Default_Protocol], Seite 80, für Details.

EINGABEN
keine

5.304 easy:UnsetOpt_DirListOnly

BEZEICHNUNG
easy:UnsetOpt_DirListOnly – fragt wieder nach der gesammten Verzeichnisliste

ÜBERSICHT
```
easy:UnsetOpt_DirListOnly()
```

BESCHREIBUNG
Siehe Abschnitt 5.87 [easy:SetOpt_DirListOnly], Seite 81, für Details.

EINGABEN
keine

5.305 easy:UnsetOpt_DNS_Cache_Timeout

BEZEICHNUNG
easy:UnsetOpt_DNS_Cache_Timeout – setzt die Lebensdauer für DNS-Cache-Einträge wieder zurück

ÜBERSICHT
`easy:UnsetOpt_DNS_Cache_Timeout()`

BESCHREIBUNG
Siehe Abschnitt 5.88 [easy:SetOpt_DNS_Cache_Timeout], Seite 82, für Details.

EINGABEN
keine

5.306 easy:UnsetOpt_DNS_Interface

BEZEICHNUNG
easy:UnsetOpt_DNS_Interface – bindet nicht mehr an eine bestimmte Schnittstelle

ÜBERSICHT
`easy:UnsetOpt_DNS_Interface()`

BESCHREIBUNG
Siehe Abschnitt 5.89 [easy:SetOpt_DNS_Interface], Seite 82, für Details.

EINGABEN
keine

5.307 easy:UnsetOpt_DNS_Local_IP4

BEZEICHNUNG
easy:UnsetOpt_DNS_Local_IP4 – bindet nicht mehr an eine bestimmte IP-Adresse

ÜBERSICHT
`easy:UnsetOpt_DNS_Local_IP4()`

BESCHREIBUNG
Siehe Abschnitt 5.90 [easy:SetOpt_DNS_Local_IP4], Seite 83, für Details.

EINGABEN
keine

5.308 easy:UnsetOpt_DNS_Local_IP6

BEZEICHNUNG
easy:UnsetOpt_DNS_Local_IP6 – bindet nicht mehr an eine bestimmte IP-Adresse

ÜBERSICHT
`easy:UnsetOpt_DNS_Local_IP6()`

BESCHREIBUNG
Siehe Abschnitt 5.91 [easy:SetOpt_DNS_Local_IP6], Seite 83, für Details.

EINGABEN
keine

5.309 easy:UnsetOpt_DNS_Servers

BEZEICHNUNG
easy:UnsetOpt_DNS_Servers – setzt die DNS-Server-Liste auf die Systemvoreinstellung zurück

ÜBERSICHT
```
easy:UnsetOpt_DNS_Servers()
```

BESCHREIBUNG
Siehe Abschnitt 5.92 [easy:SetOpt_DNS_Servers], Seite 83, für Details.

EINGABEN
keine

5.310 easy:UnsetOpt_DNS_Use_Global_Cache

BEZEICHNUNG
easy:UnsetOpt_DNS_Use_Global_Cache – deaktiviert den globalen DNS-Cache

ÜBERSICHT
```
easy:UnsetOpt_DNS_Use_Global_Cache()
```

BESCHREIBUNG
Siehe Abschnitt 5.93 [easy:SetOpt_DNS_Use_Global_Cache], Seite 84, für Details.

EINGABEN
keine

5.311 easy:UnsetOpt_EGDSocket

BEZEICHNUNG
easy:UnsetOpt_EGDSocket – setzt den EGD-Socketpfad zurück

ÜBERSICHT
```
easy:UnsetOpt_EGDSocket()
```

BESCHREIBUNG
Siehe Abschnitt 5.94 [easy:SetOpt_EGDSocket], Seite 84, für Details.

EINGABEN
keine

5.312 easy:UnsetOpt_Expect_100_Timeout_MS

BEZEICHNUNG
easy:UnsetOpt_Expect_100_Timeout_MS – setzt die Zeitüberschreitung bei der Antwort Expect: 100-continue zurück

ÜBERSICHT
`easy:UnsetOpt_Expect_100_Timeout_MS()`

BESCHREIBUNG
Siehe Abschnitt 5.95 [easy:SetOpt_Expect_100_Timeout_MS], Seite 84, für Details.

EINGABEN
keine

5.313 easy:UnsetOpt_FailOnError

BEZEICHNUNG
easy:UnsetOpt_FailOnError – setzt den Anforderungsfehler bei HTTP-Antwort >= 400 zurück

ÜBERSICHT
`easy:UnsetOpt_FailOnError()`

BESCHREIBUNG
Siehe Abschnitt 5.96 [easy:SetOpt_FailOnError], Seite 85, für Details.

EINGABEN
keine

5.314 easy:UnsetOpt_FileTime

BEZEICHNUNG
easy:UnsetOpt_FileTime – fordert keine Änderungszeit der Remote-Datenquelle an

ÜBERSICHT
`easy:UnsetOpt_FileTime()`

BESCHREIBUNG
Siehe Abschnitt 5.97 [easy:SetOpt_FileTime], Seite 85, für Details.

EINGABEN
keine

5.315 easy:UnsetOpt_FNMatch_Function

BEZEICHNUNG
easy:UnsetOpt_FNMatch_Function – setzt die Callback-Platzhalterabgleich-Funktion zurück

ÜBERSICHT
`easy:UnsetOpt_FNMatch_Function()`

BESCHREIBUNG
Siehe Abschnitt 5.98 [`easy:SetOpt_FNMatch_Function`], Seite 86, für Details.

EINGABEN
keine

5.316 easy:UnsetOpt_FollowLocation

BEZEICHNUNG
easy:UnsetOpt_FollowLocation – stellt wieder die HTTP 3xx Standortverfolgung ein

ÜBERSICHT
`easy:UnsetOpt_FollowLocation()`

BESCHREIBUNG
Siehe Abschnitt 5.99 [`easy:SetOpt_FollowLocation`], Seite 86, für Details.

EINGABEN
keine

5.317 easy:UnsetOpt_Forbid_Reuse

BEZEICHNUNG
easy:UnsetOpt_Forbid_Reuse – lässt die Verbindung stehen, nachdem die Übertragung beendet ist

ÜBERSICHT
`easy:UnsetOpt_Forbid_Reuse()`

BESCHREIBUNG
Siehe Abschnitt 5.100 [`easy:SetOpt_Forbid_Reuse`], Seite 87, für Details.

EINGABEN
keine

5.318 easy:UnsetOpt_Fresh_Connect

BEZEICHNUNG
easy:UnsetOpt_Fresh_Connect – erzwingt keine neue Verbindung mehr

ÜBERSICHT
`easy:UnsetOpt_Fresh_Connect()`

BESCHREIBUNG
Siehe Abschnitt 5.101 [`easy:SetOpt_Fresh_Connect`], Seite 87, für Details.

EINGABEN
keine

5.319 easy:UnsetOpt_FTP_Account

BEZEICHNUNG
easy:UnsetOpt_FTP_Account – sendet keine Kontoinformationen an den FTP-Server

ÜBERSICHT
`easy:UnsetOpt_FTP_Account()`

BESCHREIBUNG
Siehe Abschnitt 5.102 [`easy:SetOpt_FTP_Account`], Seite 88, für Details.

EINGABEN
keine

5.320 easy:UnsetOpt_FTP_Alternative_To_User

BEZEICHNUNG
easy:UnsetOpt_FTP_Alternative_To_User – FTP wird nicht mehr anstelle von USER verwendet

ÜBERSICHT
`easy:UnsetOpt_FTP_Alternative_To_User()`

BESCHREIBUNG
Siehe Abschnitt 5.103 [`easy:SetOpt_FTP_Alternative_To_User`], Seite 88, für Details.

EINGABEN
keine

5.321 easy:UnsetOpt_FTP_Create_Missing_Dirs

BEZEICHNUNG
easy:UnsetOpt_FTP_Create_Missing_Dirs – erstellt keine fehlende Verzeichnisse mehr für FTP und SFTP

ÜBERSICHT
`easy:UnsetOpt_FTP_Create_Missing_Dirs()`

BESCHREIBUNG
Siehe Abschnitt 5.104 [`easy:SetOpt_FTP_Create_Missing_Dirs`], Seite 89, für Details.

EINGABEN
keine

5.322 easy:UnsetOpt_FTP_FileMethod

BEZEICHNUNG
easy:UnsetOpt_FTP_FileMethod – stellt das Verzeichnisdurchlaufverfahren für FTP wieder auf Standard

ÜBERSICHT
`easy:UnsetOpt_FTP_FileMethod()`

BESCHREIBUNG
Siehe Abschnitt 5.105 [`easy:SetOpt_FTP_FileMethod`], Seite 89, für Details.

EINGABEN
keine

5.323 easy:UnsetOpt_FTP_Response_Timeout

BEZEICHNUNG
easy:UnsetOpt_FTP_Response_Timeout – setzt die FTP-Antwortzeit wieder zurück

ÜBERSICHT
`easy:UnsetOpt_FTP_Response_Timeout()`

BESCHREIBUNG
Siehe Abschnitt 5.106 [`easy:SetOpt_FTP_Response_Timeout`], Seite 90, für Details.

EINGABEN
keine

5.324 easy:UnsetOpt_FTP_Skip_PASV_IP

BEZEICHNUNG
easy:UnsetOpt_FTP_Skip_PASV_IP – verwendet wieder die IP-Adresse in der PASV-Antwort

ÜBERSICHT
`easy:UnsetOpt_FTP_Skip_PASV_IP()`

BESCHREIBUNG
Siehe Abschnitt 5.107 [`easy:SetOpt_FTP_Skip_PASV_IP`], Seite 90, für Details.

EINGABEN
keine

5.325 easy:UnsetOpt_FTP_SSL_CCC

BEZEICHNUNG
easy:UnsetOpt_FTP_SSL_CCC – lässt SSL mit FTP nach der Authentifizierung eingestellt

ÜBERSICHT
`easy:UnsetOpt_FTP_SSL_CCC()`

BESCHREIBUNG
Siehe Abschnitt 5.108 [`easy:SetOpt_FTP_SSL_CCC`], Seite 91, für Details.

EINGABEN
keine

5.326 easy:UnsetOpt_FTP_Use_Eprt

BEZEICHNUNG
easy:UnsetOpt_FTP_Use_Eprt – deaktiviert die Nutzung von EPRT mit FTP

ÜBERSICHT
`easy:UnsetOpt_FTP_Use_Eprt()`

BESCHREIBUNG
Siehe Abschnitt 5.109 [`easy:SetOpt_FTP_Use_Eprt`], Seite 91, für Details.

EINGABEN
keine

5.327 easy:UnsetOpt_FTP_Use_Epsv

BEZEICHNUNG
easy:UnsetOpt_FTP_Use_Epsv – deaktiviert die Nutzung von EPSV mit FTP

ÜBERSICHT
`easy:UnsetOpt_FTP_Use_Epsv()`

BESCHREIBUNG
Siehe Abschnitt 5.110 [`easy:SetOpt_FTP_Use_Epsv`], Seite 92, für Details.

EINGABEN
keine

5.328 easy:UnsetOpt_FTP_Use_Pret

BEZEICHNUNG
easy:UnsetOpt_FTP_Use_Pret – deaktiviert den PRET-Befehl mit FTP

ÜBERSICHT
`easy:UnsetOpt_FTP_Use_Pret()`

BESCHREIBUNG
Siehe Abschnitt 5.111 [`easy:SetOpt_FTP_Use_Pret`], Seite 92, für Details.

EINGABEN
keine

5.329 easy:UnsetOpt_FTPPort

BEZEICHNUNG
easy:UnsetOpt_FTPPort – deaktiviert die FTP-Übertragung

ÜBERSICHT
```
easy:UnsetOpt_FTPPort()
```

BESCHREIBUNG
Siehe Abschnitt 5.112 [easy:SetOpt_FTPPort], Seite 93, für Details.

EINGABEN
keine

5.330 easy:UnsetOpt_FTPSSLAuth

BEZEICHNUNG
easy:UnsetOpt_FTPSSLAuth – setzt die Reihenfolge von TLS/SSL zurück

ÜBERSICHT
```
easy:UnsetOpt_FTPSSLAuth()
```

BESCHREIBUNG
Siehe Abschnitt 5.113 [easy:SetOpt_FTPSSLAuth], Seite 93, für Details.

EINGABEN
keine

5.331 easy:UnsetOpt_GSSAPI_Delegation

BEZEICHNUNG
easy:UnsetOpt_GSSAPI_Delegation – deaktiviert die erlaubte Zuordnung von GSS-APIs

ÜBERSICHT
```
easy:UnsetOpt_GSSAPI_Delegation()
```

BESCHREIBUNG
Siehe Abschnitt 5.114 [easy:SetOpt_GSSAPI_Delegation], Seite 94, für Details.

EINGABEN
keine

5.332 easy:UnsetOpt_Header

BEZEICHNUNG
easy:UnsetOpt_Header – übergibt keinen Header an den Datenstrom

ÜBERSICHT
```
easy:UnsetOpt_Header()
```

BESCHREIBUNG
Siehe Abschnitt 5.115 [easy:SetOpt_Header], Seite 94, für Details.

EINGABEN
keine

5.333 easy:UnsetOpt_HeaderFunction

BEZEICHNUNG
easy:UnsetOpt_HeaderFunction – setzt die Callback-Funktion für Header-Daten zurück

ÜBERSICHT
```
easy:UnsetOpt_HeaderFunction()
```

BESCHREIBUNG
Siehe Abschnitt 5.116 [easy:SetOpt_HeaderFunction], Seite 95, für Details.

EINGABEN
keine

5.334 easy:UnsetOpt_HeaderOpt

BEZEICHNUNG
easy:UnsetOpt_HeaderOpt – setzt das Senden von HTTP-Header zurück

ÜBERSICHT
```
easy:UnsetOpt_HeaderOpt()
```

BESCHREIBUNG
Siehe Abschnitt 5.117 [easy:SetOpt_HeaderOpt], Seite 96, für Details.

EINGABEN
keine

5.335 easy:UnsetOpt_HTTP200Aliases

BEZEICHNUNG
easy:UnsetOpt_HTTP200Aliases – setzt alternative Übereinstimmungen für HTTP 200 OK zurück

ÜBERSICHT
```
easy:UnsetOpt_HTTP200Aliases()
```

BESCHREIBUNG
Siehe Abschnitt 5.118 [easy:SetOpt_HTTP200Aliases], Seite 97, für Details.

EINGABEN
keine

5.336 easy:UnsetOpt_HTTP_Content_Decoding

BEZEICHNUNG
easy:UnsetOpt_HTTP_Content_Decoding – deaktiviert die Dekodierung von HTTP-Inhalten

ÜBERSICHT
`easy:UnsetOpt_HTTP_Content_Decoding()`

BESCHREIBUNG
Siehe Abschnitt 5.119 [`easy:SetOpt_HTTP_Content_Decoding`], Seite 97, für Details.

EINGABEN
keine

5.337 easy:UnsetOpt_HTTP_Transfer_Decoding

BEZEICHNUNG
easy:UnsetOpt_HTTP_Transfer_Decoding – deaktiviert die Dekodierung der HTTP-Übertragung

ÜBERSICHT
`easy:UnsetOpt_HTTP_Transfer_Decoding()`

BESCHREIBUNG
Siehe Abschnitt 5.120 [`easy:SetOpt_HTTP_Transfer_Decoding`], Seite 98, für Details.

EINGABEN
keine

5.338 easy:UnsetOpt_HTTP_Version

BEZEICHNUNG
easy:UnsetOpt_HTTP_Version – setzt die zu verwendende HTTP-Protokollversion zurück

ÜBERSICHT
`easy:UnsetOpt_HTTP_Version()`

BESCHREIBUNG
Siehe Abschnitt 5.121 [`easy:SetOpt_HTTP_Version`], Seite 98, für Details.

EINGABEN
keine

5.339 easy:UnsetOpt_HTTPAuth

BEZEICHNUNG
easy:UnsetOpt_HTTPAuth – setzt die HTTP-Server-Authentifizierungsmethoden zurück

ÜBERSICHT
`easy:UnsetOpt_HTTPAuth()`

BESCHREIBUNG
Siehe Abschnitt 5.122 [easy:SetOpt_HTTPAuth], Seite 99, für Details.

EINGABEN
keine

5.340 easy:UnsetOpt_HTTPGet

BEZEICHNUNG
easy:UnsetOpt_HTTPGet – setzt die HTTP GET-Anfrage zurück

ÜBERSICHT
`easy:UnsetOpt_HTTPGet()`

BESCHREIBUNG
Siehe Abschnitt 5.123 [easy:SetOpt_HTTPGet], Seite 101, für Details.

EINGABEN
keine

5.341 easy:UnsetOpt_HTTPHeader

BEZEICHNUNG
easy:UnsetOpt_HTTPHeader – setzt den benutzerdefinierten HTTP-Header zurück

ÜBERSICHT
`easy:UnsetOpt_HTTPHeader()`

BESCHREIBUNG
Siehe Abschnitt 5.124 [easy:SetOpt_HTTPHeader], Seite 101, für Details.

EINGABEN
keine

5.342 easy:UnsetOpt_HTTPPost

BEZEICHNUNG
easy:UnsetOpt_HTTPPost – setzt den mehrteiligen Formpost-Inhalt zurück

ÜBERSICHT
`easy:UnsetOpt_HTTPPost()`

BESCHREIBUNG
Siehe Abschnitt 5.125 [easy:SetOpt_HTTPPost], Seite 102, für Details.

EINGABEN
keine

5.343 easy:UnsetOpt_HTTPProxyTunnel

BEZEICHNUNG
easy:UnsetOpt_HTTPProxyTunnel – hebt den Tunnel durch einen HTTP-Proxy auf

ÜBERSICHT
```
easy:UnsetOpt_HTTPProxyTunnel()
```

BESCHREIBUNG
Siehe Abschnitt 5.126 [easy:SetOpt_HTTPProxyTunnel], Seite 103, für Details.

EINGABEN
keine

5.344 easy:UnsetOpt_Ignore_Content_Length

BEZEICHNUNG
easy:UnsetOpt_Ignore_Content_Length – bezieht den Content-Length-Header wieder ein

ÜBERSICHT
```
easy:UnsetOpt_Ignore_Content_Length()
```

BESCHREIBUNG
Siehe Abschnitt 5.127 [easy:SetOpt_Ignore_Content_Length], Seite 103, für Details.

EINGABEN
keine

5.345 easy:UnsetOpt_InFileSize

BEZEICHNUNG
easy:UnsetOpt_InFileSize – setzt die Größe der zu sendenden Eingabedatei zurück

ÜBERSICHT
```
easy:UnsetOpt_InFileSize()
```

BESCHREIBUNG
Siehe Abschnitt 5.128 [easy:SetOpt_InFileSize], Seite 104, für Details.

EINGABEN
keine

5.346 easy:UnsetOpt_InFileSize_Large

BEZEICHNUNG
easy:UnsetOpt_InFileSize_Large – setzt die Größe der zu sendenden Eingabedatei zurück

ÜBERSICHT
```
easy:UnsetOpt_InFileSize_Large()
```

BESCHREIBUNG
Siehe Abschnitt 5.129 [easy:SetOpt_InFileSize_Large], Seite 104, für Details.

EINGABEN
keine

5.347 easy:UnsetOpt_Interface

BEZEICHNUNG
easy:UnsetOpt_Interface – setzt die Quellschnittstelle für ausgehenden Datenverkehr zurück

ÜBERSICHT
```
easy:UnsetOpt_Interface()
```

BESCHREIBUNG
Siehe Abschnitt 5.130 [easy:SetOpt_Interface], Seite 105, für Details.

EINGABEN
keine

5.348 easy:UnsetOpt_IPResolve

BEZEICHNUNG
easy:UnsetOpt_IPResolve – setzt die verwendete IP-Protokollversion zurück

ÜBERSICHT
```
easy:UnsetOpt_IPResolve()
```

BESCHREIBUNG
Siehe Abschnitt 5.131 [easy:SetOpt_IPResolve], Seite 105, für Details.

EINGABEN
keine

5.349 easy:UnsetOpt_IssuerCert

BEZEICHNUNG
easy:UnsetOpt_IssuerCert – setzt den Dateiname des SSL-Zertifikats des Ausstellers zurück

ÜBERSICHT
`easy:UnsetOpt_IssuerCert()`

BESCHREIBUNG
Siehe Abschnitt 5.132 [`easy:SetOpt_IssuerCert`], Seite 106, für Details.

EINGABEN
keine

5.350 easy:UnsetOpt_Keep_Sending_On_Error

BEZEICHNUNG
easy:UnsetOpt_Keep_Sending_On_Error – sendet mit einer frühen HTTP-Antwort >= 300 nicht mehr weiter

ÜBERSICHT
`easy:UnsetOpt_Keep_Sending_On_Error()`

BESCHREIBUNG
Siehe Abschnitt 5.133 [`easy:SetOpt_Keep_Sending_On_Error`], Seite 107, für Details.

EINGABEN
keine

5.351 easy:UnsetOpt_KeyPasswd

BEZEICHNUNG
easy:UnsetOpt_KeyPasswd – setzt die Passphrase des privaten Schlüssel zurück

ÜBERSICHT
`easy:UnsetOpt_KeyPasswd()`

BESCHREIBUNG
Siehe Abschnitt 5.134 [`easy:SetOpt_KeyPasswd`], Seite 107, für Details.

EINGABEN
keine

5.352 easy:UnsetOpt_KRBLevel

BEZEICHNUNG
easy:UnsetOpt_KRBLevel – deaktiviert die FTP-Kerberos-Sicherheitsstufe

ÜBERSICHT
`easy:UnsetOpt_KRBLevel()`

BESCHREIBUNG
Siehe Abschnitt 5.135 [`easy:SetOpt_KRBLevel`], Seite 107, für Details.

EINGABEN
keine

5.353 easy:UnsetOpt_LocalPort

BEZEICHNUNG
easy:UnsetOpt_LocalPort – setzt die lokale Portnummer für Socket zurück

ÜBERSICHT
`easy:UnsetOpt_LocalPort()`

BESCHREIBUNG
Siehe Abschnitt 5.136 [`easy:SetOpt_LocalPort`], Seite 108, für Details.

EINGABEN
keine

5.354 easy:UnsetOpt_LocalPortRange

BEZEICHNUNG
easy:UnsetOpt_LocalPortRange – setzt die Anzahl zusätzlicher lokaler Ports zum Testen zurück

ÜBERSICHT
`easy:UnsetOpt_LocalPortRange()`

BESCHREIBUNG
Siehe Abschnitt 5.137 [`easy:SetOpt_LocalPortRange`], Seite 108, für Details.

EINGABEN
keine

5.355 easy:UnsetOpt_Login_Options

BEZEICHNUNG
easy:UnsetOpt_Login_Options – setzt die Login-Optionen zurück

ÜBERSICHT
`easy:UnsetOpt_Login_Options()`

BESCHREIBUNG
Siehe Abschnitt 5.138 [`easy:SetOpt_Login_Options`], Seite 109, für Details.

EINGABEN
keine

5.356 easy:UnsetOpt_Low_Speed_Limit

BEZEICHNUNG
easy:UnsetOpt_Low_Speed_Limit – setzt die niedrige Geschwindigkeitsbegrenzung zurück

ÜBERSICHT
`easy:UnsetOpt_Low_Speed_Limit()`

BESCHREIBUNG
Siehe Abschnitt 5.139 [`easy:SetOpt_Low_Speed_Limit`], Seite 109, für Details.

EINGABEN
keine

5.357 easy:UnsetOpt_Low_Speed_Time

BEZEICHNUNG
easy:UnsetOpt_Low_Speed_Time – setzt das Zeitlimit für niedrige Geschwindigkeit zurück

ÜBERSICHT
`easy:UnsetOpt_Low_Speed_Time()`

BESCHREIBUNG
Siehe Abschnitt 5.140 [`easy:SetOpt_Low_Speed_Time`], Seite 110, für Details.

EINGABEN
keine

5.358 easy:UnsetOpt_Mail_Auth

BEZEICHNUNG
easy:UnsetOpt_Mail_Auth – setzt die SMTP-Authentifizierungsadresse zurück

ÜBERSICHT
`easy:UnsetOpt_Mail_Auth()`

BESCHREIBUNG
Siehe Abschnitt 5.141 [`easy:SetOpt_Mail_Auth`], Seite 110, für Details.

EINGABEN
keine

5.359 easy:UnsetOpt_Mail_From

BEZEICHNUNG
easy:UnsetOpt_Mail_From – setzt die SMTP-Absenderadresse zurück

ÜBERSICHT
`easy:UnsetOpt_Mail_From()`

BESCHREIBUNG
Siehe Abschnitt 5.142 [`easy:SetOpt_Mail_From`], Seite 110, für Details.

EINGABEN
keine

5.360 easy:UnsetOpt_Mail_RCPT

BEZEICHNUNG
easy:UnsetOpt_Mail_RCPT – setzt die Liste der SMTP-Mail-Empfänger zurück

ÜBERSICHT
```
easy:UnsetOpt_Mail_RCPT()
```

BESCHREIBUNG
Siehe Abschnitt 5.143 [easy:SetOpt_Mail_RCPT], Seite 111, für Details.

EINGABEN
keine

5.361 easy:UnsetOpt_Max_Recv_Speed_Large

BEZEICHNUNG
easy:UnsetOpt_Max_Recv_Speed_Large – setzt die Geschwindigkeitslimit für das Herunterladen von Daten zurück

ÜBERSICHT
```
easy:UnsetOpt_Max_Recv_Speed_Large()
```

BESCHREIBUNG
Siehe Abschnitt 5.144 [easy:SetOpt_Max_Recv_Speed_Large], Seite 111, für Details.

EINGABEN
keine

5.362 easy:UnsetOpt_Max_Send_Speed_Large

BEZEICHNUNG
easy:UnsetOpt_Max_Send_Speed_Large – setzt die Geschwindigkeitslimit für das Hochladen von Daten zurück

ÜBERSICHT
```
easy:UnsetOpt_Max_Send_Speed_Large()
```

BESCHREIBUNG
Siehe Abschnitt 5.145 [easy:SetOpt_Max_Send_Speed_Large], Seite 112, für Details.

EINGABEN
keine

5.363 easy:UnsetOpt_MaxConnects

BEZEICHNUNG
easy:UnsetOpt_MaxConnects – setzt die maximale Verbindungs-Cache-Größe zurück

ÜBERSICHT
`easy:UnsetOpt_MaxConnects()`

BESCHREIBUNG
Siehe Abschnitt 5.146 [`easy:SetOpt_MaxConnects`], Seite 112, für Details.

EINGABEN
keine

5.364 easy:UnsetOpt_MaxFileSize

BEZEICHNUNG
easy:UnsetOpt_MaxFileSize – setzt die maximal zulässige Dateigröße für das Herunterladen zurück

ÜBERSICHT
`easy:UnsetOpt_MaxFileSize()`

BESCHREIBUNG
Siehe Abschnitt 5.147 [`easy:SetOpt_MaxFileSize`], Seite 113, für Details.

EINGABEN
keine

5.365 easy:UnsetOpt_MaxFileSize_Large

BEZEICHNUNG
easy:UnsetOpt_MaxFileSize_Large – setzt die maximal zulässige Dateigröße für das Herunterladen zurück

ÜBERSICHT
`easy:UnsetOpt_MaxFileSize_Large()`

BESCHREIBUNG
Siehe Abschnitt 5.148 [`easy:SetOpt_MaxFileSize_Large`], Seite 113, für Details.

EINGABEN
keine

5.366 easy:UnsetOpt_MaxRedirs

BEZEICHNUNG
easy:UnsetOpt_MaxRedirs – setzt die maximale Anzahl von erlaubten Umleitungen zurück

ÜBERSICHT
`easy:UnsetOpt_MaxRedirs()`

BESCHREIBUNG
Siehe Abschnitt 5.149 [`easy:SetOpt_MaxRedirs`], Seite 114, für Details.

EINGABEN
keine

5.367 easy:UnsetOpt_Netrc

BEZEICHNUNG
easy:UnsetOpt_Netrc – ignoriert wieder die .netrc-Informationen

ÜBERSICHT
`easy:UnsetOpt_Netrc()`

BESCHREIBUNG
Siehe Abschnitt 5.150 [easy:SetOpt_Netrc], Seite 114, für Details.

EINGABEN
keine

5.368 easy:UnsetOpt_Netrc_File

BEZEICHNUNG
easy:UnsetOpt_Netrc_File – setzt den Dateiname zum Lesen von .netrc-Informationen zurück

ÜBERSICHT
`easy:UnsetOpt_Netrc_File()`

BESCHREIBUNG
Siehe Abschnitt 5.151 [easy:SetOpt_Netrc_File], Seite 115, für Details.

EINGABEN
keine

5.369 easy:UnsetOpt_New_Directory_Perms

BEZEICHNUNG
easy:UnsetOpt_New_Directory_Perms – setzt die Berechtigungen für neu erstellte Remote-Verzeichnisse zurück

ÜBERSICHT
`easy:UnsetOpt_New_Directory_Perms()`

BESCHREIBUNG
Siehe Abschnitt 5.152 [easy:SetOpt_New_Directory_Perms], Seite 115, für Details.

EINGABEN
keine

5.370 easy:UnsetOpt_New_File_Perms

BEZEICHNUNG
easy:UnsetOpt_New_File_Perms – setzt die Berechtigungen für neu erstellte Remote-Dateien zurück

ÜBERSICHT
`easy:UnsetOpt_New_File_Perms()`

BESCHREIBUNG
Siehe Abschnitt 5.153 [`easy:SetOpt_New_File_Perms`], Seite 116, für Details.

EINGABEN
keine

5.371 easy:UnsetOpt_Nobody

BEZEICHNUNG
easy:UnsetOpt_Nobody – führt die Anfrage wieder mit Body-Download durch

ÜBERSICHT
`easy:UnsetOpt_Nobody()`

BESCHREIBUNG
Siehe Abschnitt 5.154 [`easy:SetOpt_Nobody`], Seite 116, für Details.

EINGABEN
keine

5.372 easy:UnsetOpt_NoProgress

BEZEICHNUNG
easy:UnsetOpt_NoProgress – schaltet die Fortschrittsanzeige wieder ein

ÜBERSICHT
`easy:UnsetOpt_NoProgress()`

BESCHREIBUNG
Siehe Abschnitt 5.155 [`easy:SetOpt_NoProgress`], Seite 116, für Details.

EINGABEN
keine

5.373 easy:UnsetOpt_NoProxy

BEZEICHNUNG
easy:UnsetOpt_NoProxy – aktiviert wieder die Proxy-Nutzung für bestimmte Hosts

ÜBERSICHT
`easy:UnsetOpt_NoProxy()`

BESCHREIBUNG
Siehe Abschnitt 5.156 [easy:SetOpt_NoProxy], Seite 117, für Details.

EINGABEN
keine

5.374 easy:UnsetOpt_NoSignal

BEZEICHNUNG
easy:UnsetOpt_NoSignal – überspringt die gesamte Signalverarbeitung nicht mehr

ÜBERSICHT
```
easy:UnsetOpt_NoSignal()
```

BESCHREIBUNG
Siehe Abschnitt 5.157 [easy:SetOpt_NoSignal], Seite 117, für Details.

EINGABEN
keine

5.375 easy:UnsetOpt_Password

BEZEICHNUNG
easy:UnsetOpt_Password – setzt das Passwort zur Verwendung bei der Authentifizierung zurück

ÜBERSICHT
```
easy:UnsetOpt_Password()
```

BESCHREIBUNG
Siehe Abschnitt 5.158 [easy:SetOpt_Password], Seite 118, für Details.

EINGABEN
keine

5.376 easy:UnsetOpt_Path_As_Is

BEZEICHNUNG
easy:UnsetOpt_Path_As_Is – verwendet wieder Punkt-Punkt-Sequenzen

ÜBERSICHT
```
easy:UnsetOpt_Path_As_Is()
```

BESCHREIBUNG
Siehe Abschnitt 5.159 [easy:SetOpt_Path_As_Is], Seite 118, für Details.

EINGABEN
keine

5.377 easy:UnsetOpt_PinnedPublicKey

BEZEICHNUNG
easy:UnsetOpt_PinnedPublicKey – setzt das Public Key Pinning zurück

ÜBERSICHT
```
easy:UnsetOpt_PinnedPublicKey()
```

BESCHREIBUNG
Siehe Abschnitt 5.160 [easy:SetOpt_PinnedPublicKey], Seite 119, für Details.

EINGABEN
keine

5.378 easy:UnsetOpt_PipeWait

BEZEICHNUNG
easy:UnsetOpt_PipeWait – wartet nicht mehr auf Pipelining/Multiplexing

ÜBERSICHT
```
easy:UnsetOpt_PipeWait()
```

BESCHREIBUNG
Siehe Abschnitt 5.161 [easy:SetOpt_PipeWait], Seite 119, für Details.

EINGABEN
keine

5.379 easy:UnsetOpt_Port

BEZEICHNUNG
easy:UnsetOpt_Port – die URL legt wieder fest, welcher Port verwendet wird

ÜBERSICHT
```
easy:UnsetOpt_Port()
```

BESCHREIBUNG
Siehe Abschnitt 5.162 [easy:SetOpt_Port], Seite 120, für Details.

EINGABEN
keine

5.380 easy:UnsetOpt_Post

BEZEICHNUNG
easy:UnsetOpt_Post – fordert keinen HTTP-POST mehr an

ÜBERSICHT
```
easy:UnsetOpt_Post()
```

BESCHREIBUNG
Siehe Abschnitt 5.163 [easy:SetOpt_Post], Seite 121, für Details.

EINGABEN
keine

5.381 easy:UnsetOpt_PostFields

BEZEICHNUNG
easy:UnsetOpt_PostFields – setzt die Daten zurück, welche an den Server gesendet werden

ÜBERSICHT
```
easy:UnsetOpt_PostFields()
```

BESCHREIBUNG
Siehe Abschnitt 5.164 [easy:SetOpt_PostFields], Seite 121, für Details.

EINGABEN
keine

5.382 easy:UnsetOpt_PostQuote

BEZEICHNUNG
easy:UnsetOpt_PostQuote – setzt die (S)FTP-Befehle zur Ausführung nach der Übertragung zurück

ÜBERSICHT
```
easy:UnsetOpt_PostQuote()
```

BESCHREIBUNG
Siehe Abschnitt 5.165 [easy:SetOpt_PostQuote], Seite 122, für Details.

EINGABEN
keine

5.383 easy:UnsetOpt_PostRedir

BEZEICHNUNG
easy:UnsetOpt_PostRedir – setzt die Vorgehensweise bei einer HTTP-POST-Umleitung zurück

ÜBERSICHT
```
easy:UnsetOpt_PostRedir()
```

BESCHREIBUNG
Siehe Abschnitt 5.166 [easy:SetOpt_PostRedir], Seite 123, für Details.

EINGABEN
keine

5.384 easy:UnsetOpt_Pre_Proxy

BEZEICHNUNG
easy:UnsetOpt_Pre_Proxy – setzt den Prä-Proxy für die Verwendung zurück

ÜBERSICHT
`easy:UnsetOpt_Pre_Proxy()`

BESCHREIBUNG
Siehe Abschnitt 5.167 [`easy:SetOpt_Pre_Proxy`], Seite 123, für Details.

EINGABEN
keine

5.385 easy:UnsetOpt_Prequote

BEZEICHNUNG
easy:UnsetOpt_Prequote – setzt die Befehle zurück, die vor einer FTP-Übertragung ausgeführt werden sollen

ÜBERSICHT
`easy:UnsetOpt_Prequote()`

BESCHREIBUNG
Siehe Abschnitt 5.168 [`easy:SetOpt_Prequote`], Seite 124, für Details.

EINGABEN
keine

5.386 easy:UnsetOpt_ProgressFunction

BEZEICHNUNG
easy:UnsetOpt_ProgressFunction – setzt den Callback zur Fortschrittsanzeige-Funktion zurück

ÜBERSICHT
`easy:UnsetOpt_ProgressFunction()`

BESCHREIBUNG
Siehe Abschnitt 5.169 [`easy:SetOpt_ProgressFunction`], Seite 124, für Details.

EINGABEN
keine

5.387 easy:UnsetOpt_Protocols

BEZEICHNUNG
easy:UnsetOpt_Protocols – erlaubt wieder alle Protokolle zu verwenden

ÜBERSICHT
`easy:UnsetOpt_Protocols()`

BESCHREIBUNG
Siehe Abschnitt 5.170 [`easy:SetOpt_Protocols`], Seite 125, für Details.

EINGABEN
keine

5.388 easy:UnsetOpt_Proxy

BEZEICHNUNG
easy:UnsetOpt_Proxy – setzt den Proxy für die Verwendung zurück

ÜBERSICHT
`easy:UnsetOpt_Proxy()`

BESCHREIBUNG
Siehe Abschnitt 5.171 [`easy:SetOpt_Proxy`], Seite 126, für Details.

EINGABEN
keine

5.389 easy:UnsetOpt_Proxy_CAInfo

BEZEICHNUNG
easy:UnsetOpt_Proxy_CAInfo – setzt den Pfad zum Proxy Certificate Authority (CA)-Paket zurück

ÜBERSICHT
`easy:UnsetOpt_Proxy_CAInfo()`

BESCHREIBUNG
Siehe Abschnitt 5.172 [`easy:SetOpt_Proxy_CAInfo`], Seite 127, für Details.

EINGABEN
keine

5.390 easy:UnsetOpt_Proxy_CAPath

BEZEICHNUNG
easy:UnsetOpt_Proxy_CAPath – setzt das Verzeichnis mit Proxy-CA-Zertifikaten zurück

ÜBERSICHT
`easy:UnsetOpt_Proxy_CAPath()`

BESCHREIBUNG
Siehe Abschnitt 5.173 [`easy:SetOpt_Proxy_CAPath`], Seite 128, für Details.

EINGABEN
keine

5.391 easy:UnsetOpt_Proxy_CRLFile

BEZEICHNUNG
easy:UnsetOpt_Proxy_CRLFile – setzt die Datei für Proxy-Zertifikatssperrlisten zurück

ÜBERSICHT
```
easy:UnsetOpt_Proxy_CRLFile()
```

BESCHREIBUNG
Siehe Abschnitt 5.174 [easy:SetOpt_Proxy_CRLFile], Seite 129, für Details.

EINGABEN
keine

5.392 easy:UnsetOpt_Proxy_KeyPasswd

BEZEICHNUNG
easy:UnsetOpt_Proxy_KeyPasswd – setzt die Passphrase auf privaten Proxy-Schlüssel zurück

ÜBERSICHT
```
easy:UnsetOpt_Proxy_KeyPasswd()
```

BESCHREIBUNG
Siehe Abschnitt 5.175 [easy:SetOpt_Proxy_KeyPasswd], Seite 129, für Details.

EINGABEN
keine

5.393 easy:UnsetOpt_Proxy_PinnedPublicKey

BEZEICHNUNG
easy:UnsetOpt_Proxy_PinnedPublicKey – setzt das Public Key Pinning für https-Proxy zurück

ÜBERSICHT
```
easy:UnsetOpt_Proxy_PinnedPublicKey()
```

BESCHREIBUNG
Siehe Abschnitt 5.176 [easy:SetOpt_Proxy_PinnedPublicKey], Seite 130, für Details.

EINGABEN
keine

5.394 easy:UnsetOpt_Proxy_Service_Name

BEZEICHNUNG
easy:UnsetOpt_Proxy_Service_Name – setzt den Namen wieder auf den Standarddienstnamen zurück

ÜBERSICHT
`easy:UnsetOpt_Proxy_Service_Name()`

BESCHREIBUNG
Siehe Abschnitt 5.177 [`easy:SetOpt_Proxy_Service_Name`], Seite 130, für Details.

EINGABEN
keine

5.395 easy:UnsetOpt_Proxy_SSL_Cipher_List

BEZEICHNUNG
easy:UnsetOpt_Proxy_SSL_Cipher_List – setzt die für Proxy-TLS zu verwendenden Verschlüsselungsart zurück

ÜBERSICHT
`easy:UnsetOpt_Proxy_SSL_Cipher_List()`

BESCHREIBUNG
Siehe Abschnitt 5.178 [`easy:SetOpt_Proxy_SSL_Cipher_List`], Seite 130, für Details.

EINGABEN
keine

5.396 easy:UnsetOpt_Proxy_SSL_Options

BEZEICHNUNG
easy:UnsetOpt_Proxy_SSL_Options – setzt die Proxy-SSL-Verhaltensoptionen zurück

ÜBERSICHT
`easy:UnsetOpt_Proxy_SSL_Options()`

BESCHREIBUNG
Siehe Abschnitt 5.179 [`easy:SetOpt_Proxy_SSL_Options`], Seite 131, für Details.

EINGABEN
keine

5.397 easy:UnsetOpt_Proxy_SSL_VerifyHost

BEZEICHNUNG
easy:UnsetOpt_Proxy_SSL_VerifyHost – setzt die Überprüfung des Namens auf den Standard zurück

ÜBERSICHT
`easy:UnsetOpt_Proxy_SSL_VerifyHost()`

BESCHREIBUNG
Siehe Abschnitt 5.180 [`easy:SetOpt_Proxy_SSL_VerifyHost`], Seite 132, für Details.

EINGABEN
keine

5.398 easy:UnsetOpt_Proxy_SSL_VerifyPeer

BEZEICHNUNG
easy:UnsetOpt_Proxy_SSL_VerifyPeer – deaktiviert die Überprüfung des SSL-Zertifikats des Proxys

ÜBERSICHT
```
easy:UnsetOpt_Proxy_SSL_VerifyPeer()
```

BESCHREIBUNG
Siehe Abschnitt 5.181 [easy:SetOpt_Proxy_SSL_VerifyPeer], Seite 132, für Details.

EINGABEN
keine

5.399 easy:UnsetOpt_Proxy_SSLCert

BEZEICHNUNG
easy:UnsetOpt_Proxy_SSLCert – setzt das SSL-Proxy-Client-Zertifikat zurück

ÜBERSICHT
```
easy:UnsetOpt_Proxy_SSLCert()
```

BESCHREIBUNG
Siehe Abschnitt 5.182 [easy:SetOpt_Proxy_SSLCert], Seite 133, für Details.

EINGABEN
keine

5.400 easy:UnsetOpt_Proxy_SSLCertType

BEZEICHNUNG
easy:UnsetOpt_Proxy_SSLCertType – setzt den Typ des Proxy-Client-SSL-Zertifikats zurück

ÜBERSICHT
```
easy:UnsetOpt_Proxy_SSLCertType()
```

BESCHREIBUNG
Siehe Abschnitt 5.183 [easy:SetOpt_Proxy_SSLCertType], Seite 134, für Details.

EINGABEN
keine

5.401 easy:UnsetOpt_Proxy_SSLKey

BEZEICHNUNG
easy:UnsetOpt_Proxy_SSLKey – setzt die private Schlüsseldatei für das TLS- und SSL-Proxy-Client-Zertifikat zurück

ÜBERSICHT
`easy:UnsetOpt_Proxy_SSLKey()`

BESCHREIBUNG
Siehe Abschnitt 5.184 [`easy:SetOpt_Proxy_SSLKey`], Seite 134, für Details.

EINGABEN
keine

5.402 easy:UnsetOpt_Proxy_SSLKeyType

BEZEICHNUNG
easy:UnsetOpt_Proxy_SSLKeyType – setzt den Typ der privaten Proxy-Schlüsseldatei zurück

ÜBERSICHT
`easy:UnsetOpt_Proxy_SSLKeyType()`

BESCHREIBUNG
Siehe Abschnitt 5.185 [`easy:SetOpt_Proxy_SSLKeyType`], Seite 135, für Details.

EINGABEN
keine

5.403 easy:UnsetOpt_Proxy_SSLVersion

BEZEICHNUNG
easy:UnsetOpt_Proxy_SSLVersion – setzt die bevorzugte Proxy-TLS/SSL-Version zurück

ÜBERSICHT
`easy:UnsetOpt_Proxy_SSLVersion()`

BESCHREIBUNG
Siehe Abschnitt 5.186 [`easy:SetOpt_Proxy_SSLVersion`], Seite 135, für Details.

EINGABEN
keine

5.404 easy:UnsetOpt_Proxy_TLSAuth_Password

BEZEICHNUNG
easy:UnsetOpt_Proxy_TLSAuth_Password – setzt das Passwort für die Proxy-TLS-Authentifizierung zurück

ÜBERSICHT
`easy:UnsetOpt_Proxy_TLSAuth_Password()`

BESCHREIBUNG
Siehe Abschnitt 5.187 [`easy:SetOpt_Proxy_TLSAuth_Password`], Seite 136, für Details.

EINGABEN
keine

5.405 easy:UnsetOpt_Proxy_TLSAuth_Type

BEZEICHNUNG
easy:UnsetOpt_Proxy_TLSAuth_Type – setzt die Proxy-TLS-Authentifizierungsmethoden zurück

ÜBERSICHT
`easy:UnsetOpt_Proxy_TLSAuth_Type()`

BESCHREIBUNG
Siehe Abschnitt 5.188 [`easy:SetOpt_Proxy_TLSAuth_Type`], Seite 137, für Details.

EINGABEN
keine

5.406 easy:UnsetOpt_Proxy_TLSAuth_UserName

BEZEICHNUNG
easy:UnsetOpt_Proxy_TLSAuth_UserName – setzt den Benutzername zur Verwendung für die Proxy-TLS-Authentifizierung zurück

ÜBERSICHT
`easy:UnsetOpt_Proxy_TLSAuth_UserName()`

BESCHREIBUNG
Siehe Abschnitt 5.189 [`easy:SetOpt_Proxy_TLSAuth_UserName`], Seite 137, für Details.

EINGABEN
keine

5.407 easy:UnsetOpt_Proxy_Transfer_Mode

BEZEICHNUNG
easy:UnsetOpt_Proxy_Transfer_Mode – hängt den FTP-Übertragungsmodus nicht mehr an die URL für Proxy an

ÜBERSICHT
`easy:UnsetOpt_Proxy_Transfer_Mode()`

BESCHREIBUNG
Siehe Abschnitt 5.190 [`easy:SetOpt_Proxy_Transfer_Mode`], Seite 138, für Details.

EINGABEN
keine

5.408 easy:UnsetOpt_ProxyAuth

BEZEICHNUNG
easy:UnsetOpt_ProxyAuth – setzt die HTTP-Proxy-Authentifizierungsmethoden für den Versuch zurück

ÜBERSICHT
`easy:UnsetOpt_ProxyAuth()`

BESCHREIBUNG
Siehe Abschnitt 5.191 [easy:SetOpt_ProxyAuth], Seite 138, für Details.

EINGABEN
keine

5.409 easy:UnsetOpt_ProxyHeader

BEZEICHNUNG
easy:UnsetOpt_ProxyHeader – setzt die an den Proxy zu übergebenden benutzerdefinierte HTTP-Header zurück

ÜBERSICHT
`easy:UnsetOpt_ProxyHeader()`

BESCHREIBUNG
Siehe Abschnitt 5.192 [easy:SetOpt_ProxyHeader], Seite 138, für Details.

EINGABEN
keine

5.410 easy:UnsetOpt_ProxyPassword

BEZEICHNUNG
easy:UnsetOpt_ProxyPassword – setzt das Passwort für die Proxy-Authentifizierung zurück

ÜBERSICHT
`easy:UnsetOpt_ProxyPassword()`

BESCHREIBUNG
Siehe Abschnitt 5.193 [easy:SetOpt_ProxyPassword], Seite 139, für Details.

EINGABEN
keine

5.411 easy:UnsetOpt_ProxyPort

BEZEICHNUNG
easy:UnsetOpt_ProxyPort – setzt die Portnummer für den Proxy zurück

ÜBERSICHT
```
easy:UnsetOpt_ProxyPort()
```

BESCHREIBUNG
Siehe Abschnitt 5.194 [easy:SetOpt_ProxyPort], Seite 139, für Details.

EINGABEN
keine

5.412 easy:UnsetOpt_ProxyType

BEZEICHNUNG
easy:UnsetOpt_ProxyType – setzt den Proxy-Protokolltyp zurück

ÜBERSICHT
```
easy:UnsetOpt_ProxyType()
```

BESCHREIBUNG
Siehe Abschnitt 5.195 [easy:SetOpt_ProxyType], Seite 140, für Details.

EINGABEN
keine

5.413 easy:UnsetOpt_ProxyUserName

BEZEICHNUNG
easy:UnsetOpt_ProxyUserName – setzt den Benutzername für die Proxy-Authentifizierung zurück

ÜBERSICHT
```
easy:UnsetOpt_ProxyUserName()
```

BESCHREIBUNG
Siehe Abschnitt 5.196 [easy:SetOpt_ProxyUserName], Seite 140, für Details.

EINGABEN
keine

5.414 easy:UnsetOpt_ProxyUserPwd

BEZEICHNUNG
easy:UnsetOpt_ProxyUserPwd – setzt den Benutzernamen und das Passwort für die Proxy-Authentifizierung zurück

ÜBERSICHT
`easy:UnsetOpt_ProxyUserPwd()`

BESCHREIBUNG
Siehe Abschnitt 5.197 [`easy:SetOpt_ProxyUserPwd`], Seite 141, für Details.

EINGABEN
keine

5.415 easy:UnsetOpt_Put

BEZEICHNUNG
easy:UnsetOpt_Put – stellt keine HTTP-PUT-Anfrage mehr

ÜBERSICHT
`easy:UnsetOpt_Put()`

BESCHREIBUNG
Siehe Abschnitt 5.198 [`easy:SetOpt_Put`], Seite 141, für Details.

EINGABEN
keine

5.416 easy:UnsetOpt_Quote

BEZEICHNUNG
easy:UnsetOpt_Quote – deaktiviert die (S)FTP-Befehle, die vor der Übertragung ausgeführt werden sollen

ÜBERSICHT
`easy:UnsetOpt_Quote()`

BESCHREIBUNG
Siehe Abschnitt 5.199 [`easy:SetOpt_Quote`], Seite 142, für Details.

EINGABEN
keine

5.417 easy:UnsetOpt_Random_File

BEZEICHNUNG
easy:UnsetOpt_Random_File – setzt die Quelle für zufällige Daten zurück

ÜBERSICHT
`easy:UnsetOpt_Random_File()`

BESCHREIBUNG
Siehe Abschnitt 5.200 [`easy:SetOpt_Random_File`], Seite 143, für Details.

EINGABEN
keine

5.418 easy:UnsetOpt_Range

BEZEICHNUNG
easy:UnsetOpt_Range – setzt den anzuforderenden Bytebereich zurück

ÜBERSICHT
`easy:UnsetOpt_Range()`

BESCHREIBUNG
Siehe Abschnitt 5.201 [easy:SetOpt_Range], Seite 143, für Details.

EINGABEN
keine

5.419 easy:UnsetOpt_ReadFunction

BEZEICHNUNG
easy:UnsetOpt_ReadFunction – setzt den Callback für Daten-Uploads zurück

ÜBERSICHT
`easy:UnsetOpt_ReadFunction()`

BESCHREIBUNG
Siehe Abschnitt 5.202 [easy:SetOpt_ReadFunction], Seite 144, für Details.

EINGABEN
keine

5.420 easy:UnsetOpt_Redir_Protocols

BEZEICHNUNG
easy:UnsetOpt_Redir_Protocols – setzt die Protokolle zurück, zu denen umgeleitet werden darf

ÜBERSICHT
`easy:UnsetOpt_Redir_Protocols()`

BESCHREIBUNG
Siehe Abschnitt 5.203 [easy:SetOpt_Redir_Protocols], Seite 145, für Details.

EINGABEN
keine

5.421 easy:UnsetOpt_Referer

BEZEICHNUNG
easy:UnsetOpt_Referer – setzt den HTTP Referer: Header zurück

ÜBERSICHT
`easy:UnsetOpt_Referer()`

BESCHREIBUNG
Siehe Abschnitt 5.204 [easy:SetOpt_Referer], Seite 146, für Details.

EINGABEN
keine

5.422 easy:UnsetOpt_Request_Target

BEZEICHNUNG
easy:UnsetOpt_Request_Target – setzt das alternative Ziel für diese Anforderung zurück

ÜBERSICHT
```
easy:UnsetOpt_Request_Target()
```

BESCHREIBUNG
Siehe Abschnitt 5.205 [easy:SetOpt_Request_Target], Seite 147, für Details.

EINGABEN
keine

5.423 easy:UnsetOpt_Resolve

BEZEICHNUNG
easy:UnsetOpt_Resolve – setzt den benutzerdefinierten Hostnamen für IP-Adressauflösungen zurück

ÜBERSICHT
```
easy:UnsetOpt_Resolve()
```

BESCHREIBUNG
Siehe Abschnitt 5.206 [easy:SetOpt_Resolve], Seite 147, für Details.

EINGABEN
keine

5.424 easy:UnsetOpt_Resume_From

BEZEICHNUNG
easy:UnsetOpt_Resume_From – deaktiviert die Fortsetzung der Übertragung und beginnt von vorne

ÜBERSICHT
```
easy:UnsetOpt_Resume_From()
```

BESCHREIBUNG
Siehe Abschnitt 5.207 [easy:SetOpt_Resume_From], Seite 148, für Details.

EINGABEN
keine

5.425 easy:UnsetOpt_Resume_From_Large

BEZEICHNUNG
easy:UnsetOpt_Resume_From_Large – deaktiviert die Fortsetzung der Übertragung und beginnt von vorne

ÜBERSICHT
`easy:UnsetOpt_Resume_From_Large()`

BESCHREIBUNG
Siehe Abschnitt 5.208 [`easy:SetOpt_Resume_From_Large`], Seite 148, für Details.

EINGABEN
keine

5.426 easy:UnsetOpt_RTSP_Client_CSeq

BEZEICHNUNG
easy:UnsetOpt_RTSP_Client_CSeq – setzt die RTSP-Client-CSEQ-Nummer zurück

ÜBERSICHT
`easy:UnsetOpt_RTSP_Client_CSeq()`

BESCHREIBUNG
Siehe Abschnitt 5.209 [`easy:SetOpt_RTSP_Client_CSeq`], Seite 149, für Details.

EINGABEN
keine

5.427 easy:UnsetOpt_RTSP_Request

BEZEICHNUNG
easy:UnsetOpt_RTSP_Request – setzt die RTSP-Anfrage zurück

ÜBERSICHT
`easy:UnsetOpt_RTSP_Request()`

BESCHREIBUNG
Siehe Abschnitt 5.210 [`easy:SetOpt_RTSP_Request`], Seite 149, für Details.

EINGABEN
keine

5.428 easy:UnsetOpt_RTSP_Server_CSeq

BEZEICHNUNG
easy:UnsetOpt_RTSP_Server_CSeq – setzt die CSEQ-Nummer des RTSP-Servers zurück

ÜBERSICHT
`easy:UnsetOpt_RTSP_Server_CSeq()`

BESCHREIBUNG
Siehe Abschnitt 5.211 [easy:SetOpt_RTSP_Server_CSeq], Seite 151, für Details.

EINGABEN
keine

5.429 easy:UnsetOpt_RTSP_Session_ID

BEZEICHNUNG
easy:UnsetOpt_RTSP_Session_ID – setzt die RTSP-Sitzungs-ID zurück

ÜBERSICHT
```
easy:UnsetOpt_RTSP_Session_ID()
```

BESCHREIBUNG
Siehe Abschnitt 5.212 [easy:SetOpt_RTSP_Session_ID], Seite 151, für Details.

EINGABEN
keine

5.430 easy:UnsetOpt_RTSP_Stream_URI

BEZEICHNUNG
easy:UnsetOpt_RTSP_Stream_URI – setzt die RTSP-Stream-URI zurück

ÜBERSICHT
```
easy:UnsetOpt_RTSP_Stream_URI()
```

BESCHREIBUNG
Siehe Abschnitt 5.213 [easy:SetOpt_RTSP_Stream_URI], Seite 152, für Details.

EINGABEN
keine

5.431 easy:UnsetOpt_RTSP_Transport

BEZEICHNUNG
easy:UnsetOpt_RTSP_Transport – setzt den RTSP Transport: Header zurück

ÜBERSICHT
```
easy:UnsetOpt_RTSP_Transport()
```

BESCHREIBUNG
Siehe Abschnitt 5.214 [easy:SetOpt_RTSP_Transport], Seite 152, für Details.

EINGABEN
keine

5.432 easy:UnsetOpt_SASL_IR

BEZEICHNUNG
easy:UnsetOpt_SASL_IR – deaktiviert das Senden der ersten Antwort im ersten Paket

ÜBERSICHT
```
easy:UnsetOpt_SASL_IR()
```

BESCHREIBUNG
Siehe Abschnitt 5.215 [easy:SetOpt_SASL_IR], Seite 153, für Details.

EINGABEN
keine

5.433 easy:UnsetOpt_SeekFunction

BEZEICHNUNG
easy:UnsetOpt_SeekFunction – setzt den Benutzer-Callback zum Suchen im Eingabedatenstrom zurück

ÜBERSICHT
```
easy:UnsetOpt_SeekFunction()
```

BESCHREIBUNG
Siehe Abschnitt 5.216 [easy:SetOpt_SeekFunction], Seite 153, für Details.

EINGABEN
keine

5.434 easy:UnsetOpt_Service_Name

BEZEICHNUNG
easy:UnsetOpt_Service_Name – setzt den Namen des Authentifizierungsdienstes zurück

ÜBERSICHT
```
easy:UnsetOpt_Service_Name()
```

BESCHREIBUNG
Siehe Abschnitt 5.217 [easy:SetOpt_Service_Name], Seite 154, für Details.

EINGABEN
keine

5.435 easy:UnsetOpt_Share

BEZEICHNUNG
easy:UnsetOpt_Share – setzt den Share-Handle zurück

ÜBERSICHT
```
easy:UnsetOpt_Share()
```

BESCHREIBUNG
Siehe Abschnitt 5.218 [easy:SetOpt_Share], Seite 154, für Details.

EINGABEN
keine

5.436 easy:UnsetOpt_Socks5_Auth

BEZEICHNUNG
easy:UnsetOpt_Socks5_Auth – setzt die zulässigen Methoden für die SOCKS5-Proxyauthentifizierung zurück

ÜBERSICHT
```
easy:UnsetOpt_Socks5_Auth()
```

BESCHREIBUNG
Siehe Abschnitt 5.219 [easy:SetOpt_Socks5_Auth], Seite 155, für Details.

EINGABEN
keine

5.437 easy:UnsetOpt_Socks5_GSSAPI_NEC

BEZEICHNUNG
easy:UnsetOpt_Socks5_GSSAPI_NEC – setzt den Socks Proxy gssapi Übertragungsschutz zurück

ÜBERSICHT
```
easy:UnsetOpt_Socks5_GSSAPI_NEC()
```

BESCHREIBUNG
Siehe Abschnitt 5.220 [easy:SetOpt_Socks5_GSSAPI_NEC], Seite 155, für Details.

EINGABEN
keine

5.438 easy:UnsetOpt_Socks5_GSSAPI_Service

BEZEICHNUNG
easy:UnsetOpt_Socks5_GSSAPI_Service – setzt den SOCKS5-Name des Proxy-Authentifizierungsdienstes zurück

ÜBERSICHT
```
easy:UnsetOpt_Socks5_GSSAPI_Service()
```

BESCHREIBUNG
Siehe Abschnitt 5.221 [easy:SetOpt_Socks5_GSSAPI_Service], Seite 156, für Details.

EINGABEN
keine

5.439 easy:UnsetOpt_SSH_Auth_Types

BEZEICHNUNG
easy:UnsetOpt_SSH_Auth_Types – setzt den gewünschten Authentifizierungstypen für SFTP und SCP zurück

ÜBERSICHT
`easy:UnsetOpt_SSH_Auth_Types()`

BESCHREIBUNG
Siehe Abschnitt 5.222 [`easy:SetOpt_SSH_Auth_Types`], Seite 156, für Details.

EINGABEN
keine

5.440 easy:UnsetOpt_SSH_Host_Public_Key_MD5

BEZEICHNUNG
easy:UnsetOpt_SSH_Host_Public_Key_MD5 – setzt die Prüfsumme des öffentlichen Schlüssels des SSH-Servers zurück

ÜBERSICHT
`easy:UnsetOpt_SSH_Host_Public_Key_MD5()`

BESCHREIBUNG
Siehe Abschnitt 5.223 [`easy:SetOpt_SSH_Host_Public_Key_MD5`], Seite 157, für Details.

EINGABEN
keine

5.441 easy:UnsetOpt_SSH_KnownHosts

BEZEICHNUNG
easy:UnsetOpt_SSH_KnownHosts – setzt den Dateiname mit den bekannten SSH-Hosts zurück

ÜBERSICHT
`easy:UnsetOpt_SSH_KnownHosts()`

BESCHREIBUNG
Siehe Abschnitt 5.224 [`easy:SetOpt_SSH_KnownHosts`], Seite 157, für Details.

EINGABEN
keine

5.442 easy:UnsetOpt_SSH_Private_KeyFile

BEZEICHNUNG
easy:UnsetOpt_SSH_Private_KeyFile – setzt die private Schlüsseldatei für SSH-Authentifizierung zurück

ÜBERSICHT
`easy:UnsetOpt_SSH_Private_KeyFile()`

BESCHREIBUNG
Siehe Abschnitt 5.225 [`easy:SetOpt_SSH_Private_KeyFile`], Seite 157, für Details.

EINGABEN
keine

5.443 easy:UnsetOpt_SSH_Public_KeyFile

BEZEICHNUNG
easy:UnsetOpt_SSH_Public_KeyFile – setzt die öffentliche Schlüsseldatei für die SSH-Authentifizierung zurück

ÜBERSICHT
`easy:UnsetOpt_SSH_Public_KeyFile()`

BESCHREIBUNG
Siehe Abschnitt 5.226 [`easy:SetOpt_SSH_Public_KeyFile`], Seite 158, für Details.

EINGABEN
keine

5.444 easy:UnsetOpt_SSL_Cipher_List

BEZEICHNUNG
easy:UnsetOpt_SSL_Cipher_List – setzt die Verschlüsselung zurück, die für TLS verwendet werden soll

ÜBERSICHT
`easy:UnsetOpt_SSL_Cipher_List()`

BESCHREIBUNG
Siehe Abschnitt 5.227 [`easy:SetOpt_SSL_Cipher_List`], Seite 158, für Details.

EINGABEN
keine

5.445 easy:UnsetOpt_SSL_Enable_Alpn

BEZEICHNUNG
easy:UnsetOpt_SSL_Enable_Alpn – deaktiviert ALPN

ÜBERSICHT
```
easy:UnsetOpt_SSL_Enable_Alpn()
```

BESCHREIBUNG
Siehe Abschnitt 5.228 [easy:SetOpt_SSL_Enable_Alpn], Seite 159, für Details.

EINGABEN
keine

5.446 easy:UnsetOpt_SSL_Enable_Npn

BEZEICHNUNG
easy:UnsetOpt_SSL_Enable_Npn – deaktiviert NPN

ÜBERSICHT
```
easy:UnsetOpt_SSL_Enable_Npn()
```

BESCHREIBUNG
Siehe Abschnitt 5.229 [easy:SetOpt_SSL_Enable_Npn], Seite 159, für Details.

EINGABEN
keine

5.447 easy:UnsetOpt_SSL_FalseStart

BEZEICHNUNG
easy:UnsetOpt_SSL_FalseStart – deaktiviert TLS-Fehlstart

ÜBERSICHT
```
easy:UnsetOpt_SSL_FalseStart()
```

BESCHREIBUNG
Siehe Abschnitt 5.230 [easy:SetOpt_SSL_FalseStart], Seite 160, für Details.

EINGABEN
keine

5.448 easy:UnsetOpt_SSL_Options

BEZEICHNUNG
easy:UnsetOpt_SSL_Options – setzt SSL-Verhaltensoptionen zurück

ÜBERSICHT
```
easy:UnsetOpt_SSL_Options()
```

BESCHREIBUNG
Siehe Abschnitt 5.231 [easy:SetOpt_SSL_Options], Seite 160, für Details.

EINGABEN
keine

5.449 easy:UnsetOpt_SSL_SessionID_Cache

BEZEICHNUNG
easy:UnsetOpt_SSL_SessionID_Cache – aktiviert wieder die Verwendung des SSL-Sitzungs-ID-Cache

ÜBERSICHT
```
easy:UnsetOpt_SSL_SessionID_Cache()
```

BESCHREIBUNG
Siehe Abschnitt 5.232 [easy:SetOpt_SSL_SessionID_Cache], Seite 161, für Details.

EINGABEN
keine

5.450 easy:UnsetOpt_SSL_VerifyHost

BEZEICHNUNG
easy:UnsetOpt_SSL_VerifyHost – setzt die Überprüfung des Zertifikatsnamen auf den Standard zurück

ÜBERSICHT
```
easy:UnsetOpt_SSL_VerifyHost()
```

BESCHREIBUNG
Siehe Abschnitt 5.233 [easy:SetOpt_SSL_VerifyHost], Seite 161, für Details.

EINGABEN
keine

5.451 easy:UnsetOpt_SSL_VerifyPeer

BEZEICHNUNG
easy:UnsetOpt_SSL_VerifyPeer – deaktiviert die Überprüfung das SSL-Zertifikat des Peers

ÜBERSICHT
```
easy:UnsetOpt_SSL_VerifyPeer()
```

BESCHREIBUNG
Siehe Abschnitt 5.234 [easy:SetOpt_SSL_VerifyPeer], Seite 162, für Details.

EINGABEN
keine

5.452 easy:UnsetOpt_SSL_VerifyStatus

BEZEICHNUNG
easy:UnsetOpt_SSL_VerifyStatus – überprüft den Status des Zertifikats nicht mehr

ÜBERSICHT
`easy:UnsetOpt_SSL_VerifyStatus()`

BESCHREIBUNG
Siehe Abschnitt 5.235 [`easy:SetOpt_SSL_VerifyStatus`], Seite 163, für Details.

EINGABEN
keine

5.453 easy:UnsetOpt_SSLCert

BEZEICHNUNG
easy:UnsetOpt_SSLCert – setzt das SSL-Client-Zertifikat auf Standard zurück

ÜBERSICHT
`easy:UnsetOpt_SSLCert()`

BESCHREIBUNG
Siehe Abschnitt 5.236 [`easy:SetOpt_SSLCert`], Seite 163, für Details.

EINGABEN
keine

5.454 easy:UnsetOpt_SSLCertType

BEZEICHNUNG
easy:UnsetOpt_SSLCertType – setzt den Typ des Client-SSL-Zertifikats zurück

ÜBERSICHT
`easy:UnsetOpt_SSLCertType()`

BESCHREIBUNG
Siehe Abschnitt 5.237 [`easy:SetOpt_SSLCertType`], Seite 164, für Details.

EINGABEN
keine

5.455 easy:UnsetOpt_SSLEngine

BEZEICHNUNG
easy:UnsetOpt_SSLEngine – setzt die SSL System ID zurück

ÜBERSICHT
`easy:UnsetOpt_SSLEngine()`

BESCHREIBUNG
Siehe Abschnitt 5.238 [easy:SetOpt_SSLEngine], Seite 164, für Details.

EINGABEN
keine

5.456 easy:UnsetOpt_SSLEngine_Default

BEZEICHNUNG
easy:UnsetOpt_SSLEngine_Default – setzt das SSL-System zurück

ÜBERSICHT
```
easy:UnsetOpt_SSLEngine_Default()
```

BESCHREIBUNG
Siehe Abschnitt 5.239 [easy:SetOpt_SSLEngine_Default], Seite 165, für Details.

EINGABEN
keine

5.457 easy:UnsetOpt_SSLKey

BEZEICHNUNG
easy:UnsetOpt_SSLKey – setzt die private Schlüsseldatei für TLS- und SSL-Client-Zertifikate zurück

ÜBERSICHT
```
easy:UnsetOpt_SSLKey()
```

BESCHREIBUNG
Siehe Abschnitt 5.240 [easy:SetOpt_SSLKey], Seite 165, für Details.

EINGABEN
keine

5.458 easy:UnsetOpt_SSLKeyType

BEZEICHNUNG
easy:UnsetOpt_SSLKeyType – setzt den Typ der privaten Schlüsseldatei

ÜBERSICHT
```
easy:UnsetOpt_SSLKeyType()
```

BESCHREIBUNG
Siehe Abschnitt 5.241 [easy:SetOpt_SSLKeyType], Seite 166, für Details.

EINGABEN
keine

5.459 easy:UnsetOpt_SSLVersion

BEZEICHNUNG
easy:UnsetOpt_SSLVersion – stellt die bevorzugte TLS/SSL-Version zurück

ÜBERSICHT
`easy:UnsetOpt_SSLVersion()`

BESCHREIBUNG
Siehe Abschnitt 5.242 [`easy:SetOpt_SSLVersion`], Seite 166, für Details.

EINGABEN
keine

5.460 easy:UnsetOpt_Stream_Depends

BEZEICHNUNG
easy:UnsetOpt_Stream_Depends – stellt den Stream zurück, von dem diese Übertragung abhängt

ÜBERSICHT
`easy:UnsetOpt_Stream_Depends()`

BESCHREIBUNG
Siehe Abschnitt 5.243 [`easy:SetOpt_Stream_Depends`], Seite 167, für Details.

EINGABEN
keine

5.461 easy:UnsetOpt_Stream_Depends_e

BEZEICHNUNG
easy:UnsetOpt_Stream_Depends_e – stellt den Stream zurück, von dem diese Übertragung ausschließlich abhängt

ÜBERSICHT
`easy:UnsetOpt_Stream_Depends_e()`

BESCHREIBUNG
Siehe Abschnitt 5.244 [`easy:SetOpt_Stream_Depends_e`], Seite 168, für Details.

EINGABEN
keine

5.462 easy:UnsetOpt_Stream_Weight

BEZEICHNUNG
easy:UnsetOpt_Stream_Weight – setzt die Gewichtung des numerischen Datenstroms zurück

ÜBERSICHT
`easy:UnsetOpt_Stream_Weight()`

BESCHREIBUNG
Siehe Abschnitt 5.245 [`easy:SetOpt_Stream_Weight`], Seite 168, für Details.

EINGABEN
keine

5.463 easy:UnsetOpt_Suppress_Connect_Headers

BEZEICHNUNG
easy:UnsetOpt_Suppress_Connect_Headers – unterdrückt Proxy-CONNECT-Antwort-Header von Benutzer-Callbacks nicht mehr

ÜBERSICHT
`easy:UnsetOpt_Suppress_Connect_Headers()`

BESCHREIBUNG
Siehe Abschnitt 5.246 [`easy:SetOpt_Suppress_Connect_Headers`], Seite 169, für Details.

EINGABEN
keine

5.464 easy:UnsetOpt_TCP_FastOpen

BEZEICHNUNG
easy:UnsetOpt_TCP_FastOpen – deaktiviert TCP Fast Open

ÜBERSICHT
`easy:UnsetOpt_TCP_FastOpen()`

BESCHREIBUNG
Siehe Abschnitt 5.247 [`easy:SetOpt_TCP_FastOpen`], Seite 170, für Details.

EINGABEN
keine

5.465 easy:UnsetOpt_TCP_KeepAlive

BEZEICHNUNG
easy:UnsetOpt_TCP_KeepAlive – deaktiviert die TCP-Keep-Alive-Tests

ÜBERSICHT
`easy:UnsetOpt_TCP_KeepAlive()`

BESCHREIBUNG
Siehe Abschnitt 5.248 [`easy:SetOpt_TCP_KeepAlive`], Seite 170, für Details.

EINGABEN
keine

5.466 easy:UnsetOpt_TCP_KeepIdle

BEZEICHNUNG
easy:UnsetOpt_TCP_KeepIdle – setzt die TCP-Keep-Alive Leerlaufzeit zurück

ÜBERSICHT
`easy:UnsetOpt_TCP_KeepIdle()`

BESCHREIBUNG
Siehe Abschnitt 5.249 [`easy:SetOpt_TCP_KeepIdle`], Seite 171, für Details.

EINGABEN
keine

5.467 easy:UnsetOpt_TCP_KeepIntvl

BEZEICHNUNG
easy:UnsetOpt_TCP_KeepIntvl – setzt den TCP-Keep-Alive-Intervall zurück

ÜBERSICHT
`easy:UnsetOpt_TCP_KeepIntvl()`

BESCHREIBUNG
Siehe Abschnitt 5.250 [`easy:SetOpt_TCP_KeepIntvl`], Seite 171, für Details.

EINGABEN
keine

5.468 easy:UnsetOpt_TCP_NoDelay

BEZEICHNUNG
easy:UnsetOpt_TCP_NoDelay – aktiviert die Option TCP_NODELAY

ÜBERSICHT
`easy:UnsetOpt_TCP_NoDelay()`

BESCHREIBUNG
Siehe Abschnitt 5.251 [`easy:SetOpt_TCP_NoDelay`], Seite 171, für Details.

EINGABEN
keine

5.469 easy:UnsetOpt_TelnetOptions

BEZEICHNUNG
easy:UnsetOpt_TelnetOptions – setzt die benutzerdefinierten Telnet-Optionen zurück

ÜBERSICHT
`easy:UnsetOpt_TelnetOptions()`

BESCHREIBUNG
Siehe Abschnitt 5.252 [`easy:SetOpt_TelnetOptions`], Seite 172, für Details.

EINGABEN
keine

5.470 easy:UnsetOpt_TFTP_BlkSize

BEZEICHNUNG
easy:UnsetOpt_TFTP_BlkSize – setzt die TFTP-Blockgröße auf 512 Byte zurück

ÜBERSICHT
`easy:UnsetOpt_TFTP_BlkSize()`

BESCHREIBUNG
Siehe Abschnitt 5.253 [`easy:SetOpt_TFTP_BlkSize`], Seite 172, für Details.

EINGABEN
keine

5.471 easy:UnsetOpt_TFTP_No_Options

BEZEICHNUNG
easy:UnsetOpt_TFTP_No_Options – sendet wieder TFTP-Optionsanforderungen

ÜBERSICHT
`easy:UnsetOpt_TFTP_No_Options()`

BESCHREIBUNG
Siehe Abschnitt 5.254 [`easy:SetOpt_TFTP_No_Options`], Seite 173, für Details.

EINGABEN
keine

5.472 easy:UnsetOpt_TimeCondition

BEZEICHNUNG
easy:UnsetOpt_TimeCondition – setzt die Bedingung für eine Zeitanfrage zurück

ÜBERSICHT
`easy:UnsetOpt_TimeCondition()`

BESCHREIBUNG
Siehe Abschnitt 5.255 [easy:SetOpt_TimeCondition], Seite 173, für Details.

EINGABEN
keine

5.473 easy:UnsetOpt_Timeout

BEZEICHNUNG
easy:UnsetOpt_Timeout – setzt die maximale Zeit zurück, die die Anforderung dauern darf

ÜBERSICHT
```
easy:UnsetOpt_Timeout()
```

BESCHREIBUNG
Siehe Abschnitt 5.256 [easy:SetOpt_Timeout], Seite 174, für Details.

EINGABEN
keine

5.474 easy:UnsetOpt_Timeout_MS

BEZEICHNUNG
easy:UnsetOpt_Timeout_MS – setzt die maximale Zeit zurück, die die Anforderung dauern darf

ÜBERSICHT
```
easy:UnsetOpt_Timeout_MS()
```

BESCHREIBUNG
Siehe Abschnitt 5.257 [easy:SetOpt_Timeout_MS], Seite 174, für Details.

EINGABEN
keine

5.475 easy:UnsetOpt_TimeValue

BEZEICHNUNG
easy:UnsetOpt_TimeValue – setzt den Zeitwert für bedingtes Verhalten zurück

ÜBERSICHT
```
easy:UnsetOpt_TimeValue()
```

BESCHREIBUNG
Siehe Abschnitt 5.258 [easy:SetOpt_TimeValue], Seite 175, für Details.

EINGABEN
keine

5.476 easy:UnsetOpt_TLSAuth_Password

BEZEICHNUNG
easy:UnsetOpt_TLSAuth_Password – setzt das Passwort für die TLS-Authentifizierung zurück

ÜBERSICHT
`easy:UnsetOpt_TLSAuth_Password()`

BESCHREIBUNG
Siehe Abschnitt 5.259 [`easy:SetOpt_TLSAuth_Password`], Seite 175, für Details.

EINGABEN
keine

5.477 easy:UnsetOpt_TLSAuth_Type

BEZEICHNUNG
easy:UnsetOpt_TLSAuth_Type – setzt die TLS-Authentifizierungsmethoden zurück

ÜBERSICHT
`easy:UnsetOpt_TLSAuth_Type()`

BESCHREIBUNG
Siehe Abschnitt 5.260 [`easy:SetOpt_TLSAuth_Type`], Seite 176, für Details.

EINGABEN
keine

5.478 easy:UnsetOpt_TLSAuth_UserName

BEZEICHNUNG
easy:UnsetOpt_TLSAuth_UserName – setzt den Benutzernamen zurück, der für die TLS-Authentifizierung verwendet wird

ÜBERSICHT
`easy:UnsetOpt_TLSAuth_UserName()`

BESCHREIBUNG
Siehe Abschnitt 5.261 [`easy:SetOpt_TLSAuth_UserName`], Seite 176, für Details.

EINGABEN
keine

5.479 easy:UnsetOpt_Transfer_Encoding

BEZEICHNUNG
easy:UnsetOpt_Transfer_Encoding – fordert die Übertragungscodierung nicht mehr an

ÜBERSICHT
`easy:UnsetOpt_Transfer_Encoding()`

BESCHREIBUNG
Siehe Abschnitt 5.262 [`easy:SetOpt_Transfer_Encoding`], Seite 176, für Details.

EINGABEN
keine

5.480 easy:UnsetOpt_TransferText

BEZEICHNUNG
easy:UnsetOpt_TransferText – fordet wieder eine Binärübertragung für FTP an

ÜBERSICHT
`easy:UnsetOpt_TransferText()`

BESCHREIBUNG
Siehe Abschnitt 5.263 [`easy:SetOpt_TransferText`], Seite 177, für Details.

EINGABEN
keine

5.481 easy:UnsetOpt_Unix_Socket_Path

BEZEICHNUNG
easy:UnsetOpt_Unix_Socket_Path – setzt den Unix Domain Socket zurück

ÜBERSICHT
`easy:UnsetOpt_Unix_Socket_Path()`

BESCHREIBUNG
Siehe Abschnitt 5.264 [`easy:SetOpt_Unix_Socket_Path`], Seite 177, für Details.

EINGABEN
keine

5.482 easy:UnsetOpt_Unrestricted_Auth

BEZEICHNUNG
easy:UnsetOpt_Unrestricted_Auth – sendet wieder die Authentifizierungsdaten nur an den ursprünglichen Hostnamen

ÜBERSICHT
`easy:UnsetOpt_Unrestricted_Auth()`

BESCHREIBUNG
Siehe Abschnitt 5.265 [`easy:SetOpt_Unrestricted_Auth`], Seite 178, für Details.

EINGABEN
keine

5.483 easy:UnsetOpt_Upload

BEZEICHNUNG
easy:UnsetOpt_Upload – deaktiviert das Hochladen von Daten

ÜBERSICHT
`easy:UnsetOpt_Upload()`

BESCHREIBUNG
Siehe Abschnitt 5.266 [`easy:SetOpt_Upload`], Seite 178, für Details.

EINGABEN
keine

5.484 easy:UnsetOpt_URL

BEZEICHNUNG
easy:UnsetOpt_URL – gibt die URL an, die in der Anfrage verwendet werden soll

ÜBERSICHT
`easy:UnsetOpt_URL()`

BESCHREIBUNG
Siehe Abschnitt 5.267 [`easy:SetOpt_URL`], Seite 179, für Details.

EINGABEN
keine

5.485 easy:UnsetOpt_Use_SSL

BEZEICHNUNG
easy:UnsetOpt_Use_SSL – fordert für die Übertragung kein SSL/TLS mehr an

ÜBERSICHT
`easy:UnsetOpt_Use_SSL()`

BESCHREIBUNG
Siehe Abschnitt 5.268 [`easy:SetOpt_Use_SSL`], Seite 185, für Details.

EINGABEN
keine

5.486 easy:UnsetOpt_UserAgent

BEZEICHNUNG
easy:UnsetOpt_UserAgent – setzt den HTTP-User-Agent-Header zurück

ÜBERSICHT
`easy:UnsetOpt_UserAgent()`

BESCHREIBUNG
Siehe Abschnitt 5.269 [easy:SetOpt_UserAgent], Seite 185, für Details.

EINGABEN
keine

5.487 easy:UnsetOpt_UserName

BEZEICHNUNG
easy:UnsetOpt_UserName – setzt den Benutzername für die Authentifizierung zurück

ÜBERSICHT
```
easy:UnsetOpt_UserName()
```

BESCHREIBUNG
Siehe Abschnitt 5.270 [easy:SetOpt_UserName], Seite 186, für Details.

EINGABEN
keine

5.488 easy:UnsetOpt_UserPwd

BEZEICHNUNG
easy:UnsetOpt_UserPwd – setzt den Benutzername und das Passwort für die Authentifizierung zurück

ÜBERSICHT
```
easy:UnsetOpt_UserPwd()
```

BESCHREIBUNG
Siehe Abschnitt 5.271 [easy:SetOpt_UserPwd], Seite 186, für Details.

EINGABEN
keine

5.489 easy:UnsetOpt_Verbose

BEZEICHNUNG
easy:UnsetOpt_Verbose – schaltet den ausführlichen Modus aus

ÜBERSICHT
```
easy:UnsetOpt_Verbose()
```

BESCHREIBUNG
Siehe Abschnitt 5.272 [easy:SetOpt_Verbose], Seite 187, für Details.

EINGABEN
keine

5.490 easy:UnsetOpt_WildcardMatch

BEZEICHNUNG
easy:UnsetOpt_WildcardMatch – deaktiviert die Übertragung von Verzeichnis-Platzhaltern

ÜBERSICHT
`easy:UnsetOpt_WildcardMatch()`

BESCHREIBUNG
Siehe Abschnitt 5.273 [`easy:SetOpt_WildcardMatch`], Seite 188, für Details.

EINGABEN
keine

5.491 easy:UnsetOpt_WriteFunction

BEZEICHNUNG
easy:UnsetOpt_WriteFunction – setzt den Callback zum Schreiben empfangener Daten zurück

ÜBERSICHT
`easy:UnsetOpt_WriteFunction()`

BESCHREIBUNG
Siehe Abschnitt 5.274 [`easy:SetOpt_WriteFunction`], Seite 189, für Details.

EINGABEN
keine

5.492 easy:UnsetOpt_XOAuth2_Bearer

BEZEICHNUNG
easy:UnsetOpt_XOAuth2_Bearer – setzt den OAuth 2.0 Access Token zurück

ÜBERSICHT
`easy:UnsetOpt_XOAuth2_Bearer()`

BESCHREIBUNG
Siehe Abschnitt 5.275 [`easy:SetOpt_XOAuth2_Bearer`], Seite 190, für Details.

EINGABEN
keine

6 Form-Methoden

6.1 form:AddBuffer

BEZEICHNUNG

form:AddBuffer – fügt die Datei-Hochlade-Sektion aus dem Puffer hinzu

ÜBERSICHT

```
form:AddBuffer(name, filename, content[, type, headers])
```

BESCHREIBUNG

`form:AddBuffer()` wird zum Anhängen einer Datei-Hochlade-Sektion (aus einer Pufferquelle) beim Erstellen eines HTTP-POST mit mehreren Multipart/Formdata (manchmal als RFC 2388-Posts bezeichnet) verwendet. Übergeben Sie den Form-Handle als Parameter an `#CURLOPT_HTTPPOST`, nachdem Sie alle Sektionen hinzugefügt haben, die Sie einschließen möchten. Siehe Abschnitt 5.125 [easy:SetOpt_HTTPPost], Seite 102, für Details.

Sie müssen `form:Free()` aufrufen, nachdem der Form-Post ausgeführt wurde, um die Ressourcen freizugeben.

Die Verwendung von POST mit HTTP 1.1 impliziert die Verwendung eines Headers "Expect: 100-continue". Sie können diese Header wie gewohnt mit `#CURLOPT_HTTPHEADER` deaktivieren.

Erstens gibt es einige Grundlagen, die Sie über Posts mit mehreren Multipart/Formdata verstehen müssen. Jeder Teil besteht aus mindestens einem `NAME-` und einem `CONTENTS-`Teil. Wenn der Teil zum Hochladen von Dateien erstellt wurde, gibt es auch einen gespeicherten `CONTENT-TYPE`} und einen `FILENAME` (Dateiname). Im Folgenden werden die Optionen erläutert, mit denen Sie diese Eigenschaften in den Teilen festlegen, die Sie zu Ihrem Post hinzufügen möchten.

Das Argument `name` muss eine Zeichenkette sein, die den Namen dieses Teils angibt. Der Name darf keine nullwertigen Bytes enthalten. Das Argument `filename` muss eine Zeichenkette sein, die das Feld Dateiname in dem Content-Header enthält. Das Argument `content` muss die tatsächlich zu sendenden Daten enthalten. Das optionale Argument `type` kann verwendet werden, um den Content-Typ für den Teil festzulegen und das optionale Argument `headers` kann verwendet werden, um zusätzliche Header für die POST-Sektion des Formulars anzugeben. Dies erfordert eine Tabelle mit einer Liste und hängt die Liste der Header an die von libcurl automatisch erzeugten an.

EINGABEN

`name`	Name des Elements
`filename`	Dateiname für die Content-Header
`content`	zu sendende Ist-Daten
`type`	Optional: Content-Typ für das Element
`headers`	Optional: zusätzliche Header für die POST-Sektion

6.2 form:AddContent

BEZEICHNUNG

form:AddContent – fügt eine Sektion zu einem HTTP-POST mit mehreren Multipart/Formdata hinzu

ÜBERSICHT

```
form:AddContent(name, content[, type, headers])
```

BESCHREIBUNG

`form:AddContent()` wird zum Anhängen einer Sektion beim Erstellen eines HTTP-POST mit mehreren Multipart/Formdata (Teilen/Formulardaten) (manchmal als RFC 2388-Posts bezeichnet) verwendet. Übergeben Sie den Form-Handle als Parameter an `#CURLOPT_HTTPPOST`, nachdem Sie alle Sektionen hinzugefügt haben, die Sie einschließen möchten. Siehe Abschnitt 5.125 [easy:SetOpt_HTTPPost], Seite 102, für Details.

Sie müssen `form:Free()` aufrufen, nachdem der Form-Post ausgeführt wurde, um die Ressourcen freizugeben.

Die Verwendung von POST mit HTTP 1.1 impliziert die Verwendung eines Headers "Expect: 100-continue". Sie können diesen Header wie gewohnt mit `#CURLOPT_HTTPHEADER` deaktivieren.

Erstens gibt es einige Grundlagen, die Sie über Posts mit mehreren Multipart/Formdata verstehen müssen. Jeder Teil besteht aus mindestens einem `NAME-` und einem `CONTENTS-`Teil. Wenn der Teil zum Hochladen von Dateien erstellt wurde, gibt es auch einen gespeicherten `CONTENT-TYPE` und einen `FILENAME` (Dateiname). Im Folgenden werden die Optionen erläutert, mit denen Sie diese Eigenschaften in den Teilen festlegen, die Sie zu Ihrem Post hinzufügen möchten.

Das Argument `name` muss eine Zeichenkette sein, die den Namen von diesem Teil angibt. Der Name darf keine nullwertigen Bytes enthalten. Das Argument `content` muss die tatsächlich zu sendenden Daten enthalten. Das optionale Argument `type` kann verwendet werden, um den Content-Typ für den Teil festzulegen und das optionale Argument `headers` kann verwendet werden, um zusätzliche Header für die POST-Sektion des Formulars anzugeben. Dies erfordert eine Tabelle mit einer Liste und hängt die Liste der Header an die von libcurl automatisch erzeugten an.

EINGABEN

`name`	Name des Elements
`content`	tatsächliche zu sendende Daten
`type`	Optional: Content-Typ für das Element
`headers`	Optional: zusätzlichen Header für die POST-Sektion

6.3 form:AddFile

BEZEICHNUNG

form:AddFile – fügt eine Datei-Upload-Sektion zu einem Multipart/Formdata-HTTP-POST

ÜBERSICHT

```
form:AddFile(name, path[, type, filename, headers])
```

BESCHREIBUNG

`form:AddFile()` wird verwendet, um eine Datei-Upload-Sektion beim Erstellen eines Multipart/Formdata-HTTP-POST (Teilen/Formulardaten) (manchmal auch als RFC 2388-ähnlichen Post bezeichnet) anzuhängen. Nachdem Sie alle gewünschten Sektionen hinzugefügt haben, übergeben Sie den Form-Handle als Parameter an `#CURLOPT_HTTPPOST`. Siehe Abschnitt 5.125 [easy:SetOpt_HTTPPost], Seite 102, für Details.

Sie müssen `form:Free()` aufrufen, nachdem der Form-Post erstellt wurde, um die Ressourcen freizugeben.

Die Verwendung von POST mit HTTP 1.1 impliziert die Verwendung eines "Expect: 100-continue" Headers. Sie können diesen Header mit `#CURLOPT_HTTPHEADER` wie gewohnt deaktivieren.

Erstens gibt es einige Grundlagen, die Sie über Multipart/Formdata-Post verstehen müssen. Jeder Teil besteht aus mindestens einem `NAME-` und einem `CONTENTS`-Teil. Wenn der Teil für den Datei-Upload erstellt wird, gibt es auch einen gespeicherten `CONTENT-TYPE` und einen `FILENAME` (Dateiname). Im Folgenden werden wir besprechen, welche Optionen Sie verwenden, um diese Eigenschaften in den Teilen festzulegen, die Sie Ihrem Post hinzufügen möchten.

Das Argument `name` muss eine Zeichenkette sein, die den Namen dieses Elements liefert. Der Name darf keine nullwertigen Bytes enthalten.

Das Argument `path` muss auf den Pfad eines Dateinamens gesetzt werden, der hochgeladen werden soll. Libcurl setzt das Dateinamenfeld auf den Basisnamen des angegebenen Dateinamens, liest den Content der Datei und übergibt ihn als Daten und legt den Content-Typ fest, wenn die angegebene Datei mit einer der intern bekannten Dateierweiterungen übereinstimmt. Die angegebene Upload-Datei muss bereits zu Beginn des Uploads vollständig im Dateisystem vorhanden sein, da libcurl zuvor die richtige Dateigröße lesen muss. Die angegebene Datei muss beibehalten werden, bis die zugehörige Übertragung abgeschlossen ist.

Das optionale Argument `type` kann verwendet werden, um den Content-Typ für den Teil festzulegen. Das optionale Argument `filename` kann verwendet werden, um einen anderen Dateinamen als den von `path` abgeleiteten für den Upload zu verwenden. Mit dem optionalen Argument `headers` können zusätzliche Header für die POST-Sektion des Formulars angegeben werden. Dies erfordert eine Tabelle mit einer Liste und hängt die Liste des Headers an die von libcurl automatisch erzeugten an.

EINGABEN

`name`	Name des Elements
`path`	Pfad zu einer hochzuladenden Datei
`type`	Optional: Content-Typ für das Element
`filename`	Optional: Dateiname für den Content-Header
`headers`	Optional: zusätzlicher Header für die POST-Sektion

6.4 form:AddFiles

BEZEICHNUNG

form:AddFiles – fügt mehrere Datei-Upload-Sektionen zu einem HTTP-POST hinzu

ÜBERSICHT

```
form:AddFiles(name, table)
```

BESCHREIBUNG

`form:AddFiles()` wird verwendet, um mehrere Datei-Upload-Sektionen beim Erstellen eines Multipart/Formdata-HTTP-POST (manchmal auch als RFC 2388-ähnliche Post bezeichnet) anzuhängen. Nachdem Sie alle gewünschten Sektionen hinzugefügt haben, übergeben Sie das Formularelement als Parameter an `#CURLOPT_HTTPPOST`. Siehe Abschnitt 5.125 [easy:SetOpt_HTTPPost], Seite 102, für Details.

Sie müssen `form:Free()` aufrufen, nachdem der Form-Post erstellt wurde, um die Ressourcen freizugeben.

Die Verwendung von POST mit HTTP 1.1 impliziert die Verwendung eines "Expect: 100-continue" Headers. Sie können diesen Header mit `#CURLOPT_HTTPHEADER` wie gewohnt deaktivieren.

Erstens gibt es einige Grundlagen, die Sie über Multipart/Formdata-Post verstehen müssen. Jeder Teil besteht aus mindestens einem `NAME-` und einem `CONTENTS`-Teil. Wenn der Teil für den Datei-Upload gemacht wird, gibt es auch einen gespeicherten `CONTENT-TYPE` und einen `FILENAME` (Dateiname). Im Folgenden werden wir besprechen, welche Optionen Sie verwenden, um diese Eigenschaften in den Teilen festzulegen, die Sie Ihrem Post hinzufügen möchten.

Das Argument `name` muss eine Zeichenkette sein, die den Namen dieses Teils angibt. Der Name darf keine nullwertigen Bytes enthalten.

Das Argument `table` muss eine Tabelle enthalten, die eine Liste der Dateien beschreibt, die dem Form-Post-Objekt hinzugefügt werden sollen. In der Tabelle muss ein Element pro Datei vorhanden sein. Die einzelnen Tabellenelemente können drei verschiedene Typen haben:

1. Eine Zeichenkette: In diesem Fall muss die Zeichenkette lediglich den Pfad zu der hochzuladenden Datei enthalten.
2. Eine Tabelle mit zwei Zeichenketten: In diesem Fall muss die erste Zeichenkette den Pfad zu der hochzuladenden Datei enthalten und die zweite Zeichenkette muss den Content-Typ für die Datei enthalten.
3. Eine Tabelle mit drei Zeichenketten: Wie oben, aber die dritte Zeichenkette muss einen Dateinamen enthalten, der anstelle des Dateinamens für den Teil verwendet werden soll, der aus dem in der ersten Zeichenkette in der Tabelle angegebenen Pfad abgeleitet ist.

EINGABEN

`name`	Name des Elements
`table`	Tabelle mit den hochzuladenden Dateien (siehe oben)

6.5 form:AddStream

BEZEICHNUNG

form:AddStream – fügt Datenströme zu einem Multipart/Formdata-HTTP-POST hinzu

ÜBERSICHT

```
form:AddStream(name, len, func[, userdata, type, filename, headers])
```

BESCHREIBUNG

`form:AddStream()` wird zum Anhängen einer Datenstromsektion verwendet, wenn ein HTTP-POST mit Multipart/Formdata erstellt wird (manchmal als RFC 2388-Posts bezeichnet). Übergeben Sie das Formularelement als Parameter an `#CURLOPT_HTTPPOST`, nachdem Sie alle Sektionen hinzugefügt haben, die Sie einschließen möchten. Siehe Abschnitt 5.125 [easy:SetOpt_HTTPPost], Seite 102, für Details.

Sie müssen `form:Free()` aufrufen, nachdem der Formularbeitrag erstellt wurde, um die Ressourcen freizugeben.

Die Verwendung von POST mit HTTP 1.1 impliziert die Verwendung eines "Expect: 100-continue" Headers. Sie können diesen Header mit `#CURLOPT_HTTPHEADER` wie gewohnt deaktivieren.

Erstens gibt es einige Grundlagen, die Sie über Multipart/Formdata-Post verstehen müssen. Jeder Teil besteht aus mindestens einem `NAME-` und einem `CONTENTS`-Teil. Wenn der Teil für den Datei-Upload gemacht wird, gibt es auch einen gespeicherten `CONTENT-TYPE` und einen `FILENAME` (Dateiname). Im Folgenden werden wir besprechen, welche Optionen Sie verwenden, um diese Eigenschaften in den Teilen festzulegen, die Sie Ihrem Beitrag hinzufügen möchten.

Das Argument `name` muss eine Zeichenkette sein, die den Namen dieses Teils liefert. Der Name darf keine nullwertigen Bytes enthalten.

Der Parameter `len` muss die Anzahl der hinzuzufügenden Bytes enthalten. Der Parameter `func` muss ein Callback sein. Die Callback-Funktion wird aufgerufen, um die tatsächlichen Daten zum Hinzufügen bereitzustellen. Diese Callback-Funktion wird wiederholt aufgerufen, bis es genau `len` Bytes zurückgegeben hat. Die Callback-Funktion verhält sich genau wie der Befehl `#CURLOPT_READFUNCTION`. Siehe Abschnitt 5.202 [easy:SetOpt_ReadFunction], Seite 144, für Details. Wenn Sie das optionale Argument `userdata` übergeben, wird der Wert, der von jedem Typ sein kann, als zweiter Parameter an Ihre Callback-Funktion übergeben.

Das optionale Argument `type` kann verwendet werden, um den CONTENT-Typ für das Teil festzulegen. Das optionale Argument `filename` kann verwendet werden, um einen gewünschten Dateinamen für die Datenströme festzulegen. Das optionale Argument `headers` kann verwendet werden, um zusätzliche Header für die POST-Sektion des Formulars anzugeben. Dies erfordert eine Tabelle mit einer Liste und hängt die Liste der Headers an die von libcurl automatisch erzeugten an.

EINGABEN

`name`	Name des Elements
`len`	Anzahl der zu übermittelnden Bytes
`func`	Callback Funktion die den Datenstrom bereitstellt

`userdata` Optional: Benutzerdaten, die an die Callback-Funktion übergeben werden sollen

`type` Optional: Inhaltstyp für das Element

`filename` Optional: Dateiname für den Content-Header

`headers` Optional: zusätzlichen Header für die POST-Sektion

6.6 form:Free

BEZEICHNUNG

form:Free – löscht eine zuvor erstellten Multipart/Formdata-HTTP-POST

ÜBERSICHT

```
form:Free()
```

BESCHREIBUNG

`form:Free()` wird verwendet, um Daten zu bereinigen, die zuvor mit `hurl.Form()` erstellt/angehängt wurden. Dies muss aufgerufen werden, wenn die Daten verwendet wurden, was normalerweise bedeutet, dass `easy:Perform()` aufgerufen wurde.

EINGABEN

keine

6.7 form:Get

BEZEICHNUNG

form:Get – serialisiert eine zuvor erstellten Multipart/Formdata-HTTP-POST

ÜBERSICHT

```
s$ = form:Get()
form:Get(callback[, userdata])
```

BESCHREIBUNG

`form:Get()` wird verwendet, um Daten zu serialisieren, die zuvor mit `hurl.Form()` erstellt/angehängt wurden.

Es gibt zwei Möglichkeiten, diesen Befehl zu verwenden: Sie können ihn entweder ohne Argumente aufrufen. In diesem Fall wird eine Zeichenkette zurückgegeben, die die serialisierten Daten enthält. Alternativ können Sie auch eine Callback-Funktion übergeben. In diesem Fall wird die in `callback` übergebene Funktion von `form:Get()` aufgerufen und erhält die serialisierten Daten im ersten Parameter. Wenn Sie das optionale Argument `userdata` übergeben, wird der hier angegebene Wert (er kann von einem beliebigen Typ sein) als zweiter Parameter an Ihre Callback-Funktion übergeben.

EINGABEN

`callback` Callback-Funktion

`userdata` Optional: Benutzerdaten, die an die Callback-Funktion übergeben werden sollen

RÜCKGABEWERTE

`s$`	serialisierte Daten

7 Multi-Methoden

7.1 multi:AddHandle

BEZEICHNUNG

multi:AddHandle – fügt einen Easy-Handle zu einer Multi-Sitzung hinzu

ÜBERSICHT

```
multi:AddHandle(handle)
```

BESCHREIBUNG

Fügt dem Multi-Stack einen Easy-Handle hinzu. Dieser Befehlsaufruf macht diese Multi-Handle-Steuerung zum angegebenen Easy-Handle.

Während einem Multi-Stack ein Easy-Handle hinzugefügt wird, können und müssen Sie `easy:Perform()` für diesen Handle nicht verwenden. Nachdem Sie den Easy-Handle wieder vom Multi-Stack entfernt haben, können Sie ihn problemlos wieder mit der Easy-Schnittstelle verwenden.

Wenn der Easy-Handle nicht für die Verwendung eines Share- (`#CURLOPT_SHARE`) oder globalen DNS-Caches (`#CURLOPT_DNS_USE_GLOBAL_CACHE`) festgelegt ist, wird der DNS-Cache verwendet, der von allen Easy-Handle im Multi-Handle gemeinsam genutzt wird, wenn `multi:AddHandle()` aufgerufen wird.

Wenn einem Multi-Handle eine einfache Schnittstelle hinzugefügt wird, wird ein gemeinsamer Verbindungs-Cache verwendet, der dem Multi-Handle gehört. Das Entfernen und Hinzufügen neuer Easy-Handle hat keine Auswirkungen auf den Pool von Verbindungen oder die Möglichkeit, Verbindungen erneut zu verwenden.

Wenn Sie `#CURLMOPT_TIMERFUNCTION` im Multi-Handle gesetzt haben (und Sie sollten wirklich, wenn Sie ereignisbasiert mit `multi:SocketAction()` und so arbeiten), diesen Callback aus diesem Befehl heraus aufrufen, um nach einer Timer-Aktualisierung zu fragen, sodass Ihre Hauptereignisschleife die Aktivität auf diesem Handle zum Starten bringt.

Der Easy-Handle wird dem Multi-Handle hinzugefügt, bis Sie ihn mit `multi:RemoveHandle()` wieder entfernen - auch wenn eine Übertragung mit diesem speziellen Easy-Handle abgeschlossen ist.

Sie sollten den Easy-Handle aus dem Multi-Stack entfernen, bevor Sie zuerst den Easy-Handle und dann den Multi-Handle beenden:

1. `multi:RemoveHandle()`
2. `easy:Close()`
3. `multi:Close()`

EINGABEN

`handle` Easy-Handle zum Multi-Handle hinzufügen

7.2 multi:Close

BEZEICHNUNG

multi:Close – schließt eine Multi-Sitzung

ÜBERSICHT

```
multi:Close()
```

BESCHREIBUNG

Bereinigt und entfernt einen ganzen Multi-Stack. Dabei werden keine einzelnen Easy-Handle gelöscht oder berührt - sie müssen trotzdem einzeln geschlossen werden, und zwar auf die übliche `easy:Close()` Weise. Die Reihenfolge der Aufräumarbeiten sollte sein: Leert und entfernt einen ganzen Multi-Stack. Er gibt keine einzelne Easy-Handle frei oder greift auf sie in irgendeiner Weise zu - sie müssen immer noch einzeln geschlossen werden und zwar auf die übliche Weise mit `easy:Close()`. Die Reihenfolge der Bereinigung sollte sein:

1. `multi:RemoveHandle()` bevor irgendwelche Easy-Handle aufgeräumt werden.
2. `easy:Close()` kann nun unabhängig voneinander aufgerufen werden, da der Easy-Handle nicht mehr mit dem Multi-Handle verbunden ist.
3. `multi:Close()` sollte aufgerufen werden, wenn alle Easy-Handle entfernt sind.

EINGABEN

keine

7.3 multi:InfoRead

BEZEICHNUNG

multi:InfoRead – liest Multi-Informationen aus

ÜBERSICHT

```
msg, result, remaining, handle = multi:InfoRead()
```

BESCHREIBUNG

Fragt den Multi-Handle ab, ob Nachrichten/Informationen von den einzelnen Übertragungen vorhanden sind. Nachrichten können Informationen wie einen Fehlercode von der Übertragung oder nur die Tatsache enthalten, dass eine Übertragung abgeschlossen ist. Weitere Details dazu sollten ebenfalls notiert werden.

Dieser Aufruf gibt vier Werte zurück: `msg` enthält den Typ der empfangenen Nachricht. Dies kann `#CURLMSG_NONE` oder `#CURLMSG_DONE` sein. `result` enthält das Nachrichtenergebnis. Der Rückgabewert `remaining` gibt an, wie viele Nachrichten sich noch in der Warteschlange befinden, nachdem dieser Befehl aufgerufen wurde. Der Rückgabewert von `handle` enthält den Easy-Handle, der zuvor dem Multi-Handle hinzugefügt wurde.

Wenn Sie eine Nachricht mit diesem Befehl abrufen, wird sie aus der internen Warteschlange entfernt, sodass ein erneuter Aufruf von diesem Befehl nicht mehr dieselbe Nachricht zurückgibt. Stattdessen werden bei jedem neuen Aufruf neue Nachrichten zurückgegeben, bis die Warteschlange geleert ist.

Wenn `msg` den Wert `#CURLMSG_DONE` hat, gibt die Nachricht an, dass eine Übertragung durchgeführt wurde und `result` enthält den Rückkehrcode für den soeben abgeschlossenen Easy-Handle.

EINGABEN
keine

RÜCKGABEWERTE

`msg` Der zu lesende Nachrichtentyp (siehe oben für mögliche Typen)

`result` Nachrichtenspezifischer Ergebniscode

`remaining`
Anzahl verbleibender Nachrichten

`handle` der Easy-Handle, der zuvor zum Multi-Handle hinzugefügt wurde (V1.1)

7.4 multi:Perform

BEZEICHNUNG
multi:Perform – liest/schreibt verfügbare Daten von jedem Easy-Handle

ÜBERSICHT
```
running = multi:Perform()
```

BESCHREIBUNG
Dieser Befehl verarbeitet die Übertragung aller hinzugefügten Handle, die nicht blockiert werden müssen.

Wenn eine Anwendung herausgefunden hat, dass Daten für das Multi-Handle verfügbar sind oder eine Zeitüberschreitung verstrichen ist, sollte die Anwendung diese Funktion aufrufen, um alles zu lesen/schreiben, was gerade gelesen oder geschrieben werden muss usw. `multi:Perform()` kehrt zurück sobald die Lese-/Schreibvorgänge abgeschlossen sind. Diese Funktion setzt nicht voraus, dass tatsächlich Daten zum Lesen verfügbar sind oder dass Daten geschrieben werden können, sie kann nur für den Fall der Fälle aufgerufen werden. Es wird die Anzahl der Handles zurückgegeben, die noch Daten übertragen.

Wenn sich die Anzahl der ausgeführten Handle gegenüber dem vorherigen Aufruf geändert hat (oder die Anzahl der Easy-Handle, die Sie zum Multi-Handle hinzugefügt haben, unterschreitet), wissen Sie, dass eine oder mehrere Übertragungen weniger ausgeführt werden. Sie können dann `multi:InfoRead()` aufrufen, um Informationen zu jeder einzelnen abgeschlossenen Übertragung abzurufen. Diese zurückgegebenen Informationen enthalten CURL-Code und mehr. Wenn ein hinzugefügter Handle sehr schnell fehlschlägt, wird es möglicherweise nie als laufender Handle gezählt.

Wenn `running` bei der Rückkehr von diesem Befehl auf Null gesetzt wird, werden keine Übertragungen mehr ausgeführt.

EINGABEN
keine

RÜCKGABEWERTE

`running` Anzahl der laufenden Handle

7.5 multi:RemoveHandle

BEZEICHNUNG

multi:RemoveHandle – entfernt einen Easy-Handle aus einer Multi-Sitzung heraus

ÜBERSICHT

```
multi:RemoveHandle(handle)
```

BESCHREIBUNG

Entfernt einen bestimmten Handle aus dem Handle. Dadurch wird der angegebene Easy-Handle aus dem Steuerelement dieses Multi-Handle entfernt.

Wenn der Easy-Handle von einem Multi-Stack entfernt wurde, ist es wieder völlig legal, `easy:Perform()` für diesen Easy-Handle aufzurufen.

Das Entfernen eines Easy-Handle während der Verwendung ist völlig legal und stoppt die Übertragung mit diesem Easy-Handle effektiv. Alle anderen Easy-Handle und Übertragungen bleiben davon unberührt.

Es ist in Ordnung, ein Handle jederzeit während einer Übertragung zu entfernen, nur nicht innerhalb einer libcurl-Callbackfunktion.

EINGABEN

`handle` Easy-Handle zum Entfernen aus dem Multi-Handle

7.6 multi:SetOpt

BEZEICHNUNG

multi:SetOpt – legt Optionen für einen Curl-Multi-Handle fest

ÜBERSICHT

```
multi:SetOpt(option, param)
```

BESCHREIBUNG

`multi:SetOpt()` wird verwendet, um einem libcurl-Multi-Handle mitzuteilen, wie er sich verhalten soll. Mit den entsprechenden Optionen für `multi:SetOpt()` können Sie das Verhalten von libcurl bei Verwendung dieses Multi-Handles ändern. Alle Optionen werden mit der Option gefolgt vom Parameter `param` festgelegt. Dieser Parameter kann eine Zahl, eine Funktion, eine Zeichenkette oder eine Tabelle sein, je nachdem, was die jeweilige Option erwartet. Lesen Sie diese Befehle sorgfältig durch, da schlechte Eingabewerte dazu führen können, dass sich libcurl schlecht verhält! Sie können in jedem Befehlsaufruf nur eine Option einstellen.

Die folgenden Typen werden derzeit für `option` unterstützt:

`#CURLMOPT_CHUNK_LENGTH_PENALTY_SIZE`
: Setzt die Chunk-Längenschwelle für Pipelining. Siehe Abschnitt 7.7 [multi:SetOpt_Chunk_Length_Penalty_Size], Seite 283, für Details.

`#CURLMOPT_CONTENT_LENGTH_PENALTY_SIZE`
: Setzt die Inhaltsgrösse fürs Pipelining. Siehe Abschnitt 7.8 [multi:SetOpt_Content_Length_Penalty_Size], Seite 284, für Details.

`#CURLMOPT_MAXCONNECTS`
: Legt die Größe des Verbindungscaches fest. Siehe Abschnitt 7.9 [`multi:SetOpt_MaxConnects`], Seite 284, für Details.

`#CURLMOPT_MAX_HOST_CONNECTIONS`
: Legt die maximale Anzahl von Verbindungen zu einem einzelnen Host fest. Siehe Abschnitt 7.10 [`multi:SetOpt_Max_Host_Connections`], Seite 285, für Details.

`#CURLMOPT_MAX_PIPELINE_LENGTH`
: Setzt die maximale Anzahl von Anforderungen in einer Pipeline. Siehe Abschnitt 7.11 [`multi:SetOpt_Max_Pipeline_Length`], Seite 285, für Details.

`#CURLMOPT_MAX_TOTAL_CONNECTIONS`
: Setzt die maximal gleichzeitig offenen Verbindungen. Siehe Abschnitt 7.12 [`multi:SetOpt_Max_Total_Connections`], Seite 286, für Details.

`#CURLMOPT_PIPELINING`
: Aktiviert gleichzeitiges HTTP-Pipelining und Multiplexing. Siehe Abschnitt 7.13 [`multi:SetOpt_Pipelining`], Seite 286, für Details.

`#CURLMOPT_PIPELINING_SERVER_BL`
: Setzt die Schwarze Liste von Pipeline-Servern. Siehe Abschnitt 7.14 [`multi:SetOpt_Pipelining_Server_Bl`], Seite 287, für Details.

`#CURLMOPT_PIPELINING_SITE_BL`
: Setzt die Schwarze Liste von Hosts. Siehe Abschnitt 7.15 [`multi:SetOpt_Pipelining_Site_Bl`], Seite 288, für Details.

`#CURLMOPT_SOCKETFUNCTION`
: Setzt den Callback für Information, worauf gewartet werden soll. Siehe Abschnitt 7.16 [`multi:SetOpt_SocketFunction`], Seite 288, für Details.

`#CURLMOPT_TIMERFUNCTION`
: Setzt den Callback, um Timeout-Werte zu erhalten. Siehe Abschnitt 7.17 [`multi:SetOpt_TimerFunction`], Seite 289, für Details.

EINGABEN

`option` zu setzender Optionstyp

`parameter`
: Wert, auf den die Option gesetzt werden soll

7.7 multi:SetOpt_Chunk_Length_Penalty_Size

BEZEICHNUNG

multi:SetOpt_Chunk_Length_Penalty_Size – setzt die Chunk-Längenschwelle für Pipelining

ÜBERSICHT

```
multi:SetOpt_Chunk_Length_Penalty_Size(size)
```

BESCHREIBUNG

Übergeben Sie eine Zahl mit einer Größe in Byte. Wenn eine durchführende Verbindung derzeit eine Teilanforderung (Transfer-encoding: chunked) mit einer aktuellen Teillänge größer als `#CURLMOPT_CHUNK_LENGTH_PENALTY_SIZE` verarbeitet, wird diese Durchführung nicht für zusätzliche Anforderungen berücksichtigt, auch wenn sie kürzer als `#CURLMOPT_MAX_PIPELINE_LENGTH` ist.

EINGABEN

`size` Eingabewert

7.8 multi:SetOpt_Content_Length_Penalty_Size

BEZEICHNUNG

multi:SetOpt_Content_Length_Penalty_Size – setzt die Inhaltsgrösse fürs Pipelining

ÜBERSICHT

```
multi:SetOpt_Content_Length_Penalty_Size(size)
```

BESCHREIBUNG

Übergeben Sie eine Zahl mit einer Größe in Byte. Wenn eine durchführende Verbindung derzeit eine Anforderung mit einer Inhaltslänge verarbeitet, die größer als `#CURLMOPT_CONTENT_LENGTH_PENALTY_SIZE` ist, wird diese Durchführung für zusätzliche Anforderungen nicht berücksichtigt, auch wenn sie kürzer als ist `#CURLMOPT_MAX_PIPELINE_LENGTH`.

EINGABEN

`size` Eingabewert

7.9 multi:SetOpt_MaxConnects

BEZEICHNUNG

multi:SetOpt_MaxConnects – legt die Größe des Verbindungscaches fest

ÜBERSICHT

```
multi:SetOpt_MaxConnects(max)
```

BESCHREIBUNG

Übergeben Sie eine Zahl, die den maximalen Wert angibt. Die eingestellte Anzahl wird als die maximale Anzahl gleichzeitig offener Verbindungen verwendet, die libcurl nach Beendigung der Nutzung in seinem Verbindungscache halten kann. Standardmäßig vergrößert libcurl die Größe für jeden zusätzlichen Easy-Handle, so dass er das Vierfache der Anzahl der zusätzlichen Easy-Handle ausmacht.

Wenn Sie diese Option aktivieren, können Sie verhindern, dass die Cache-Größe über die von Ihnen festgelegte Grenze hinaus wächst.

Wenn der Cache voll ist, schließt curl die älteste im Cache, um zu verhindern, dass die Anzahl der offenen Verbindungen zunimmt.

Diese Option ist nur für die Verwendung des Multi-Handle gedacht, bei Verwendung der Easy-Schnittstelle sollten Sie stattdessen die Option `#CURLOPT_MAXCONNECTS` verwenden.

Siehe `#CURLMOPT_MAX_TOTAL_CONNECTIONS`, um die Anzahl der aktiven Verbindungen zu begrenzen.

EINGABEN

`max` Eingabewert

7.10 multi:SetOpt_Max_Host_Connections

BEZEICHNUNG

multi:SetOpt_Max_Host_Connections – legt die maximale Anzahl von Verbindungen zu einem einzelnen Host fest

ÜBERSICHT

```
multi:SetOpt_Max_Host_Connections(max)
```

BESCHREIBUNG

Die in `max` eingestellte Nummer wird als maximale Anzahl von gleichzeitig geöffneten Verbindungen zu einem einzelnen Host verwendet (ein Host ist identisch mit einem Hostnamen + Portnummernpaar). Für jede neue Sitzung auf einem Host öffnet libcurl eine neue Verbindung bis zu dem von `#CURLMOPT_MAX_HOST_CONNECTIONS` festgelegten Limit. Wenn das Limit erreicht ist, sind die Sitzungen ausstehend, bis eine Verbindung verfügbar wird. Wenn `#CURLMOPT_PIPELINING` aktiviert ist, wird libcurl versuchen, eine Durchführung zu erstellen, wenn die Option host dazu in der Lage ist.

Der voreingestellte Maximalwert ist 0, unbegrenzt. Aus Gründen der Abwärtskompatibilität wird das Setzen auf 0, wenn `#CURLMOPT_PIPELINING` 1 ist, jedoch nicht als unbegrenzt behandelt. Stattdessen öffnet es nur 1 Verbindung und versucht, sie mit einer Pipeline zu verbinden.

Dieses festgelegte Limit wird auch für Proxy-Verbindungen verwendet. Anschließend wird der Proxy als der Host betrachtet, für den dieses Limit gilt.

EINGABEN

`max` Eingabewert

7.11 multi:SetOpt_Max_Pipeline_Length

BEZEICHNUNG

multi:SetOpt_Max_Pipeline_Length – setzt die maximale Anzahl von Anforderungen in einer Pipeline

ÜBERSICHT

```
multi:SetOpt_Max_Pipeline_Length(max)
```

BESCHREIBUNG

Die in `max` eingestellte Zahl wird als maximale Anzahl ausstehender Anfragen in einer HTTP/1.1-Pipeline-Verbindung verwendet. Diese Option wird nur für HTTP/1.1 Pipeline verwendet, nicht für HTTP/2 Multiplexing.

Wenn diese Grenze erreicht ist, verwendet libcurl eine andere Verbindung zum gleichen Host (siehe `#CURLMOPT_MAX_HOST_CONNECTIONS`) oder stellt die Anforderung in

eine Warteschlange, bis eine der Verbindungen zum Host bereit ist, eine Anforderung anzunehmen. Somit ist die Gesamtzahl der Anfragen während des Betriebs `#CURLMOPT_MAX_HOST_CONNECTIONS` * `#CURLMOPT_MAX_PIPELINE_LENGTH`.

EINGABEN

`max` Eingabewert

7.12 multi:SetOpt_Max_Total_Connections

BEZEICHNUNG

multi:SetOpt_Max_Total_Connections – setzt die maximal gleichzeitig offenen Verbindungen

ÜBERSICHT

```
multi:SetOpt_Max_Total_Connections(amount)
```

BESCHREIBUNG

Übergeben Sie eine Zahl in `amount`. Die eingestellte Anzahl wird als die maximale Anzahl gleichzeitig geöffneter Verbindungen insgesamt verwendet, die diesen Multi-Handle verwenden. Für jede neue Sitzung öffnet libcurl eine neue Verbindung bis zu dem durch `#CURLMOPT_MAX_TOTAL_CONNECTIONS` festgelegten Limit. Wenn das Limit erreicht ist, stehen die Sitzungen an, bis Verbindungen verfügbar sind. Wenn `#CURLMOPT_PIPELINING` aktiviert ist, versucht libcurl, eine Pipeline zu erstellen oder Multiplexing zu verwenden, wenn der Host dazu in der Lage ist.

EINGABEN

`amount` Eingabewert

7.13 multi:SetOpt_Pipelining

BEZEICHNUNG

multi:SetOpt_Pipelining – aktiviert gleichzeitiges HTTP-Pipelining und Multiplexing

ÜBERSICHT

```
multi:SetOpt_Pipelining(bitmask)
```

BESCHREIBUNG

Übergeben Sie den Parameter `bitmask`, um libcurl anzuweisen, das HTTP-Piplining und/oder HTTP/2-Multiplexing für diesen Multi-Handle zu aktivieren.

Wenn diese Option aktiviert ist, versucht libcurl, diese Protokollfunktionen zu verwenden, wenn parallele Anforderungen an dieselben Hosts gesendet werden.

Bei Pipelining bedeutet dies, dass beim Hinzufügen einer zweiten Anforderung, die eine bereits vorhandene Verbindung verwenden kann, die zweite Anforderung auf derselben Verbindung "weitergeleitet" wird, anstatt parallel ausgeführt zu werden.

Für das Multiplexing bedeutet dies, dass Folge-Anforderungen eine vorhandene Verbindung wiederverwenden und die neue Anforderung gleichzeitig übertragen können, während andere Übertragungen diese einzelne Verbindung bereits verwenden.

Es gibt mehrere andere verwandte Optionen, die interessant zu optimieren und anzupassen sind, um zu ändern, wie libcurl Anforderungen auf verschiedene Verbindungen verteilt oder nicht usw.

Vor 7.43.0 wurde diese Option auf 1 und 0 gesetzt, um die HTTP/1.1-Pipelining zu aktivieren und zu deaktivieren.

Ab 7.43.0 hat auch das zweite Bit der Bitmaske eine Bedeutung und Sie können unabhängig voneinander Pipelining und Multiplexing anfordern, indem Sie die richtigen Bits umschalten.

`#CURLPIPE_NOTHING`
: Standard, d.h. keine Versuche für Pipelining oder Multiplexing.

`#CURLPIPE_HTTP1`
: Wenn dieses Bit gesetzt ist, versucht libcurl, HTTP/1.1-Anfragen an bereits bestehende und verwendete Verbindungen zu Hosts zu leiten. Dieses Bit ist veraltet und hat seit der Version 7.62.0 keine Wirkung mehr.

`#CURLPIPE_MULTIPLEX`
: Wenn dieses Bit gesetzt ist, wird libcurl versuchen, die neue Übertragung nach Möglichkeit über eine bestehende Verbindung zu übertragen. Dazu ist HTTP/2 erforderlich.

EINGABEN

`bitmask` Eingabewert

7.14 multi:SetOpt_Pipelining_Server_Bl

BEZEICHNUNG

multi:SetOpt_Pipelining_Server_Bl – setzt die Schwarze Liste von Pipeline-Servern

ÜBERSICHT

```
multi:SetOpt_Pipelining_Server_Bl(servers)
```

BESCHREIBUNG

Übergeben Sie hier eine Tabelle mit einer Liste von Zeichenketten. Dies ist eine Liste von Servertyp-Präfixen (Server: HTTP-Header), die von Pipelining ausgeschlossen sind, d.h. Servertypen, von denen bekannt ist, dass sie kein HTTP-Pipelining unterstützen.

Beachten Sie, dass der Vergleich übereinstimmt, wenn die Server: Header mit der Zeichenkette in der Schwarzen Liste beginnt, d.h. "Server: Ninja 1.2.3" und "Server: Ninja 1.4.0" können beide auf die Schwarze Liste gesetzt werden, indem "Ninja" in der Schwarzen Liste steht.

Übergeben Sie eine leere Tabelle, um die schwarze Liste zu löschen.

EINGABEN

`servers` Eingabewert

7.15 multi:SetOpt_Pipelining_Site_Bl

BEZEICHNUNG

multi:SetOpt_Pipelining_Site_Bl – setzt die Schwarze Liste von Hosts

ÜBERSICHT

```
multi:SetOpt_Pipelining_Site_Bl(hosts)
```

BESCHREIBUNG

Übergeben Sie hier eine Tabelle mit einer Liste von Zeichenketten. Dies ist eine Liste von Webseiten, die auf die Schwarze Liste gesetzt werden, d.h. Webseiten, von denen bekannt ist, dass sie HTTP-Pipelining nicht unterstützen.

Übergeben Sie eine leere Tabelle, um die schwarze Liste zu löschen.

EINGABEN

`hosts` Eingabewert

7.16 multi:SetOpt_SocketFunction

BEZEICHNUNG

multi:SetOpt_SocketFunction – setzt den Callback für Information, worauf gewartet werden soll

ÜBERSICHT

```
multi:SetOpt_SocketFunction(socket_callback[, userdata])
```

BESCHREIBUNG

Übergibt eine Callback-Funktion.

Wenn die Funktion `multi:SocketAction()` ausgeführt wird, informiert sie die Anwendung über Aktualisierungen im Socket-Status (Datei-Deskriptor), indem sie keine, einen oder mehrere Aufrufe an den Socket-Callback durchführen. Der Callback erhält Statusaktualisierungen mit Änderungen gegenüber dem vorherigen Zeitpunkt des Callbacks.

Der Callback erhält drei Argumente: Das erste Argument ist ein Easy-Handle, das zweite Argument ist ein Socket-Deskriptor und das dritte Argument informiert den Callback über den Status des gegebenen Sockets. Es kann einen dieser Werte enthalten:

`#CURL_POLL_IN`
: Wartet auf eingehende Daten. Damit der Socket lesbar wird.

`#CURL_POLL_OUT`
: Wartet auf ausgehende Daten. Damit der Socket beschreibbar wird.

`#CURL_POLL_INOUT`
: Wartet auf eingehende und ausgehende Daten. Damit der Socket lesbar oder beschreibbar wird.

`#CURL_POLL_REMOVE`
: Der angegebene Socket/Datei-Deskriptor wird von libcurl nicht mehr verwendet.

Wenn Sie das optionale Argument `userdata` übergeben, wird der Wert, den Sie in `userdata` übergeben, als vierter Parameter an Ihre Callback-Funktion übergeben. Der Parameter `userdata` kann von beliebigem Typ sein.

EINGABEN

`socket_callback`
: Eingabewert

`userdata`
: Optional: Benutzerdaten, die an die Callback-Funktion übergeben werden sollen

7.17 multi:SetOpt_TimerFunction

BEZEICHNUNG

multi:SetOpt_TimerFunction – setzt den Callback, um Timeout-Werte zu erhalten

ÜBERSICHT

```
multi:SetOpt_TimerFunction(timer_callback[, userdata])
```

BESCHREIBUNG

Übergibt eine Callback-Funktion.

Bestimmte Funktionen, wie Zeitüberschreitungs- und Wiederholungsversuche, erfordern den Aufruf von libcurl, auch wenn es keine Aktivität auf den Dateideskriptoren gibt.

Ihre Callback-Funktion erhält einen einzigen Parameter namens `timeout_ms`. Wenn Sie das optionale Argument `userdata` übergeben, wird der Wert, den Sie in `userdata` übergeben, als zweiter Parameter an Ihre Callback-Funktion übergeben. Der Parameter `userdata` kann von beliebigem Typ sein.

Nach dem Aufruf sollte Ihre Callback-Funktion einen sich nicht wiederholenden Timer mit einem Intervall von `timeout_ms` installieren. Jedes Mal, wenn der Timer ausgelöst wird, rufen Sie entweder `multi:SocketAction()` oder `multi:Perform()` auf, je nachdem, welche Schnittstelle Sie verwenden.

Ein `timeout_ms-Wert` von -1 bedeutet, dass Sie Ihren Timer löschen sollten.

Ein `timeout_ms-Wert` von 0 bedeutet, dass Sie so schnell wie möglich `multi:SocketAction()` oder `multi:Perform()` (einmal) aufrufen sollten.

Der Timer-Callback wird nur dann aufgerufen, wenn sich `timeout_ms` ändert.

Der Timer-Callback sollte bei Erfolg 0 und bei Fehler -1 zurückgeben. Dieser Callback kann anstelle von oder zusätzlich zu `multi:Timeout()` verwendet werden.

EINGABEN

`timer_callback`
: Eingabewert

`userdata`
: Optional: Benutzerdaten, die an die Callback-Funktion übergeben werden sollen

7.18 multi:SocketAction

BEZEICHNUNG

multi:SocketAction – liest/schreibt verfügbare Daten bei einer Aktion

ÜBERSICHT

```
running = multi:SocketAction(socket, mask)
```

BESCHREIBUNG

Wenn die Anwendung eine Aktion für einen von libcurl behandelten Socket erkannt hat, sollte sie `multi:SocketAction()` aufrufen, wobei das Argument `socket` auf den Socket mit der Aktion festgelegt ist. Wenn die Ereignisse auf einem Socket bekannt sind, können sie als Ereignisbitmaske `maske` übergeben werden, indem zuerst `maske` auf 0 gesetzt und dann mit bitweisem ODER (`|`) eine beliebige Kombination von Ereignissen hinzugefügt wird, die aus `#CURL_CSELECT_IN`, `#CURL_CSELECT_OUT` oder `#CURL_CSELECT_ERR` ausgewählt werden soll. Wenn die Ereignisse an einem Socket unbekannt sind, übergeben Sie stattdessen 0, und libcurl testet den Deskriptor intern. Es ist auch zulässig, `#CURL_SOCKET_TIMEOUT` an den Parameter `socket` zu übergeben, um den gesamten Prozess einzuleiten oder wenn eine Zeitüberschreitung auftritt.

Bei der Rückkehr enthält `running` die Anzahl der laufenden Easy-Handle innerhalb des Multi-Handle. Wenn diese Zahl Null erreicht, sind alle Übertragungen abgeschlossen. Wenn Sie `multi:SocketAction()` auf einem bestimmten Socket aufrufen und der Zähler um eins abnimmt, bedeutet das NICHT unbedingt, dass dieser genau Socket/Transfer derjenige ist, der abgeschlossen wurde. Verwenden Sie `multi:InfoRead()`, um herauszufinden, welcher Easy-Handle das erledigt hat.

Der Befehl `multi:SocketAction()` informiert die Anwendung über Aktualisierungen des Socket-Status (Dateideskriptor), indem sie keine, einen oder mehrere Aufrufe der Socket-Callback-Funktion ausführen, die mit der Option `#CURLMOPT_SOCKETFUNCTION` auf `multi:SetOpt()` gesetzt wurde. Sie aktualisieren den Status mit Änderungen seit dem letzten Aufruf des Callbacks.

Rufen Sie die Timeout-Zeit ab, indem Sie die Option `#CURLMOPT_TIMERFUNCTION` mit `multi:SetOpt()` setzen. Ihre Anwendung wird dann mit der Information aufgerufen, wie lange Sie höchstens auf Socket-Aktionen warten müssen, bevor Sie die Zeitüberschreitungs-Aktion durchführen: Rufen Sie den Befehl `multi:SocketAction()` mit dem Argument `socket` auf, das auf `#CURL_SOCKET_TIMEOUT` gesetzt ist. Sie können auch den Befehl `multi:Timeout()` aufrufen, um den Wert zu einem bestimmten Zeitpunkt abzufragen. Aber für ein ereignisbasiertes System, das den Callback verwendet, ist viel besser, als sich auf die Abfrage des Zeitüberschreitungs-Wertes zu verlassen.

EINGABEN

`socket` zu verwendender Socket

`mask` zu verwendende Maske

RÜCKGABEWERTE

`running` Anzahl der laufenden Zugriffe

7.19 multi:Timeout

BEZEICHNUNG

multi:Timeout – stellt die Wartezeit auf Socket-Aktion ein, bevor fortgefahren wird

ÜBERSICHT

```
ms = multi:Timeout()
```

BESCHREIBUNG

Eine Anwendung, die die libcurl Multi-Schnittstelle verwendet, sollte `multi:Timeout()` aufrufen, um herauszufinden, wie lange auf Socket-Aktionen gewartet werden soll, bevor Sie fortfahren.

Zum Fortfahren müssen Sie entweder die Zeitüberschreitungsaktion im Socket-Stil ausführen: Rufen Sie die Funktion `multi:SocketAction()` mit dem Argument `sockfd` auf `#CURL_SOCKET_TIMEOUT` auf, oder rufen Sie `multi:Perform()` auf, wenn Sie die Einfacheren und älteren Multi-Interface-Ansätze verwenden.

Der zurückgegebene Zeitüberschreitungswert wird in diesem Moment in Millisekunden angegeben. Wenn 0, bedeutet dies, dass Sie sofort fortfahren sollten, ohne auf etwas zu warten. Wenn -1 zurückgegeben wird, ist überhaupt keine Zeitüberschreitung festgelegt.

Eine Anwendung, die die Multi_Socket-API verwendet, SOLLTE diese Funktion NICHT verwenden, sondern stattdessen `multi:SetOpt()` und die zugehörige Option `#CURLMOPT_TIMERFUNCTION` benutzen, um das richtige und gewünschte Verhalten zu erzielen.

Hinweis: Wenn libcurl hier eine Zeitüberschreitung von -1 zurückgibt, bedeutet dies nur, dass in libcurl derzeit keine Zeitüberschreitung gespeichert ist. Sie dürfen nicht zu lange (vielleicht länger als ein paar Sekunden) warten, bevor Sie `multi:Perform()` erneut aufrufen.

EINGABEN

keine

RÜCKGABEWERTE

`ms`	aktueller Zeitüberschreitungswert

7.20 multi:Wait

BEZEICHNUNG

multi:Wait – fragt alle Easy-Handle in einem Multi-Handle ab

ÜBERSICHT

```
multi:Wait(timeout_ms)
```

BESCHREIBUNG

`multi:Wait()` fragt alle Dateideskriptoren ab, die von den im angegebenen Multi-Handle-Set enthaltenen Curl-Easy-Handles verwendet werden. Es wird blockiert, bis eine Aktivität an mindestens einem der Handles erkannt wird oder `timeout_ms` verstrichen ist. Wenn das Multi-Handle einen anstehenden internen Timeout hat, der eine kürzere Ablaufzeit als `timeout_ms` hat, wird diese kürzere Zeit stattdessen verwendet, um sicherzustellen, dass die Timeout-Genauigkeit angemessen eingehalten wird.

EINGABEN

`timeout_ms`
: maximale Wartezeit (in Millisekunden)

8 Share-Methoden

8.1 share:Close

BEZEICHNUNG
share:Close – bereinigt ein Share-Objekt

ÜBERSICHT
`share:Close()`

BESCHREIBUNG
Diese Funktion löscht ein Share-Objekt. Der Share-Handle kann nicht mehr verwendet werden, wenn diese Funktion aufgerufen wurde.

EINGABEN
keine

8.2 share:SetOpt

BEZEICHNUNG
share:SetOpt – legt Optionen für ein Share-Objekt fest

ÜBERSICHT
`share:SetOpt(option, parameter)`

BESCHREIBUNG
Setzen Sie die `option` auf `parameter` für das angegebene Share-Objekt.

Die folgenden Optionstypen werden derzeit für `option` unterstützt:

`#CURLSHOPT_SHARE`
: Legt den Typ der gemeinsam zu nutzenden Daten fest. Siehe Abschnitt 8.3 [share:SetOpt_Share], Seite 293, für Details.

`#CURLSHOPT_UNSHARE`
: Legt den nicht gemeinsam genutzter Datentyp fest. Siehe Abschnitt 8.4 [share:SetOpt_Unshare], Seite 295, für Details.

EINGABEN

`option` — einzustellender Optionstyp

`parameter`
: Wert, auf den die Option gesetzt werden soll

8.3 share:SetOpt_Share

BEZEICHNUNG
share:SetOpt_Share – legt den Typ der gemeinsam zu nutzenden Daten fest

ÜBERSICHT
`share:SetOpt_Share(type)`

BESCHREIBUNG

Der Parameter `type` gibt einen Datentyp an, der gemeinsam genutzt werden soll. Dieser kann auf einen der unten beschriebenen Werte gesetzt werden.

`#CURL_LOCK_DATA_COOKIE`

Cookie-Daten werden über den Easy-Handle mit diesem Share-Objekt geteilt.

`#CURL_LOCK_DATA_DNS`

Mit diesem Share-Objekt werden zwischengespeicherte DNS-Hosts über den Easy-Handle hinweg freigegeben. Beachten Sie, dass bei Verwendung der Multi-Schnittstelle alle Easy-Handle, die demselben Multi-Handle hinzugefügt wurden, standardmäßig den DNS-Cache gemeinsam nutzen, ohne diese Option zu verwenden.

`#CURL_LOCK_DATA_SSL_SESSION`

SSL-Sitzungs-IDs werden mithilfe dieses Share-Objekts für alle Easy-Handles freigegeben. Dadurch wird der Zeitaufwand für den SSL-Neuaufbau beim erneuten herstellen einer Verbindung mit demselben Server verringert. Hinweis: SSL-Sitzungs-IDs werden standardmäßig im selben Easy-Handle wiederverwendet. Beachten Sie, dass dieses Symbol in 7.10.3 hinzugefügt wurde, aber erst in 7.23.0 implementiert wurde.

`#CURL_LOCK_DATA_CONNECT`

Fügen Sie den Verbindungscache in das Share-Objekt ein und stellen Sie sicher, dass alle Easy-Handle, die dieses Share-Objekt verwenden, den Verbindungscache gemeinsam nutzen. Auf diese Weise können Sie zum Beispiel mit Multi-Threaded-libcurl mit einem Handle in jedem Thread verwenden und dennoch einen gemeinsamen Pool nicht verwendeter Verbindungen haben. Auf diese Weise wird die Wiederverwendung von Verbindungen wesentlich besser, als wenn Sie in jedem Thread einen separaten Pool verwenden.

Verbindungen, die für HTTP/1.1-Pipeline oder HTTP/2-Multiplexing verwendet werden, werden nur dann zusätzliche Übertragungen hinzugefügt, wenn die vorhandene Verbindung von demselben Multi-Handle oder Easy-Handle gehalten wird. libcurl unterstützt keine HTTP/2-Datenströme in verschiedenen Threads über eine gemeinsam genutzte Verbindung.

Beachten Sie, dass, wenn Sie die Multi-Schnittstelle verwenden, alle Easy-Handle, die demselben Multi-Handle hinzugefügt werden, standardmäßig den Verbindungscache gemeinsam nutzen, ohne diese Option zu verwenden.

`#CURL_LOCK_DATA_PSL`

Die im Share-Objekt gespeicherte öffentliche Suffix-Liste wird allen an die später gebundenen Easy-Handle zur Verfügung gestellt. Da die öffentliche Suffix-Liste regelmäßig aktualisiert wird, vermeidet dies Aktualisierungen in zu vielen verschiedenen Kontexten. Beachten Sie, dass bei Verwendung der Mehrfach-Schnittstelle alle Easy-Handle, die demselben Multi-Handle hinzugefügt werden, standardmäßig den PSL-Cache gemeinsam nutzen, ohne diese Option zu verwenden.

EINGABEN

`type` gewünschter Typ (siehe oben)

8.4 share:SetOpt_Unshare

BEZEICHNUNG

share:SetOpt_Unshare – legt den nicht gemeinsam genutzter Datentyp fest

ÜBERSICHT

```
share:SetOpt_Unshare(type)
```

BESCHREIBUNG

Diese Option bewirkt das Gegenteil von `#CURLSHOPT_SHARE`. Sie gibt an, dass der angegebene Parameter nicht mehr freigegeben wird. Gültige Werte sind dieselben wie für `#CURLSHOPT_SHARE`. Siehe Abschnitt 8.3 [share:SetOpt_Share], Seite 293, für Details.

EINGABEN

`type` gewünschter Typ (siehe oben)

Anhang A Lizenzen

A.1 Curl-Lizenz

A.2 LuaCurl-Lizenz

A.3 OpenSSL-Lizenz

texts. Actually both licenses are BSD-style Open Source licenses. In case of any license issues related to OpenSSL please contact openssl-core@openssl.org.

OpenSSL License

Copyright (c) 1998-2018 The OpenSSL Project. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. All advertising materials mentioning features or use of this software must display the following acknowledgment: "This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit. (http://www.openssl.org/)"
4. The names "OpenSSL Toolkit" and "OpenSSL Project" must not be used to endorse or promote products derived from this software without prior written permission. For written permission, please contact openssl-core@openssl.org.
5. Products derived from this software may not be called "OpenSSL" nor may "OpenSSL" appear in their names without prior written permission of the OpenSSL Project.
6. Redistributions of any form whatsoever must retain the following acknowledgment: "This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit (http://www.openssl.org/)"

THIS SOFTWARE IS PROVIDED BY THE OpenSSL PROJECT "AS IS" AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE OpenSSL PROJECT OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

This product includes cryptographic software written by Eric Young (eay@cryptsoft.com). This product includes software written by Tim Hudson (tjh@cryptsoft.com).

Original SSLeay License

Copyright (C) 1995-1998 Eric Young (eay@cryptsoft.com) All rights reserved.

This package is an SSL implementation written by Eric Young (eay@cryptsoft.com). The implementation was written so as to conform with Netscapes SSL.

This library is free for commercial and non-commercial use as long as the following conditions are aheared to. The following conditions apply to all code found in this distribution,

be it the RC4, RSA, lhash, DES, etc., code; not just the SSL code. The SSL documentation included with this distribution is covered by the same copyright terms except that the holder is Tim Hudson (tjh@cryptsoft.com).

Copyright remains Eric Young's, and as such any Copyright notices in the code are not to be removed. If this package is used in a product, Eric Young should be given attribution as the author of the parts of the library used. This can be in the form of a textual message at program startup or in documentation (online or textual) provided with the package.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. All advertising materials mentioning features or use of this software must display the following acknowledgement: "This product includes cryptographic software written by Eric Young (eay@cryptsoft.com)" The word 'cryptographic' can be left out if the routines from the library being used are not cryptographic related :-).
4. If you include any Windows specific code (or a derivative thereof) from the apps directory (application code) you must include an acknowledgement: "This product includes software written by Tim Hudson (tjh@cryptsoft.com)"

THIS SOFTWARE IS PROVIDED BY ERIC YOUNG "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

The licence and distribution terms for any publically available version or derivative of this code cannot be changed. i.e. this code cannot simply be copied and put under another distribution licence [including the GNU Public Licence.]

A.4 libssh2 license

Copyright (c) 2004-2007 Sara Golemon <sarag@libssh2.org>

Copyright (c) 2005,2006 Mikhail Gusarov <dottedmag@dottedmag.net>

Copyright (c) 2006-2007 The Written Word, Inc.

Copyright (c) 2007 Eli Fant <elifantu@mail.ru>

Copyright (c) 2009-2014 Daniel Stenberg

Index

E

F

H

M

S