

ERiC API-Referenz

Version 43.3.2.0

Inhalt

START	4
Suchfunktion	4
Dokumentation	4
Encoding und Zeichensatz	5
DATENSTRUKTUR-VERZEICHNIS	6
DATEI-VERZEICHNIS	7
DATENSTRUKTUR-DOKUMENTATION	8
eric_druck_parameter_t.....	8
eric_verschluesselungs_parameter_t.....	12
eric_zertifikat_parameter_t.....	14
OttoProxyKonfiguration	18
DATEI-DOKUMENTATION.....	21
eric_fehlercodes.h	21
ericapi.h	50
Inhalt des Rückgabepuffers und des Serverantwortpuffers	60
Erfolgsfall.....	61
Hinweise.....	61
Plausibilitätsfehler	62
Fehler in der Serverantwort.....	62
Sonstige Fehler	63
Fortschrittcallbacks.....	63
ericmtapi.h	127
Inhalt des Rückgabepuffers und des Serverantwortpuffers	137
Erfolgsfall.....	138
Hinweise.....	138
Plausibilitätsfehler	139
Fehler in der Serverantwort.....	139
Sonstige Fehler	140
Fortschrittcallbacks.....	140
ericversion.h	209
erictoolkit.h	210
otto.h.....	221
otto_statuscode.h	259
otto_types.h	270
eric_types.h	275

ericapiExport.h.....	284
ericdef.h.....	286
platform.h.....	288
INDEX.....	290

Start

Diese API-Referenz enthält detaillierte Informationen der ERiC API-Funktionen, Typdefinitionen, Aufzählungen, Datenstrukturen und Headerdateien. Die Funktionsdeklarationen für die ERiC Multithreading-API werden in [ericmtapi.h](#), die Deklarationen der Singlethreading-API in [ericapi.h](#) bereitgestellt.

In [erictoolkit.h](#) werden Prüffunktionen bereitgestellt, deren Funktionalität identisch zu denen in [ericapi.h](#) und [ericmtapi.h](#) ist. Die [erictoolkit.h](#) hat keine Abhängigkeiten zu anderen ERiC-Bibliotheken und kann somit unabhängig von diesen eingesetzt werden.

Suchfunktion

Die ERiC API-Referenz im HTML-Format enthält ein Suchfeld mit integrierter Filterfunktion. Damit können Sie eine gezielte Suche nach Symbolen durchführen. Voraussetzung: In Ihrem Browser ist JavaScript aktiviert.

Für eine Volltextsuche nutzen Sie bitte die PDF-Ausgabe der ERiC API-Referenz.

Dokumentation

Das Dokumentationspaket beinhaltet:

- Das [ERiC-Entwicklerhandbuch.pdf](#). Hier finden Sie sowohl allgemeine Zusatzinformationen als auch spezielle Hinweise zum Gebrauch der Bibliotheken, Datensätze, Datensatzformate und Werte.
- Das [ERiC-Tutorial.pdf](#). Es illustriert detailliert die Softwareentwicklung mit ERiC an den mitgelieferten Beispielen `ericdemo` und `ottodemo`.
- Die [ERiC-Releasenotes.pdf](#). Sie enthalten die Änderungen der aktuell unterstützten ERiC Releases.
- Die [Datenartversionmatrix.xml](#). Sie enthält eine Übersicht der `datenartVersion`en, die ERiC unterstützt. Einige API-Funktionen verwenden die `datenartVersion` als Parameter, weitere Informationen siehe [ERiC-Entwicklerhandbuch.pdf](#), Kap. "datenartVersion – Definition und Verwendung."
- Diese API-Referenz sowie die Dokumentation aller Feldkennungen, [Plausibilitätsprüfungen](#), Schemata und [Schnittstellenbeschreibungen](#).

Encoding und Zeichensatz

Alle Daten, die an die ELSTER Annahmeserver übermittelt werden, sind in UTF-8 zu kodieren. Hierbei dürfen die zu übermittelnden Daten keine BOM (=Byte Order Mark) enthalten.

Der Datentyp **char** zeigt an, wo UTF-8 kodierte Zeichenketten zu verwenden sind. Der Datentyp [byteChar](#) zeigt an, wo ASCII zu verwenden ist bzw. bei Pfadangaben das betriebssystemspezifische Encoding, siehe [ERiC-Entwicklerhandbuch.pdf](#), Kap. "Übergabe von Pfaden an ERiC API-Funktionen."

Die erlaubte Zeichenmenge lässt sich dem Datentyp *BaseStringSType* aus dem ElsterBasisSchema [headerbasis_datentypen.xsd](#) der Schnittstellenbeschreibung entnehmen.

Bei der Eingabe von PINs sind nur Zeichen aus dem ASCII Zeichensatz, ohne Sonder- und Steuerzeichen, erlaubt, siehe <https://de.wikipedia.org/wiki/ASCII>.

Datenstruktur-Verzeichnis

Datenstrukturen

Hier folgt die Aufzählung aller Datenstrukturen mit einer Kurzbeschreibung:

<u>eric_druck_parameter_t</u> (Diese Struktur enthält alle für den Druck notwendigen Informationen)	8
<u>eric_verschluesselungs_parameter_t</u> (Für die Signatur oder Authentifizierung benötigte Informationen)	12
<u>eric_zertifikat_parameter_t</u> (Struktur mit Informationen zur Erzeugung von Zertifikaten mit <u>EricCreateKey()</u>)	14
<u>OttoProxyKonfiguration</u> (Diese Struktur enthält alle Informationen, die Otto benötigt, um die Verbindung zum OTTER-Server oder dem ELSTER-eID-Server über einen Proxy aufzubauen)	18

Datei-Verzeichnis

Auflistung der Dateien

Hier folgt die Aufzählung aller Dateien mit einer Kurzbeschreibung:

<u>eric_fehlercodes.h</u> (Auflistung der ERIC API-Fehlercodes)	21
<u>ericapi.h</u> (Deklaration der ERiC API-Funktionen für die Singlethreading-API)	50
<u>ericmtapi.h</u> (Deklaration der ERiC API-Funktionen für die Multithreading-API)	127
<u>ericversion.h</u>	209
<u>erictoolkit.h</u> (Bereitstellung von Prüffunktionen ohne Abhängigkeit zu anderen ERiC Bibliotheken)	210
<u>otto.h</u> (Deklaration der Otto-Funktionen)	221
<u>otto_statuscode.h</u> (Auflistung der Otto-Statuscodes)	259
<u>otto_types.h</u> (Definition von Datenstrukturen und Datentypen)	270
<u>eric_types.h</u> (Definition von Datenstrukturen und Datentypen)	275
<u>ericapiExport.h</u>	284
<u>ericdef.h</u>	286
<u>platform.h</u>	288

Datenstruktur-Dokumentation

eric_druck_parameter_t Strukturreferenz

Diese Struktur enthält alle für den Druck notwendigen Informationen.

```
#include <eric_types.h>
```

Zusammengehörigkeiten von eric_druck_parameter_t:

eric_druck_parameter_t
+ version
+ vorschau
+ duplexDruck
+ pdfName
+ fussText
+ pdfCallback
+ pdfCallbackBenutzerdaten

Datenfelder

- [uint32_t version](#)
Version dieser Struktur. Die Version muss derzeit 4 sein. Bei Änderungen dieser Struktur wird dieser Wert inkrementiert.
- [uint32_t vorschau](#)
Soll ein Vorschau-PDF erstellt werden?
- [uint32_t duplexDruck](#)
Soll die PDF-Datei für einen doppelseitigen Ausdruck mit Heftrand zum Lochen vorbereitet werden?
- const [byteChar](#) * [pdfName](#)
Pfad der erzeugten PDF-Datei.
- const char * [fussText](#)

Fußtext der auf dem Ausdruck verwendet werden soll (optional).

- [EricPdfCallback pdfCallback](#)

Optionale Angabe einer Callback-Funktion für die Übergabe eines PDFs vom ERiC an die Anwendung. Wenn hier eine Callback-Funktion angegeben wird, schreibt der ERiC PDFs nicht in eine Datei, sondern ruft stattdessen die Callback-Funktion auf. Wenn hier NULL angegeben wird, schreibt der ERiC die PDFs in Dateien.

- void * [pdfCallbackBenutzerdaten](#)

Zeiger auf Benutzerdaten, der bei einem Aufruf des `pdfCallback` vom ERiC als Parameter unverändert wieder an die Anwendung mitgegeben wird.

Ausführliche Beschreibung

Diese Struktur enthält alle für den Druck notwendigen Informationen.

Der Anwendungsentwickler muss diese Struktur allokalieren und nach Verwendung wieder freigeben.

Dokumentation der Felder

[uint32_t eric_druck_parameter_t::duplexDruck](#)

Soll die PDF-Datei für einen doppelseitigen Ausdruck mit Heftrand zum Lochen vorbereitet werden?

Anwendungsfälle:

- duplexDruck = 1: Die geraden Seiten werden für einen Heftrand zum Lochen nach links eingerückt. Für Details siehe [ERIC-Entwicklerhandbuch.pdf](#).
- duplexDruck = 0: Es erfolgt keine Einrückung der geraden Seiten. Das erstellte PDF ist nur zum blattweisen Ausdruck der Seiten vorgesehen.

Zu beachten:

Bei Werten ungleich 0 oder 1 wird [ERIC_GLOBAL_UNGUELTIGER_PARAMETER](#) zurückgegeben und eine Fehlermeldung in die Logdatei geschrieben.

const char* eric_druck_parameter_t::fussText

Fußtext der auf dem Ausdruck verwendet werden soll (optional).

Wenn der übergebene Text länger als [ERIC_MAX_LAENGE_FUSSTEXT](#)

Zeichen ist, dann bricht der Druck mit Fehlercode

[ERIC_PRINT_FUSSTEXT_ZU_LANG](#) ab!

Zu beachten:

Fachliche Informationen sind im [ERiC-Entwicklerhandbuch.pdf](#) nachzulesen.

[EricPdfCallback](#) eric_druck_parameter_t::pdfCallback

Optionale Angabe einer Callback-Funktion für die Übergabe eines PDFs vom ERiC an die Anwendung. Wenn hier eine Callback-Funktion angegeben wird, schreibt der ERiC PDFs nicht in eine Datei, sondern ruft stattdessen die Callback-Funktion auf. Wenn hier NULL angegeben wird, schreibt der ERiC die PDFs in Dateien.

void* eric_druck_parameter_t::pdfCallbackBenutzerdaten

Zeiger auf Benutzerdaten, der bei einem Aufruf des `pdfCallback` vom ERiC als Parameter unverändert wieder an die Anwendung mitgegeben wird.

const [byteChar](#)* eric_druck_parameter_t::pdfName

Pfad der erzeugten PDF-Datei.

Pfade müssen auf Windows in der für Dateifunktionen benutzten ANSI-Codepage, auf Linux, AIX und Linux Power in der für das Dateisystem benutzten Locale und auf macOS in der "decomposed form" von UTF-8 übergeben werden. Weiterführende Informationen hierzu, sowie zu nicht erlaubten Zeichen in Pfaden und Pfadtrennzeichen, relative Pfadangabe, etc. siehe [ERiC-Entwicklerhandbuch.pdf](#), Kap. "Übergabe von Pfaden an ERiC API-Funktionen".

Windows-Beispiel: "c:\\test\\ericprint.pdf"

Soll eine PDF-Datei angelegt werden, ist der `pdfName` zwingend erforderlich.

Besonderheiten bei Sammeldaten:

Für Sammeldaten wird dem PDF-Dateinamen vor der Dateiendung das Nutzdatenticket angefügt:

<PDF-Dateiname>_<Nutzdatenticket>.pdf

Optional kann der PDF-Dateiname den Platzhalter "%t" enthalten, der dann durch das Nutzdatenticket ersetzt wird:

"%t_ericprint.pdf" --> "<Nutzdatenticket>_ericprint.pdf"

Zu beachten:

Es ist sicherzustellen, dass alle PDF-Dateien im Dateisystem erstellt bzw. geschrieben werden können. Falls es beim Erstellen der PDF-Dokumente einen Fehler gibt oder falls diese nicht geschrieben werden können, wird die Bearbeitung abgebrochen, eine Log-Ausgabe erstellt, aus der hervorgeht, welcher Steuerfall nicht gedruckt werden konnte, und eine Fehlermeldung an den Aufrufer zurückgeliefert.

uint32_t eric_druck_parameter_t::version

Version dieser Struktur. Die Version muss derzeit 4 sein. Bei Änderungen dieser Struktur wird dieser Wert inkrementiert.

Zu beachten:

Bei einem Wert ungleich 4 wird [ERIC_GLOBAL_UNGUELTIGE_PARAMETER_VERSION](#) zurückgegeben und eine Fehlermeldung in die Logdatei geschrieben.

uint32_t eric_druck_parameter_t::vorschau

Soll ein Vorschau-PDF erstellt werden?

Anwendungsfälle:

- vorschau = 1: Ein Vorschau-PDF wird erzeugt und als solches gekennzeichnet.
- vorschau = 0: Es wird kein Vorschau-PDF erzeugt.

Zu beachten:

Bei Werten ungleich 0 oder 1 wird [ERIC_GLOBAL_UNGUELTIGER_PARAMETER](#) zurückgegeben und eine Fehlermeldung in die Logdatei geschrieben.

Die Dokumentation für diese Struktur wurde erzeugt aufgrund der Datei:

[eric_types.h](#)

eric_verschluesselungs_parameter_t Strukturreferenz

Für die Signatur oder Authentifizierung benötigte Informationen.

```
#include <eric_types.h>
```

Zusammengehörigkeiten von eric_verschluesselungs_parameter_t:

eric_verschluesselungs _parameter_t
+ version
+ zertifikatHandle
+ pin

Datenfelder

- [uint32_t version](#)

Version dieser Struktur. Muss derzeit immer 3 sein. Bei Änderungen dieser Struktur wird dieser Wert inkrementiert.

- [EricZertifikatHandle zertifikatHandle](#)

Verweis auf den KeyStore, siehe [EricGetHandleToCertificate\(\)](#).

- const [byteChar](#) * [pin](#)

PIN für den KeyStore.

Ausführliche Beschreibung

Für die Signatur oder Authentifizierung benötigte Informationen.

Diese Struktur ist vom Anwender zu allokalieren und samt Inhalt auch wieder freizugeben.

Dokumentation der Felder

const [byteChar](#)* eric_verschluesselungs_parameter_t::pin

PIN für den KeyStore.

[uint32_t](#) eric_verschluesselungs_parameter_t::version

Version dieser Struktur. Muss derzeit immer 3 sein. Bei Änderungen dieser Struktur wird dieser Wert inkrementiert.

Zu beachten:

Bei einem Wert ungleich 3 wird

[ERIC_GLOBAL_UNGUELTIGE_PARAMETER_VERSION](#) zurückgegeben und eine Fehlermeldung in die Logdatei geschrieben.

[EricZertifikatHandle](#) eric_verschluesselungs_parameter_t::zertifikatHandle

Verweis auf den KeyStore, siehe [EricGetHandleToCertificate\(\)](#).

Die Dokumentation für diese Struktur wurde erzeugt aufgrund der Datei:
[eric_types.h](#)

eric_zertifikat_parameter_t Strukturreferenz

Struktur mit Informationen zur Erzeugung von Zertifikaten mit [EricCreateKey\(\)](#).

```
#include <eric_types.h>
```

Zusammengehörigkeiten von eric_zertifikat_parameter_t:

eric_zertifikat_parameter_t	
+	version
+	name
+	land
+	ort
+	adresse
+	email
+	organisation
+	abteilung
+	beschreibung

Datenfelder

- [uint32_t version](#)

Version dieser Struktur. Muss derzeit immer 1 sein. Bei Änderungen dieser Struktur wird dieser Wert inkrementiert.

- const char * [name](#)

Name des Anwenders.

- const char * [land](#)

*Land (Länderkürzel) des Anwenders. **Beispiel:** "DE".*

- const char * [ort](#)

*Wohnort des Anwenders, inklusive PLZ. **Beispiel:** "D-10179 Berlin".*

- const char * [adresse](#)

*Straßenangabe mit Hausnummer des Anwenders mit Zusätzen, **Beispiel:** "Musterstraße 123 Zugang im Rückgebäude".*

- `const char * email`
E-Mail-Adresse des Anwenders.
 - `const char * organisation`
Name der Organisation.
 - `const char * abteilung`
Name der Abteilung (organizational unit) der Organisation.
 - `const char * beschreibung`
Beschreibung, welche für den Anwender im Zertifikat abgelegt wird.
-

Ausführliche Beschreibung

Struktur mit Informationen zur Erzeugung von Zertifikaten mit [EricCreateKey\(\)](#).

Die Elemente der Struktur beschreiben den Anwender, für den ein Schlüssel erstellt werden soll. Unbenutzte Parameter müssen mit NULL oder Leerstring initialisiert werden.

Diese Struktur und ihre Elemente sind vom Anwender zu allokalieren und samt Inhalt auch wieder freizugeben. Alle Elemente sind vom Anwender zu initialisieren.

Dokumentation der Felder

`const char* eric_zertifikat_parameter_t::abteilung`

Name der Abteilung (organizational unit) der Organisation.

Die Angabe dieses Wertes ist optional. Wenn `organisation` und `abteilung` nicht angegeben werden, wird "ERiC" verwendet.

const char* eric_zertifikat_parameter_t::adresse

Straßenangabe mit Hausnummer des Anwenders mit Zusätzen, **Beispiel:**
"Musterstraße 123 Zugang im Rückgebäude".

Die Angabe dieses Wertes ist optional.

const char* eric_zertifikat_parameter_t::beschreibung

Beschreibung, welche für den Anwender im Zertifikat abgelegt wird.

Die Angabe dieses Wertes ist optional.

const char* eric_zertifikat_parameter_t::email

E-Mail-Adresse des Anwenders.

Die Angabe dieses Wertes ist optional.

const char* eric_zertifikat_parameter_t::land

Land (Länderkürzel) des Anwenders. **Beispiel:** "DE".

Die Angabe dieses Wertes ist optional.

const char* eric_zertifikat_parameter_t::name

Name des Anwenders.

Die Angabe des Namens ist obligatorisch. Der Parameter darf nicht mit NULL oder einem Leerstring belegt werden.

const char* eric_zertifikat_parameter_t::organisation

Name der Organisation.

Die Angabe dieses Wertes ist optional. Wenn `organisation` und `abteilung` nicht angegeben werden, wird "ELSTER" verwendet.

const char* eric_zertifikat_parameter_t::ort

Wohnort des Anwenders, inklusive PLZ. **Beispiel:** "D-10179 Berlin".

Die Angabe dieses Wertes ist optional.

uint32 t eric_zertifikat_parameter_t::version

Version dieser Struktur. Muss derzeit immer 1 sein. Bei Änderungen dieser Struktur wird dieser Wert inkrementiert.

Zu beachten:

Bei einem Wert ungleich 1 wird

[ERIC_GLOBAL_UNGUELTIGE_PARAMETER_VERSION](#) zurückgegeben und eine Fehlermeldung in die Logdatei geschrieben.

Die Dokumentation für diese Struktur wurde erzeugt aufgrund der Datei:

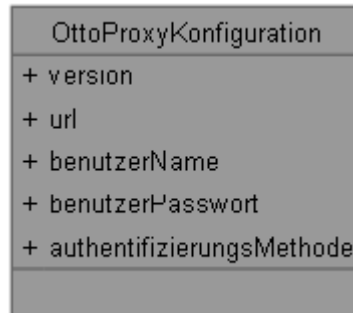
[eric_types.h](#)

OttoProxyKonfiguration Strukturreferenz

Diese Struktur enthält alle Informationen, die Otto benötigt, um die Verbindung zum OTTER-Server oder dem ELSTER-eID-Server über einen Proxy aufzubauen.

```
#include <otto_types.h>
```

Zusammengehörigkeiten von OttoProxyKonfiguration:



Datenfelder

- int [version](#)
Die Version der Struktur.
- const [byteChar](#) * [url](#)
Die URL des Proxies einschließlich Port.
- const [byteChar](#) * [benutzerName](#)
Der Benutzername für eine Proxy-Authentifizierung.
- const [byteChar](#) * [benutzerPasswort](#)
Das Passwort für eine Proxy-Authentifizierung.
- const char * [authentifizierungsmethode](#)
Die Authentifizierungsmethode, mit der der Proxy arbeitet.

Ausführliche Beschreibung

Diese Struktur enthält alle Informationen, die Otto benötigt, um die Verbindung zum OTTER-Server oder dem ELSTER-eID-Server über einen Proxy aufzubauen.

Dokumentation der Felder

const char* OttoProxyKonfiguration::authentifizierungsmethode

Die Authentifizierungsmethode, mit der der Proxy arbeitet.

Folgende Methoden werden unterstützt:

- "Any"
- "Basic"
- "Digest"
- "DigestIE"
- "NTLM"
- "SPNEGO"

Mehrere Werte sind durch Kommas getrennt anzugeben. Die Groß-, Kleinschreibung der Werte wird ignoriert. Dieses Element darf NULL sein, wenn der Proxy keine Authentifizierung erfordert.

const [byteChar](#)* OttoProxyKonfiguration::benutzerName

Der Benutzername für eine Proxy-Authentifizierung.

Dieses Element darf NULL sein.

const [byteChar](#)* OttoProxyKonfiguration::benutzerPasswort

Das Passwort für eine Proxy-Authentifizierung.

Dieses Element darf NULL sein.

const [byteChar](#)* OttoProxyKonfiguration::url

Die URL des Proxies einschließlich Port.

IPv6-Adressen sind in eckigen Klammern anzugeben. Otto unterstützt folgende Protokolle:

- http
- socks4

- socks5

Beispiele:

- mein.pro.xy:1234
- <http://203.0.113.0:1234>
- socks4://mein.pro.xy:1234
- socks5://[2001:0DB8:AC10:FE01::]:1234

Dieses Element darf nicht NULL sein.

int OttoProxyKonfiguration::version

Die Version der Struktur.

Hier ist aktuell fest der Wert 1 zu setzen.

Die Dokumentation für diese Struktur wurde erzeugt aufgrund der Datei:

[otto_types.h](#)

Datei-Dokumentation

eric_fehlercodes.h-Dateireferenz

Auflistung der ERIC API-Fehlercodes.

Typdefinitionen

- typedef enum [eric_fehlercode](#) [eric_fehlercode_t](#)

Aufzählungen

- enum [eric_fehlercode](#) { [ERIC_OK](#) = 0, [ERIC_GLOBAL_UNKNOWN](#) = 610001001, [ERIC_GLOBAL_PRUEF_FEHLER](#) = 610001002, [ERIC_GLOBAL_HINWEISE](#) = 610001003, [ERIC_GLOBAL_FEHLERMELDUNG_NICHT_VORHANDEN](#) = 610001007, [ERIC_GLOBAL_KEINE_DATEN_VORHANDEN](#) = 610001008, [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#) = 610001013, [ERIC_GLOBAL_DATEI_NICHT_GEFUNDEN](#) = 610001014, [ERIC_GLOBAL_HERSTELLER_ID_NICHT_ERLAUBT](#) = 610001016, [ERIC_GLOBAL_ILLEGAL_STATE](#) = 610001017, [ERIC_GLOBAL_FUNKTION_NICHT_ERLAUBT](#) = 610001018, [ERIC_GLOBAL_ECHTFALL_NICHT_ERLAUBT](#) = 610001019, [ERIC_GLOBAL_NO_VERSAND_IN_BETA_VERSION](#) = 610001020, [ERIC_GLOBAL_TESTMERKER_UNGUELTIG](#) = 610001025, [ERIC_GLOBAL_DATENSATZ_ZU_GROSS](#) = 610001026, [ERIC_GLOBAL_VERSCHLUESSELUNGS_PARAMETER_NICHT_ERLAUBT](#) = 610001027, [ERIC_GLOBAL_NUR_PORTALZERTIFIKAT_ERLAUBT](#) = 610001028, [ERIC_GLOBAL_ERROR_XML_CREATE](#) = 610001030, [ERIC_GLOBAL_TEXTPUFFERGROESSE_FIX](#) = 610001031, [ERIC_GLOBAL_INTERNER_FEHLER](#) = 610001032, [ERIC_GLOBAL_ARITHMETIKFEHLER](#) = 610001033, [ERIC_GLOBAL_STEUERNUMMER_UNGUELTIG](#) = 610001034, [ERIC_GLOBAL_STEUERNUMMER_FALSCHE_LAENGE](#) = 610001035, [ERIC_GLOBAL_STEUERNUMMER_NICHT_NUMERISCH](#) = 610001036, [ERIC_GLOBAL_LANDESNUMMER_UNBEKANNT](#) = 610001037, [ERIC_GLOBAL_BUFANR_UNBEKANNT](#) = 610001038, [ERIC_GLOBAL_LANDESNUMMER_BUFANR](#) = 610001039, [ERIC_GLOBAL_PUFFER_ZUGRIFFSKONFLIKT](#) = 610001040, [ERIC_GLOBAL_PUFFER_UEBERLAUF](#) = 610001041,

[ERIC GLOBAL DATENARTVERSION UNBEKANNT](#) = 610001042,
[ERIC GLOBAL DATENARTVERSION XML INKONSISTENT](#) = 610001044,
[ERIC GLOBAL COMMONDATA NICHT VERFUEGBAR](#) = 610001045,
[ERIC GLOBAL LOG EXCEPTION](#) = 610001046,
[ERIC GLOBAL TRANSPORTSCHLUESSEL NICHT ERLAUBT](#) = 610001047,
[ERIC GLOBAL OEFFENTLICHER SCHLUESSEL UNGUELTIG](#) = 610001048,
[ERIC GLOBAL TRANSPORTSCHLUESSEL TYP FALSCH](#) = 610001049,
[ERIC GLOBAL PUFFER UNGLEICHER INSTANZ](#) = 610001050,
[ERIC GLOBAL VORSATZ UNGUELTIG](#) = 610001051,
[ERIC GLOBAL DATEIZUGRIFF VERWEIGERT](#) = 610001053,
[ERIC GLOBAL UNGUELTIGE INSTANZ](#) = 610001080,
[ERIC GLOBAL NICHT INITIALISIERT](#) = 610001081,
[ERIC GLOBAL MEHRFACHE INITIALISIERUNG](#) = 610001082,
[ERIC GLOBAL FEHLER INITIALISIERUNG](#) = 610001083,
[ERIC GLOBAL UNKNOWN PARAMETER ERROR](#) = 610001102,
[ERIC GLOBAL CHECK CORRUPTED NDS](#) = 610001108,
[ERIC GLOBAL VERSCHLUESSELUNGS PARAMETER NICHT ANGEGEBE](#)
[N](#) = 610001206, [ERIC GLOBAL SEND FLAG MEHR ALS EINES](#) =
610001209, [ERIC GLOBAL UNGUELTIGE FLAG KOMBINATION](#) =
610001218, [ERIC GLOBAL UNGUELTIGER PARAMETER](#) = 610001222,
[ERIC GLOBAL DRUCK FUER VERFAHREN NICHT ERLAUBT](#) =
610001224, [ERIC GLOBAL VERSAND ART NICHT UNTERSTUETZT](#) =
610001225, [ERIC GLOBAL UNGUELTIGE PARAMETER VERSION](#) =
610001226, [ERIC GLOBAL TRANSFERHANDLE](#) = 610001227,
[ERIC GLOBAL PLUGININITIALISIERUNG](#) = 610001228,
[ERIC GLOBAL INKOMPATIBLE VERSIONEN](#) = 610001229,
[ERIC GLOBAL VERSCHLUESSELUNGSVERFAHREN NICHT UNTERSTUE](#)
[TZT](#) = 610001230,
[ERIC GLOBAL MEHRFACHAUFRUFE NICHT UNTERSTUETZT](#) =
610001231, [ERIC GLOBAL UTI COUNTRY NOT SUPPORTED](#) = 610001404,
[ERIC GLOBAL IBAN FORMALER FEHLER](#) = 610001501,
[ERIC GLOBAL IBAN LAENDERCODE FEHLER](#) = 610001502,
[ERIC GLOBAL IBAN LANDESFORMAT FEHLER](#) = 610001503,
[ERIC GLOBAL IBAN PRUEFZIFFER FEHLER](#) = 610001504,
[ERIC GLOBAL BIC FORMALER FEHLER](#) = 610001510,
[ERIC GLOBAL BIC LAENDERCODE FEHLER](#) = 610001511,
[ERIC GLOBAL ZULASSUNGSNUMMER ZU LANG](#) = 610001519,
[ERIC GLOBAL IDNUMMER UNGUELTIG](#) = 610001525,
[ERIC GLOBAL NULL PARAMETER](#) = 610001526,
[ERIC GLOBAL EWAZ UNGUELTIG](#) = 610001527,
[ERIC GLOBAL EWAZ LANDESKUERZEL UNBEKANNT](#) = 610001528,
[ERIC GLOBAL UPDATE NECESSARY](#) = 610001851,
[ERIC GLOBAL EINSTELLUNG NAME UNGUELTIG](#) = 610001860,
[ERIC GLOBAL EINSTELLUNG WERT UNGUELTIG](#) = 610001861,

[ERIC GLOBAL ERR DEKODIEREN](#) = 610001862,
[ERIC GLOBAL FUNKTION NICHT UNTERSTUETZT](#) = 610001863,
[ERIC GLOBAL NUTZDATENTICKETS NICHT EINDEUTIG](#) = 610001865,
[ERIC GLOBAL NUTZDATENHEADERVERSIONEN UNEINHEITLICH](#) =
610001866, [ERIC GLOBAL BUNDESLAENDER UNEINHEITLICH](#) =
610001867, [ERIC GLOBAL ZEITRAEUME UNEINHEITLICH](#) = 610001868,
[ERIC GLOBAL NUTZDATENHEADER EMPFAENGER NICHT KORREKT](#) =
610001869, [ERIC TRANSFER COM ERROR](#) = 610101200,
[ERIC TRANSFER VORGANG NICHT UNTERSTUETZT](#) = 610101201,
[ERIC TRANSFER ERR XML THEADER](#) = 610101210,
[ERIC TRANSFER ERR PARAM](#) = 610101251,
[ERIC TRANSFER ERR DATENTEILENDNOTFOUND](#) = 610101253,
[ERIC TRANSFER ERR BEGINDATENLIEFERANT](#) = 610101255,
[ERIC TRANSFER ERR ENDDATENLIEFERANT](#) = 610101256,
[ERIC TRANSFER ERR BEGINTRANSPORTSCHLUESSEL](#) = 610101257,
[ERIC TRANSFER ERR ENDTRANSPORTSCHLUESSEL](#) = 610101258,
[ERIC TRANSFER ERR BEGINDATENGROESSE](#) = 610101259,
[ERIC TRANSFER ERR ENDDATENGROESSE](#) = 610101260,
[ERIC TRANSFER ERR SEND](#) = 610101271,
[ERIC TRANSFER ERR NOTENCRYPTED](#) = 610101274,
[ERIC TRANSFER ERR PROXYCONNECT](#) = 610101276,
[ERIC TRANSFER ERR CONNECTSERVER](#) = 610101278,
[ERIC TRANSFER ERR NORESPONSE](#) = 610101279,
[ERIC TRANSFER ERR PROXYAUTH](#) = 610101280,
[ERIC TRANSFER ERR SEND INIT](#) = 610101282,
[ERIC TRANSFER ERR TIMEOUT](#) = 610101283,
[ERIC TRANSFER ERR PROXYPORT INVALID](#) = 610101284,
[ERIC TRANSFER ERR OTHER](#) = 610101291,
[ERIC TRANSFER ERR XML NHEADER](#) = 610101292,
[ERIC TRANSFER ERR XML ENCODING](#) = 610101293,
[ERIC TRANSFER ERR ENDSIGUSER](#) = 610101294,
[ERIC TRANSFER ERR XMLTAG NICHT GEFUNDEN](#) = 610101295,
[ERIC TRANSFER ERR DATENTEILFEHLER](#) = 610101297,
[ERIC TRANSFER EID ZERTIFIKATFEHLER](#) = 610101500,
[ERIC TRANSFER EID KEINKONTO](#) = 610101510,
[ERIC TRANSFER EID IDNRNICHTEINDEUTIG](#) = 610101511,
[ERIC TRANSFER EID SERVERFEHLER](#) = 610101512,
[ERIC TRANSFER EID KEINCLIENT](#) = 610101520,
[ERIC TRANSFER EID CLIENTFEHLER](#) = 610101521,
[ERIC TRANSFER EID FEHLENDEFELDER](#) = 610101522,
[ERIC TRANSFER EID IDENTIFIKATIONABGEBROCHEN](#) = 610101523,
[ERIC TRANSFER EID NPABLOCKIERT](#) = 610101524,
[ERIC CRYPT ERROR CREATE KEY](#) = 610201016,
[ERIC CRYPT E INVALID HANDLE](#) = 610201101,

[ERIC CRYPT E MAX SESSION](#) = 610201102, [ERIC CRYPT E BUSY](#) = 610201103, [ERIC CRYPT E OUT OF MEM](#) = 610201104, [ERIC CRYPT E PSE PATH](#) = 610201105, [ERIC CRYPT E PIN WRONG](#) = 610201106, [ERIC CRYPT E PIN LOCKED](#) = 610201107, [ERIC CRYPT E P7 READ](#) = 610201108, [ERIC CRYPT E P7 DECODE](#) = 610201109, [ERIC CRYPT E P7 RECIPIENT](#) = 610201110, [ERIC CRYPT E P12 READ](#) = 610201111, [ERIC CRYPT E P12 DECODE](#) = 610201112, [ERIC CRYPT E P12 SIG KEY](#) = 610201113, [ERIC CRYPT E P12 ENC KEY](#) = 610201114, [ERIC CRYPT E P11 SIG KEY](#) = 610201115, [ERIC CRYPT E P11 ENC KEY](#) = 610201116, [ERIC CRYPT E XML PARSE](#) = 610201117, [ERIC CRYPT E XML SIG ADD](#) = 610201118, [ERIC CRYPT E XML SIG TAG](#) = 610201119, [ERIC CRYPT E XML SIG SIGN](#) = 610201120, [ERIC CRYPT E ENCODE UNKNOWN](#) = 610201121, [ERIC CRYPT E ENCODE ERROR](#) = 610201122, [ERIC CRYPT E XML INIT](#) = 610201123, [ERIC CRYPT E ENCRYPT](#) = 610201124, [ERIC CRYPT E DECRYPT](#) = 610201125, [ERIC CRYPT E P11 SLOT EMPTY](#) = 610201126, [ERIC CRYPT E NO SIG ENC KEY](#) = 610201127, [ERIC CRYPT E LOAD DLL](#) = 610201128, [ERIC CRYPT E NO SERVICE](#) = 610201129, [ERIC CRYPT E ESICL EXCEPTION](#) = 610201130, [ERIC CRYPT E ESIGNER NICHT GELADEN](#) = 610201140, [ERIC CRYPT E INKOMPATIBLE ESIGNER VERSION](#) = 610201141, [ERIC CRYPT E VERALTETE ESIGNER VERSION](#) = 610201142, [ERIC CRYPT E TOKEN TYPE MISMATCH](#) = 610201144, [ERIC CRYPT E P12 CREATE](#) = 610201146, [ERIC CRYPT E VERIFY CERT CHAIN](#) = 610201147, [ERIC CRYPT E P11 ENGINE LOADED](#) = 610201148, [ERIC CRYPT E USER CANCEL](#) = 610201149, [ERIC CRYPT ZERTIFIKAT](#) = 610201200, [ERIC CRYPT SIGNATUR](#) = 610201201, [ERIC CRYPT NICHT UNTERSTUETZTES PSE FORMAT](#) = 610201203, [ERIC CRYPT PIN BENOETIGT](#) = 610201205, [ERIC CRYPT PIN STAERKE NICHT AUSREICHEND](#) = 610201206, [ERIC CRYPT E INTERN](#) = 610201208, [ERIC CRYPT ZERTIFIKATSPFAD KEIN VERZEICHNIS](#) = 610201209, [ERIC CRYPT ZERTIFIKATSDATEI EXISTIERT BEREITS](#) = 610201210, [ERIC CRYPT PIN ENTHAELT UNGUELTIGE ZEICHEN](#) = 610201211, [ERIC CRYPT CORRUPTED](#) = 610201213, [ERIC CRYPT EIDKARTE NICHT UNTERSTUETZT](#) = 610201214, [ERIC CRYPT E SC SLOT EMPTY](#) = 610201215, [ERIC CRYPT E SC NO APPLETT](#) = 610201216, [ERIC CRYPT E SC SESSION](#) = 610201217, [ERIC CRYPT E P11 NO SIG CERT](#) = 610201218,

[ERIC CRYPT E P11 INIT FAILED](#) = 610201219,
[ERIC CRYPT E P11 NO ENC CERT](#) = 610201220,
[ERIC CRYPT E P12 NO SIG CERT](#) = 610201221,
[ERIC CRYPT E P12 NO ENC CERT](#) = 610201222,
[ERIC CRYPT E SC ENC KEY](#) = 610201223,
[ERIC CRYPT E SC NO SIG CERT](#) = 610201224,
[ERIC CRYPT E SC NO ENC CERT](#) = 610201225,
[ERIC CRYPT E SC INIT FAILED](#) = 610201226,
[ERIC CRYPT E SC SIG KEY](#) = 610201227,
[ERIC CRYPT E DATA NOT INITIALIZED](#) = 610201228,
[ERIC CRYPT E ASN1 READ BUFFER TOO SMALL](#) = 610201229,
[ERIC CRYPT E ASN1 READ DATA INCOMPLETE](#) = 610201230,
[ERIC CRYPT E ASN1 NO ENVELOPED DATA](#) = 610201231,
[ERIC CRYPT E ASN1 NO CONTENT DATA](#) = 610201232,
[ERIC IO FEHLER](#) = 610301001, [ERIC IO DATEI INKORREKT](#) = 610301005,
[ERIC IO PARSE FEHLER](#) = 610301006,
[ERIC IO NDS GENERIERUNG FEHLGESCHLAGEN](#) = 610301007,
[ERIC IO MASTERDATENSERVICE NICHT VERFUEGBAR](#) = 610301010,
[ERIC IO STEUERZEICHEN IM NDS](#) = 610301014,
[ERIC IO VERSIONSINFORMATIONEN NICHT GEFUNDEN](#) = 610301031,
[ERIC IO FALSCHES VERFAHREN](#) = 610301104,
[ERIC IO READER MEHRFACHE STEUERFAELLE](#) = 610301105,
[ERIC IO READER UNERWARTETE ELEMENTE](#) = 610301106,
[ERIC IO READER FORMALE FEHLER](#) = 610301107,
[ERIC IO READER FALSCHES ENCODING](#) = 610301108,
[ERIC IO READER MEHRFACHE NUTZDATEN ELEMENTE](#) = 610301109,
[ERIC IO READER MEHRFACHE NUTZDATENBLOCK ELEMENTE](#) =
610301110, [ERIC IO UNBEKANNTE DATENART](#) = 610301111,
[ERIC IO READER UNTERSACHBEREICH UNGUELTIG](#) = 610301114,
[ERIC IO READER ZU VIELE NUTZDATENBLOCK ELEMENTE](#) =
610301115, [ERIC IO READER STEUERZEICHEN IM TRANSFERHEADER](#) =
610301150, [ERIC IO READER STEUERZEICHEN IM NUTZDATENHEADER](#)
= 610301151, [ERIC IO READER STEUERZEICHEN IN DEN NUTZDATEN](#) =
610301152, [ERIC IO READER RABE FEHLER](#) = 610301170,
[ERIC IO READER KEINE RABEID](#) = 610301171,
[ERIC IO READER RABEID UNGUELTIG](#) = 610301172,
[ERIC IO READER RABE VERIFIKATIONSID UNGUELTIG](#) = 610301173,
[ERIC IO READER RABE REFERENZID UNGUELTIG](#) = 610301174,
[ERIC IO READER RABE REFERENZID NICHT ERLAUBT](#) = 610301175,
[ERIC IO READER RABE REFERENZIDS NICHT EINDEUTIG](#) = 610301176,
[ERIC IO READER ZU VIELE ANHAENGE](#) = 610301190,
[ERIC IO READER ANHANG ZU GROSS](#) = 610301191,
[ERIC IO READER ANHAENGE ZU GROSS](#) = 610301192,
[ERIC IO READER ANHANG ZU KLEIN](#) = 610301193,

```

ERIC IO READER SCHEMA VALIDIERUNGSFEHLER = 610301200,
ERIC IO READER UNBEKANNTE XML ENTITY = 610301201,
ERIC IO TESTHERSTELLERID GESPERRT = 610301202,
ERIC IO DATENTEILNOTFOUND = 610301252,
ERIC IO DATENTEILENDNOTFOUND = 610301253,
ERIC IO UEBERGABEPARAMETER FEHLERHAFT = 610301300,
ERIC IO UNGUELTIGE UTF8 SEQUENZ = 610301400,
ERIC IO UNGUELTIGE ZEICHEN IN PARAMETER = 610301401,
ERIC PRINT INTERNER FEHLER = 610501001,
ERIC PRINT DRUCKVORLAGE NICHT GEFUNDEN = 610501002,
ERIC PRINT UNGUELTIGER DATEI PFAD = 610501004,
ERIC PRINT INITIALISIERUNG FEHLERHAFT = 610501007,
ERIC PRINT AUSGABEZIEL UNBEKANNT = 610501008,
ERIC PRINT ABRUCH DRUCKVORBEREITUNG = 610501009,
ERIC PRINT ABRUCH GENERIERUNG = 610501010,
ERIC PRINT STEUERFALL NICHT UNTERSTUETZT = 610501011,
ERIC PRINT FUSSTEXT ZU LANG = 610501012,
ERIC PRINT PDFCALLBACK = 610501015 }

```

Ausführliche Beschreibung

Auflistung der ERIC API-Fehlercodes.

Dokumentation der benutzerdefinierten Typen

```
typedef enum eric fehlercode eric fehlercode t
```

Dokumentation der Aufzählungstypen

```
enum eric fehlercode
```

Aufzählungswerte:

ERIC_OK	[0] Verarbeitung fehlerfrei.
---------	------------------------------

ERIC_GLOBAL_UNKNOWN	[610001001] Verarbeitung fehlerhaft, keine genaueren Informationen vorhanden. Details stehen ggf. im Logfile (eric.log).
ERIC_GLOBAL_PRUEF_FEHLER	[610001002] Fehler während der Plausibilitätsprüfung, Datensatz nicht plausibel. Zur Ermittlung der fehlgeschlagenen Plausibilitätsprüfungen muss der Rückgabepuffer (Parameter "rueckgabeXmlPuffer") ausgewertet werden.
ERIC_GLOBAL_HINWEISE	[610001003] Hinweise während der Plausibilitätsprüfung, Datensatz ist aber plausibel. Zur Ermittlung der anzuzeigenden Hinweise muss der Rückgabepuffer (Parameter "rueckgabeXmlPuffer") ausgewertet werden.
ERIC_GLOBAL_FEHLERMELDUNG_NICHT_VORHANDEN	[610001007] Keine Klartextfehlermeldung vorhanden.
ERIC_GLOBAL_KEINE_DATEN_VORHANDEN	[610001008] Für den übergebenen Wert sind keine Daten vorhanden.
ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER	[610001013] Es ist nicht genügend Arbeitsspeicher vorhanden.
ERIC_GLOBAL_DATEI_NICHT_GEFUNDEN	[610001014] Datei nicht gefunden.
ERIC_GLOBAL_HERSTELLER_ID_NICHT_ERLAUBT	[610001016] Für dieses Verfahren/diese Datenart ist eine Bearbeitung mit der angegebenen Hersteller-ID nicht erlaubt.
ERIC_GLOBAL_ILLEGAL_STATE	[610001017] Ungültiger Zustand.

ERIC_GLOBAL _FUNKTION_N ICHT_ERLAUB T	[610001018] Die aufgerufene Funktion ist nicht erlaubt.
ERIC_GLOBAL _ECHTFALL_N ICHT_ERLAUB T	[610001019] Für dieses Verfahren/diese Datenart/diese Test-Hersteller-ID/diese ERiC-Einstellungen sind Echtfälle nicht erlaubt.
ERIC_GLOBAL _NO_VERSAN D_IN_BETA_V ERSION	[610001020] Der Versand von Echtfällen (= Fällen ohne gesetzten Testmerker) ist mit einer BETA-Version nicht möglich.
ERIC_GLOBAL _TESTMERKE R_UNGUELTIG	[610001025] Der übergebene Testmerker ist für das angegebene Verfahren nicht zulässig.
ERIC_GLOBAL _DATENSATZ _ZU_GROSS	[610001026] Der zu versendende Datensatz ist zu groß.
ERIC_GLOBAL _VERSCHLUE SSELUNGS_P ARAMETER_N ICHT_ERLAUB T	[610001027] Der Verschlüsselungsparameter darf nur bei authentifiziertem Versand angegeben werden.
ERIC_GLOBAL _NUR_PORTA LZERTIFIKAT_ ERLAUBT	[610001028] Bei der angegebenen Versandart sind nur Portal-Zertifikate erlaubt.
ERIC_GLOBAL _ERROR_XML _CREATE	[610001030] Es ist ein Fehler bei der Umwandlung nach XML aufgetreten.
ERIC_GLOBAL _TEXTPUFFE RGROESSE_F IX	[610001031] Die Größe des Textpuffers kann nicht verändert werden.
ERIC_GLOBAL _INTERNER_F EHLER	[610001032] Interner Fehler aufgetreten. Details stehen ggf. im Logfile (eric.log).

ERIC_GLOBAL _ARITHMETIK FEHLER	[610001033] Bei einer arithmetischen Operation ist ein Fehler aufgetreten. Details stehen im Logfile (eric.log).
ERIC_GLOBAL _STEUERNUM MER_UNGUEL TIG	[610001034] Ungültige Steuernummer.
ERIC_GLOBAL _STEUERNUM MER_FALSCH E_LAENGE	[610001035] Ungültige Steuernummer: Es werden 13 Stellen erwartet.
ERIC_GLOBAL _STEUERNUM MER_NICHT_ NUMERISCH	[610001036] Ungültige Steuernummer: Es werden nur Ziffern erwartet.
ERIC_GLOBAL _LANDESNUM MER_UNBEKA NNT	[610001037] Ungültige Landesnummer.
ERIC_GLOBAL _BUFANR_UN BEKANNT	[610001038] Ungültige Bundesfinanzamtsnummer.
ERIC_GLOBAL _LANDESNUM MER_BUFANR	[610001039] Ungültige Bundesfinanzamtsnummer.
ERIC_GLOBAL _PUFFER_ZU GRIFFSKONFL IKT	[610001040] Ein Puffer-Handle wurde mehrfach übergeben.
ERIC_GLOBAL _PUFFER_UE BERLAUF	[610001041] Es wurde versucht, einen Puffer über die maximal mögliche Länge hinaus zu beschreiben.
ERIC_GLOBAL _DATENARTV ERSION_UNB EKANNT	[610001042] Die übergebene Datenartversion ist unbekannt oder das benötigte ERiC-Plugin wurde nicht gefunden. Beachten Sie bitte, dass die Datenartversion case-sensitive ist.

ERIC_GLOBAL _DATENARTV ERSION_XML_ INKONSISTEN T	[610001044] Die übergebene Datenartversion passt nicht zum Eingangs-XML. Details stehen ggf. im Logfile (eric.log).
ERIC_GLOBAL _COMMONDA TA_NICHT_VE RFUEGBAR	[610001045] Das Plugin 'commonData' konnte nicht geladen werden oder bietet einen benötigten Service nicht an. Details stehen im Logfile (eric.log).
ERIC_GLOBAL _LOG_EXCEP TION	[610001046] Beim Schreiben in die Protokolldatei ist eine Ausnahme aufgetreten.
ERIC_GLOBAL _TRANSPORT SCHLUESSEL _NICHT_ERLA UBT	[610001047] Für diese Datenart darf im TransferHeader kein TransportSchlüssel angegeben werden.
ERIC_GLOBAL _OEFFENTLIC HER_SCHLUE SSEL_UNGUE LTIG	[610001048] Der übergebene öffentliche Schlüssel kann nicht eingelesen werden.
ERIC_GLOBAL _TRANSPORT SCHLUESSEL _TYP_FALSCH	[610001049] Der Typ des im TransferHeader angegebenen Transportschlüssels ist für diese Datenart nicht erlaubt.
ERIC_GLOBAL _PUFFER_UN GLEICHER_IN STANZ	[610001050] Das übergebene Puffer-Handle wurde nicht mit der vorliegenden Instanz erzeugt.
ERIC_GLOBAL _VORSATZ_U NGUELTIG	[610001051] Das Element "Vorsatz" enthält ungültige Werte, Details stehen im Logfile (eric.log).
ERIC_GLOBAL _DATEIZUGRI FF_VERWEIG ERT	[610001053] Auf eine Datei konnte nicht in gewünschter Weise zugegriffen werden. Details stehen im Logfile (eric.log).
ERIC_GLOBAL _UNGUELTIG E_INSTANZ	[610001080] Die übergebene Instanz ist gleich NULL oder bereits freigegeben worden.

ERIC_GLOBAL _NICHT_INITIA LISIERT	[610001081] Der Singlethread-ERiC wurde nicht mit EricInitialisiere initialisiert.
ERIC_GLOBAL _MEHRFACHE _INITIALISIER UNG	[610001082] Der Singlethread-ERiC wurde bereits mit EricInitialisiere initialisiert.
ERIC_GLOBAL _FEHLER_INIT IALISIERUNG	[610001083] Der angeforderte ERiC-Instanz konnte nicht erstellt werden. Details stehen ggf. im Logfile (eric.log).
ERIC_GLOBAL _UNKNOWN_ PARAMETER_ ERROR	[610001102] Unbekannter Parameterfehler.
ERIC_GLOBAL _CHECK_COR RUPTED_NDS	[610001108] Defekter Nutzdatensatz.
ERIC_GLOBAL _VERSCHLUE SSELUNGS_P ARAMETER_N ICHT_ANGEG EBEN	[610001206] Verschlüsselter/authentifizierter Versand gewünscht, aber keine notwendigen Verschlüsselungsparameter angegeben.
ERIC_GLOBAL _SEND_FLAG _MEHR_ALS_ EINES	[610001209] Es ist mehr als ein Versandflag angegeben.
ERIC_GLOBAL _UNGUELTIG E_FLAG_KOM BINATION	[610001218] Die übergebene Kombination von Bearbeitungsflags ist nicht erlaubt.
ERIC_GLOBAL _UNGUELTIG ER_PARAMET ER	[610001222] Die angegebenen Parameter sind ungültig oder unvollständig.
ERIC_GLOBAL _DRUCK_FUE R_VERFAHRE N_NICHT_ERL AUBT	[610001224] Für das angegebene Verfahren wird der Druck nicht unterstützt.

ERIC_GLOBAL _VERSAND_A RT_NICHT_UN TERSTUETZT	[610001225] Die Versandart ist für die angegebene Datenartversion nicht erlaubt.
ERIC_GLOBAL _UNGUELTIG E_PARAMETE R_VERSION	[610001226] Die Version eines der angegebenen Parameter ist ungültig.
ERIC_GLOBAL _TRANSFERH ANDLE	[610001227] Für das Verfahren Datenabholung wurde ein illegales Transferhandle angegeben.
ERIC_GLOBAL _PLUGININITI ALISIERUNG	[610001228] Die Initialisierung eines Plugins ist fehlgeschlagen.
ERIC_GLOBAL _INKOMPATIB LE_VERSIONE N	[610001229] Die Versionen der im Logfile genannten ERiC-Dateien sind nicht kompatibel. (Siehe eric.log.)
ERIC_GLOBAL _VERSCHLUE SSELUNGSVE RFAHREN_NI CHT_UNTERS TUETZT	[610001230] Das im XML-Element "<Verschlüsselung>" angegebene Verschlüsselungsverfahren wird vom ERiC nicht unterstützt.
ERIC_GLOBAL _MEHRFACHA UFRUFE_NIC HT_UNTERST UETZT	[610001231] Der Aufruf einer API-Funktion des ERiCs darf erst dann erfolgen, wenn ein vorheriger Aufruf zurückgekehrt ist.
ERIC_GLOBAL _UTI_COUNTR Y_NOT_SUPP ORTED	[610001404] Das Bundesland/Finanzamt mit der angegebenen Nummer nimmt bei der angegebenen Steuerart am ELSTER-Verfahren nicht teil.
ERIC_GLOBAL _IBAN_FORM ALER_FEHLE R	[610001501] Ungültige IBAN: IBAN muss aus zweistelligem Ländercode gefolgt von zweistelliger Prüfziffer gefolgt von der Basic Bank Account Number bestehen.

ERIC_GLOBAL_IBAN_LAENDERCODE_FEHLER	[610001502] Ungültige IBAN: Der angegebene Ländercode ist ungültig oder wird aktuell im ELSTER-Verfahren nicht unterstützt.
ERIC_GLOBAL_IBAN_LANDESFORMAT_FEHLER	[610001503] Ungültige IBAN: Die angegebene IBAN entspricht nicht dem für das angegebene Land definierten formalen Aufbau der IBAN oder die IBAN ist unzulässig.
ERIC_GLOBAL_IBAN_PRUEFZIFFER_FEHLER	[610001504] Ungültige IBAN: Die Prüfziffernberechnung zur angegebenen IBAN führt zu einer abweichenden Prüfziffer.
ERIC_GLOBAL_BIC_FORMALER_FEHLER	[610001510] Ungültiger BIC: Der formale Aufbau des angegebenen BIC ist ungültig.
ERIC_GLOBAL_BIC_LAENDERCODE_FEHLER	[610001511] Ungültiger BIC: Der angegebene Ländercode ist ungültig oder wird aktuell im ELSTER-Verfahren nicht unterstützt.
ERIC_GLOBAL_ZULASSUNGSNUMMER_ZU_LANG	[610001519] Die angegebene Zulassungsnummer entspricht nicht den Längenvorgaben. Es sind maximal 6 Stellen erlaubt.
ERIC_GLOBAL_IDNUMMER_UNGUELTIG	[610001525] Die übergebene IDNummer ist ungültig.
ERIC_GLOBAL_NULL_PARAMETER	[610001526] Es wurde der Parameter NULL übergeben.
ERIC_GLOBAL_EWAZ_UNGUELTIG	[610001527] Das übergebene Einheitswert-Aktenzeichen ist ungültig.
ERIC_GLOBAL_EWAZ_LANDESKUERZEL_UNBEKANNT	[610001528] Das übergebene Landeskürzel ist unbekannt oder leer.
ERIC_GLOBAL_UPDATE_NECCESSARY	[610001851] Update des ERiC erforderlich. Starten Sie nun das Update.

ERIC_GLOBAL _EINSTELLUN G_NAME_UNG UELTIG	[610001860] Ungültiger Name für Einstellung.
ERIC_GLOBAL _EINSTELLUN G_WERT_UN GUELTIG	[610001861] Ungültiger Wert für Einstellung.
ERIC_GLOBAL _ERR_DEKOD IEREN	[610001862] Fehler beim Dekodieren.
ERIC_GLOBAL _FUNKTION_N ICHT_UNTERS TUETZT	[610001863] Die aufgerufene Funktion wird nicht unterstützt.
ERIC_GLOBAL _NUTZDATEN TICKETS_NIC HT_EINDEUTI G	[610001865] Fehler im übergebenen EDS-XML: In den Sammeldaten wurde ein Nutzdatenticket für mehrere Steuerfälle verwendet. Für jeden Steuerfall muss jedoch ein eigenes Nutzdatenticket angegeben werden.
ERIC_GLOBAL _NUTZDATEN HEADERVERS IONEN_UNEIN HEITLICH	[610001866] Fehler im übergebenen EDS-XML: Bei den Sammeldaten wurden unterschiedliche Versionen des Nutzdaten-Headers verwendet. Innerhalb einer Datenlieferung ist jedoch nur eine Nutzdaten-Header-Version zulässig.
ERIC_GLOBAL _BUNDESLAE NDER_UNEIN HEITLICH	[610001867] Fehler im übergebenen EDS-XML: Es wurden Fälle für mehrere Bundesländer angegeben. Innerhalb einer Datenlieferung dürfen jedoch nur Fälle für ein Bundesland angegeben werden.
ERIC_GLOBAL _ZEITRAEUME _UNEINHEITLI CH	[610001868] Fehler im übergebenen EDS-XML: Es wurden Fälle für unterschiedliche Jahre angegeben. Innerhalb einer Datenlieferung dürfen jedoch nur Fälle für ein und dasselbe Jahr angegeben werden.

ERIC_GLOBAL_NUTZDATEN_HEADER_EMPFAENGER_NICHT_KORREKT	[610001869] Fehler im übergebenen EDS-XML: Der Inhalt des Nutzdaten-Elements "<Empfaenger>" ist für diese Datenart nicht korrekt.
ERIC_TRANSFER_COM_ERROR	[610101200] Allgemeiner Kommunikationsfehler.
ERIC_TRANSFER_VORGANG_NICHT_UNTERSTUETZT	[610101201] Dieser Vorgang wird von der aufgerufenen Funktion nicht unterstützt.
ERIC_TRANSFER_ERR_XML_HEADER	[610101210] Fehler im Transferheader. Der ELSTER-Annahmeserver hat einen Fehler zurückgemeldet. Bitte werten Sie die Serverantwort aus.
ERIC_TRANSFER_ERR_PARAMETER	[610101251] Es wurden ungültige Parameter übergeben.
ERIC_TRANSFER_ERR_DATENTEILENDNOTFOUND	[610101253] Im XML-String konnte der Text "</DatenTeil>" nicht gefunden werden.
ERIC_TRANSFER_ERR_BEGINN_DATENLIEFERANT	[610101255] Im XML-String konnte der Text "<DatenLieferant>" nicht gefunden werden.
ERIC_TRANSFER_ERR_ENDDATENLIEFERANT	[610101256] Im XML-String konnte der Text "</DatenLieferant>" nicht gefunden werden.
ERIC_TRANSFER_ERR_BEGINN_TRANSPORTSCHLUESSEL	[610101257] Im XML-String konnte der Text "<TransportSchluessel>" nicht gefunden werden.
ERIC_TRANSFER_ERR_ENDTRANSPORTSCHLUESSEL	[610101258] Im XML-String konnte der Text "</TransportSchluessel>" nicht gefunden werden.

ERIC_TRANSF ER_ERR_BEGI NDATENGRO ESSE	[610101259] Im XML-String konnte der Text "<DatenGroesse>" nicht gefunden werden.
ERIC_TRANSF ER_ERR_END DATENGROES SE	[610101260] Im XML-String konnte der Text "</DatenGroesse>" nicht gefunden werden.
ERIC_TRANSF ER_ERR_SEN D	[610101271] Beim Datenaustausch ist ein Fehler aufgetreten.
ERIC_TRANSF ER_ERR_NOT ENCRYPTED	[610101274] Die Antwortdaten waren nicht PKCS#7-verschlüsselt.
ERIC_TRANSF ER_ERR_PRO XYCONNECT	[610101276] Verbindung zum ProxyServer konnte nicht aufgebaut werden.
ERIC_TRANSF ER_ERR_CON NECTSERVER	[610101278] Zu den Servern konnte keine Verbindung aufgebaut werden.
ERIC_TRANSF ER_ERR_NOR ESPONSE	[610101279] Von der Clearingstelle konnte keine Antwort empfangen werden.
ERIC_TRANSF ER_ERR_PRO XYAUTH	[610101280] Der Proxyserver erwartet Anmeldedaten.
ERIC_TRANSF ER_ERR_SEN D_INIT	[610101282] Fehler bei der Initialisierung des Versands, Details stehen ggf. im Logfile (eric.log).
ERIC_TRANSF ER_ERR_TIME OUT	[610101283] Bei der Kommunikation mit dem Server kam es zu einer Zeitüberschreitung.
ERIC_TRANSF ER_ERR_PRO XYPORT_INVA LID	[610101284] Es wurde kein gültiger Port für den Proxy angegeben.
ERIC_TRANSF ER_ERR_OTH ER	[610101291] Sonstiger, nicht definierter Fehler aufgetreten.

ERIC_TRANSF ER_ERR_XML _NHEADER	[610101292] Fehler im NutzdatenHeader. Der ELSTER-Annahmeserver hat einen Fehler zurückgemeldet. Bitte werten Sie die Serverantwort aus. Bei Sammeldaten sind alle Nutzdatenblöcke zu prüfen, um den fehlerhaften Datensatz identifizieren zu können.
ERIC_TRANSF ER_ERR_XML _ENCODING	[610101293] Das XML liegt im falschen Encoding vor.
ERIC_TRANSF ER_ERR_END SIGUSER	[610101294] Im XML-String konnte der Text "</SigUser>" nicht gefunden werden.
ERIC_TRANSF ER_ERR_XML TAG_NICHT_G EFUNDEN	[610101295] Im XML-String konnte ein Tag nicht gefunden werden.
ERIC_TRANSF ER_ERR_DAT ENTEILFEHLE R	[610101297] Das XML-Element "<DatenTeil>" konnte nicht gelesen werden.
ERIC_TRANSF ER_EID_ZERT IFIKATFEHLE R	[610101500] Es konnte kein Ad Hoc-Zertifikat fuer den Personalausweis oder den Aufenthaltstitel erzeugt bzw. gefunden werden, Details stehen ggf. im Logfile (eric.log).
ERIC_TRANSF ER_EID_KEIN KONTO	[610101510] Für die Identifikationsnummer des Benutzers existiert kein Konto bei ELSTER.
ERIC_TRANSF ER_EID_IDNR NICHT EINDEU TIG	[610101511] Dem Benutzer konnte keine eindeutige Identifikationsnummer zugeordnet werden.
ERIC_TRANSF ER_EID_SERV ERFEHLER	[610101512] Das nPA-Servlet konnte keine Verbindung zum eID-Server aufbauen.
ERIC_TRANSF ER_EID_KEIN CLIENT	[610101520] Der eID-Client ist nicht erreichbar. Wahrscheinlich wurde er nicht gestartet oder die übergebene lokale URL ist nicht korrekt.

ERIC_TRANSF ER_EID_CLIE NTFEHLER	[610101521] Der eID-Client hat einen Fehler gemeldet. Details zu dem Fehler finden Sie im Log des eID-Clients oder ggf. im ERiC Logfile (eric.log).
ERIC_TRANSF ER_EID_FEHL ENDEFELDER	[610101522] Es konnten nicht alle benötigten Datenfelder des Personalausweises ausgelesen werden. Bitte prüfen Sie über die Funktion "Selbstauskunft" des eID-Clients, ob folgende Daten von Ihrem Personalausweis korrekt bereitgestellt werden: Familienname, Vorname(n), Geburtsdatum, Anschrift (mit Postleitzahl) und Dokumentenart.
ERIC_TRANSF ER_EID_IDEN TIFIKATIONAB GEBROCHEN	[610101523] Das Auslesen der Daten aus dem Personalausweis wurde vom Anwender abgebrochen.
ERIC_TRANSF ER_EID_NPAB LOCKIERT	[610101524] Der Personalausweis wird von einem anderen Vorgang blockiert. Beenden Sie den anderen Vorgang und versuchen Sie es dann erneut.
ERIC_CRYPT_ ERROR_CREA TE_KEY	[610201016] Fehler bei der Schlüsselerzeugung.
ERIC_CRYPT_ E_INVALID_H ANDLE	[610201101] eSigner: Ungültiges Token Handle.
ERIC_CRYPT_ E_MAX_SESSI ON	[610201102] eSigner: Zu viele Sessions geöffnet.
ERIC_CRYPT_ E_BUSY	[610201103] eSigner: Überlastung.
ERIC_CRYPT_ E_OUT_OF_M EM	[610201104] eSigner: Speicherzuordnungsfehler.
ERIC_CRYPT_ E_PSE_PATH	[610201105] eSigner: Ungültiger PSE Pfad.
ERIC_CRYPT_ E_PIN_WRON G	[610201106] eSigner: Es wurde ein falsches Passwort bzw. eine falsche PIN angegeben.

ERIC_CRYPT_ E_PIN_LOCKE D	[610201107] eSigner: Das Passwort bzw. die PIN ist gesperrt.
ERIC_CRYPT_ E_P7_READ	[610201108] eSigner: Fehler beim Lesen des PKCS#7-Objekts.
ERIC_CRYPT_ E_P7_DECODE	[610201109] eSigner: Fehler beim PKCS#7 Dekodieren.
ERIC_CRYPT_ E_P7_RECEIPT NT	[610201110] eSigner: Entschlüsselungszertifikat nicht in Empfängerliste enthalten.
ERIC_CRYPT_ E_P12_READ	[610201111] eSigner: Fehler beim Lesen des PKCS#12-Objekts.
ERIC_CRYPT_ E_P12_DECODE	[610201112] eSigner: Fehler beim Dekodieren des PKCS#12-Objekts.
ERIC_CRYPT_ E_P12_SIG_KEY	[610201113] eSigner: Fehler beim Zugriff auf Soft-PSE-Signaturschlüssel.
ERIC_CRYPT_ E_P12_ENC_KEY	[610201114] eSigner: Fehler beim Zugriff auf Soft-PSE Entschlüsselungsschlüssel.
ERIC_CRYPT_ E_P11_SIG_KEY	[610201115] eSigner: Fehler beim Zugriff auf Hard-Token Signaturschlüssel.
ERIC_CRYPT_ E_P11_ENC_KEY	[610201116] eSigner: Fehler beim Zugriff auf Hard-Token Entschlüsselungsschlüssel.
ERIC_CRYPT_ E_XML_PARSE	[610201117] eSigner: Fehler beim Parsen der XML-Eingabedatei.
ERIC_CRYPT_ E_XML_SIG_ADD	[610201118] eSigner: Fehler beim Erzeugen des XML-Signaturasts.
ERIC_CRYPT_ E_XML_SIG_TAG	[610201119] eSigner: XML-Signaturtag nicht vorhanden.

ERIC_CRYPT_ E_XML_SIG_ SIGN	[610201120] eSigner: Fehler bei XML-Signaturerzeugung.
ERIC_CRYPT_ E_ENCODE_ UNKNOWN	[610201121] eSigner: Parameter-Fehler, unbekanntes Encoding.
ERIC_CRYPT_ E_ENCODE_ ERROR	[610201122] eSigner: Encoding-Fehler.
ERIC_CRYPT_ E_XML_INIT	[610201123] eSigner: XML Initialisierungsfehler.
ERIC_CRYPT_ E_ENCRYPT	[610201124] eSigner: Fehler beim Verschlüsseln.
ERIC_CRYPT_ E_DECRYPT	[610201125] eSigner: Fehler beim Entschlüsseln.
ERIC_CRYPT_ E_P11_SLOT_ EMPTY	[610201126] eSigner: Keine Signaturkarte eingesteckt (PKCS#11).
ERIC_CRYPT_ E_NO_SIG_EN C_KEY	[610201127] eSigner: Keine Signatur-/Verschlüsselungs-Zertifikate/-Schlüssel gefunden (PKCS#11).
ERIC_CRYPT_ E_LOAD_DLL	[610201128] eSigner: PKCS11 bzw. PC/SC Library fehlt oder ist nicht ausführbar.
ERIC_CRYPT_ E_NO_SERVIC E	[610201129] eSigner: Der PC/SC Dienst ist nicht gestartet.
ERIC_CRYPT_ E_ESICL_EXC EPTION	[610201130] eSigner: Unbekannte Ausnahme aufgetreten.
ERIC_CRYPT_ E_ESIGNER_N ICHT_GELADE N	[610201140] eSigner: Die eSigner-Bibliothek konnte nicht geladen werden

ERIC_CRYPT_ E_INKOMPATI BLE_ESIGNER _VERSION	[610201141] eSigner: Die eSigner-Bibliothek liegt in einer inkompatiblen Version vor
ERIC_CRYPT_ E_VERALTET E_ESIGNER_V ERSION	[610201142] eSigner: Die eSigner-Bibliothek liegt in einer veralteten Version vor
ERIC_CRYPT_ E_TOKEN_TY PE_MISMATC H	[610201144] eSigner: CA Tokentyp und interner Tokentyp stimmen nicht überein.
ERIC_CRYPT_ E_P12_CREAT E	[610201146] eSigner: Temporäres PKCS#12-Token kann nicht erzeugt werden.
ERIC_CRYPT_ E_VERIFY_CE RT_CHAIN	[610201147] eSigner: Zertifikatskette konnte nicht verifiziert werden.
ERIC_CRYPT_ E_P11_ENGIN E_LOADED	[610201148] eSigner: PKCS#11 Engine mit anderer Bibliothek belegt.
ERIC_CRYPT_ E_USER_CAN CEL	[610201149] eSigner: Aktion vom Benutzer abgebrochen.
ERIC_CRYPT_ ZERTIFIKAT	[610201200] Fehler beim Zugriff auf Zertifikat.
ERIC_CRYPT_ SIGNATUR	[610201201] Fehler bei Signaturerzeugung.
ERIC_CRYPT_ NICHT_UNTE RSTUETZTES _PSE_FORMA T	[610201203] Das Format der PSE wird nicht unterstützt.
ERIC_CRYPT_ PIN_BENOETI GT	[610201205] Für die ausgewählte Operation muss ein Passwort bzw. eine PIN angegeben werden.

ERIC_CRYPT_ PIN_STAERKE _NICHT_AUSR EICHEND	[610201206] Das gewünschte Passwort ist nicht sicher genug (z.B. zu kurz).
ERIC_CRYPT_ E_INTERN	[610201208] Interner Fehler aufgetreten. Details stehen ggf. im Logfile (eric.log).
ERIC_CRYPT_ ZERTIFIKATS PFAD_KEIN_V ERZEICHNIS	[610201209] Der angegebene Zertifikatspfad ist kein Verzeichnis.
ERIC_CRYPT_ ZERTIFIKATS DATEI_EXISTI ERT_BEREITS	[610201210] Im angegebenen Verzeichnis existiert bereits ein Bestandteil eines ERiC-Zertifikats.
ERIC_CRYPT_ PIN_ENTHAEL T_UNGUELTIG E_ZEICHEN	[610201211] Das gewünschte Passwort enthält ungültige Zeichen (z.B. Umlaute).
ERIC_CRYPT_ CORRUPTED	[610201213] Das übergebene Zertifikat weist Inkonsistenzen auf und kann deswegen nicht verwendet werden. Bitte verwenden Sie ein anderes oder erzeugen und verwenden Sie ein neues Zertifikat.
ERIC_CRYPT_ EIDKARTE_NI CHT_UNTERS TUETZT	[610201214] Die aufgerufene Funktion unterstützt den neuen Personalausweis (nPA) und den elektronischen Aufenthaltstitel (eAT) nicht.
ERIC_CRYPT_ E_SC_SLOT_E MPTY	[610201215] Es ist keine Karte/kein Stick eingesteckt.
ERIC_CRYPT_ E_SC_NO_AP PLET	[610201216] Kein unterstütztes Applet gefunden.
ERIC_CRYPT_ E_SC_SESSIO N	[610201217] Fehler in der Kartensession.
ERIC_CRYPT_ E_P11_NO_SI G_CERT	[610201218] P11 Signaturzertifikat fehlt.

ERIC_CRYPT_ E_P11_INIT_F AILED	[610201219] P11 Der initiale Tokenzugriff ist fehlgeschlagen.
ERIC_CRYPT_ E_P11_NO_EN C_CERT	[610201220] P11 Verschlüsselungszertifikat fehlt.
ERIC_CRYPT_ E_P12_NO_SI G_CERT	[610201221] P12 Signaturzertifikat fehlt.
ERIC_CRYPT_ E_P12_NO_EN C_CERT	[610201222] P12 Verschlüsselungszertifikat fehlt.
ERIC_CRYPT_ E_SC_ENC_K EY	[610201223] PC/SC Der Zugriff auf den Entschlüsselungsschlüssel ist fehlgeschlagen.
ERIC_CRYPT_ E_SC_NO_SIG _CERT	[610201224] PC/SC Signaturzertifikat fehlt.
ERIC_CRYPT_ E_SC_NO_EN C_CERT	[610201225] PC/SC Verschlüsselungszertifikat fehlt.
ERIC_CRYPT_ E_SC_INIT_FA ILED	[610201226] PC/SC Der initiale Tokenzugriff ist fehlgeschlagen.
ERIC_CRYPT_ E_SC_SIG_KE Y	[610201227] PC/SC Der Zugriff auf den Signaturschlüssel ist fehlgeschlagen.
ERIC_CRYPT_ E_DATA_NOT _INITIALIZED	[610201228] Die Datenstruktur ist nicht initialisiert
ERIC_CRYPT_ E_ASN1_REA D_BUFFER_T OO_SMALL	[610201229] Der Lesebuffer zum Dekodieren der ASN.1-Struktur ist zu klein
ERIC_CRYPT_ E_ASN1_REA D_DATA_INCO MPLETE	[610201230] Die Daten der ASN.1-Struktur sind unvollständig

ERIC_CRYPT_ E_ASN1_NO_ ENVELOPED_ DATA	[610201231] Die ASN.1-Struktur enthält kein EnvelopedData
ERIC_CRYPT_ E_ASN1_NO_ CONTENT_DA TA	[610201232] Die ASN.1-Struktur enthält keine Daten
ERIC_IO_FEH LER	[610301001] Verarbeitung fehlerhaft, keine genaueren Informationen vorhanden.
ERIC_IO_DAT EI_INKORREK T	[610301005] Der Dateiaufbau ist nicht korrekt.
ERIC_IO_PAR SE_FEHLER	[610301006] Fehler beim Parsen der Eingabedaten. Details stehen im Logfile (eric.log).
ERIC_IO_NDS _GENERIERU NG_FEHLGES CHLAGEN	[610301007] Die Generierung des Nutzdatensatzes ist fehlgeschlagen.
ERIC_IO_MAS TERDATENSE RVICE_NICHT _VERFUEGBA R	[610301010] Interner Fehler, der Masterdatenservice ist nicht verfügbar.
ERIC_IO_STE UERZEICHEN _IM_NDS	[610301014] Es wurden ungültige Steuerzeichen im Nutzdatensatz gefunden.
ERIC_IO_VER SIONSINFOR MATIONEN_NI CHT_GEFUND EN	[610301031] Die Versionsinformationen der ERiC-Bibliotheken konnten nicht ausgelesen werden.
ERIC_IO_FAL SCHES_VERF AHREN	[610301104] Der Wert im Transferheader-Element "Verfahren" wird vom verwendeten Reader nicht unterstützt.

ERIC_IO_READER_MEHRFACHE_STEUERFAELLE	[610301105] Es wurde mehr als ein Steuerfall in der Eingabedatei gefunden.
ERIC_IO_READER_UNERWARTETE_ELEMENTE	[610301106] Es wurden unerwartete Elemente in der Eingabedatei gefunden, Details stehen ggf. im Logfile (eric.log).
ERIC_IO_READER_FORMALE_FEHLER	[610301107] Es wurden formale Fehler in der Eingabedatei gefunden, Details stehen ggf. im Logfile (eric.log).
ERIC_IO_READER_FALSCHES_ENCODING	[610301108] Die Eingabedaten lagen nicht im Encoding UTF-8 ohne BOM vor oder es war kein Encoding spezifiziert.
ERIC_IO_READER_MEHRFACHE_NUTZDATEN_ELEMENTE	[610301109] Es wurde mehr als ein "Nutzdaten"-Element in der Eingabedatei gefunden.
ERIC_IO_READER_MEHRFACHE_NUTZDATENBLOCK_ELEMENTE	[610301110] Es wurde mehr als ein Nutzdatenblock in der Eingabedatei gefunden.
ERIC_IO_UNBEKANNTE_DATENART	[610301111] Der im Transferheader-Element "Datenart" angegebene Wert ist unbekannt.
ERIC_IO_READER_UNTERSACHBEREICH_UNGUELTIG	[610301114] Ungültiger oder fehlender Wert für den Untersuchungsbereich.
ERIC_IO_READER_ZU_VIELE_NUTZDATENBLOCK_ELEMENTE	[610301115] Es wurden zu viele Nutzdatenblöcke in der Eingabedatei gefunden.

ERIC_IO_READER_STEUERZEICHEN_IM_TRANSFERHEADER	[610301150] Es wurden ungültige Steuerzeichen im TransferHeader-Element gefunden.
ERIC_IO_READER_STEUERZEICHEN_IM_NUTZDATENHEADER	[610301151] Es wurden ungültige Steuerzeichen im NutzdatenHeader-Element gefunden.
ERIC_IO_READER_STEUERZEICHEN_IN_DEN_NUTZDATEN	[610301152] Es wurden ungültige Steuerzeichen im Nutzdaten-Element gefunden.
ERIC_IO_READER_RABE_Fehler	[610301170] Es wurden Fehler in den Angaben zur Referenzierung von Belegen (RABE) gefunden. Details stehen im Logfile (eric.log).
ERIC_IO_READER_KEINE_RABEID	[610301171] Es wurde keine Rabeld angegeben, Details stehen ggf. im Logfile (eric.log)
ERIC_IO_READER_RABEID_UNGUELTIG	[610301172] Es wurde eine ungültige Rabeld gefunden, Details stehen ggf. im Logfile (eric.log)
ERIC_IO_READER_RABE_VERIFIKATIONSID_UNGUELTIG	[610301173] Es wurde eine ungültige Verifikationsld gefunden, Details stehen ggf. im Logfile (eric.log)
ERIC_IO_READER_RABE_REFERENZID_UNGUELTIG	[610301174] Es wurde eine ungültige Referenzld gefunden, Details stehen ggf. im Logfile (eric.log)
ERIC_IO_READER_RABE_REFERENZID_NICHT_ERLAUBT	[610301175] Es wurde eine Referenzld für ein Feld angegeben, das keine Referenzld erlaubt. Details stehen ggf. im Logfile (eric.log)

ERIC_IO_READER_RABEFERENZIDSNICHT_EINDEUTIG	[610301176] Für einen Nutzdatenblock wurde mehrfach die gleiche ReferenzId angegeben. Details stehen ggf. im Logfile (eric.log)
ERIC_IO_READER_ZU_VIELE_ANHAENGE	[610301190] Ein Nutzdatenblock enthält zu viele Anhänge. Details stehen im Logfile (eric.log).
ERIC_IO_READER_ANHANG_ZU_GROSS	[610301191] Ein Anhang ist zu groß. Details stehen im Logfile (eric.log).
ERIC_IO_READER_ANHAENGE_ZU_GROSS	[610301192] Die Gesamtgröße aller Anhänge in einem Nutzdatenblock ist zu groß. Details stehen im Logfile (eric.log).
ERIC_IO_READER_ANHANG_ZU_KLEIN	[610301193] Der referenzierte Anhang ist zu klein und muss in das XML eingebettet werden. Details stehen im Logfile (eric.log).
ERIC_IO_READER_SCHEMA_VALIDIERUNGSGEFehler	[610301200] Es traten Fehler beim Validieren des XML auf. Details stehen im Logfile (eric.log).
ERIC_IO_READER_UNBEKANNTE_XML_ENTITY	[610301201] Eine XML-Entity konnte nicht aufgelöst werden.
ERIC_IO_TESTHERSTELLERID_GESPERRT	[610301202] Die im XML angegebene Hersteller-ID ist gesperrt. Bitte verwenden Sie Ihre eigene Hersteller-ID auch für Testfälle.
ERIC_IO_DATENTEILNOTFOUND	[610301252] Im XML-String konnte der Text "<DatenTeil>" nicht gefunden werden.
ERIC_IO_DATENTEILENDNOTFOUND	[610301253] Im XML-String konnte der Text "</DatenTeil>" nicht gefunden werden.

ERIC_IO_UEBERGABEPARAMETER_FELDERHAFT	[610301300] Falsche Übergabeparameter für die Funktion.
ERIC_IO_UNGÜLTIGE_UTF8_SEQUENZ	[610301400] Der Parameter enthält ungültige UTF-8 Multibytesequenzen.
ERIC_IO_UNGÜLTIGE_ZEICHEN_IN_PARAMETER	[610301401] Der Parameter enthält mindestens ein unzulässiges Zeichen.
ERIC_PRINT_INTERNER_FELDER	[610501001] Verarbeitung fehlerhaft, keine genaueren Informationen vorhanden.
ERIC_PRINT_DRUCKVORLAGE_NICHT_GEFUNDEN	[610501002] Keine Druckvorlage für die angegebene Kombination aus Unterfallart und Veranlagungszeitraum gefunden. Bitte prüfen Sie die installierten Druckvorlagen.
ERIC_PRINT_UNGÜLTIGER_DATEIPfad	[610501004] Es wurde ein falscher Dateipfad angegeben, es fehlen Zugriffsrechte oder die Datei wird aktuell von einer anderen Anwendung verwendet.
ERIC_PRINT_INITIALIZIERUNG_FELDERHAFT	[610501007] ERiCPrint wurde nicht richtig initialisiert. Eventuell wurde ERiC nicht richtig initialisiert?
ERIC_PRINT_AUSGABEZIEL_UNBEKANNT	[610501008] Das zu verwendende Format bzw. der Zielklient sind nicht bekannt.
ERIC_PRINT_ABBRUCH_DRUCKVORBEREITUNG	[610501009] Der Beginn des Ausdruckprozesses schlug fehl. Eventuell konnten notwendige Ressourcen nicht allokiert werden.
ERIC_PRINT_ABBRUCH_GENERIERUNG	[610501010] Während der Ausgabe der Inhalte ist ein Fehler aufgetreten.

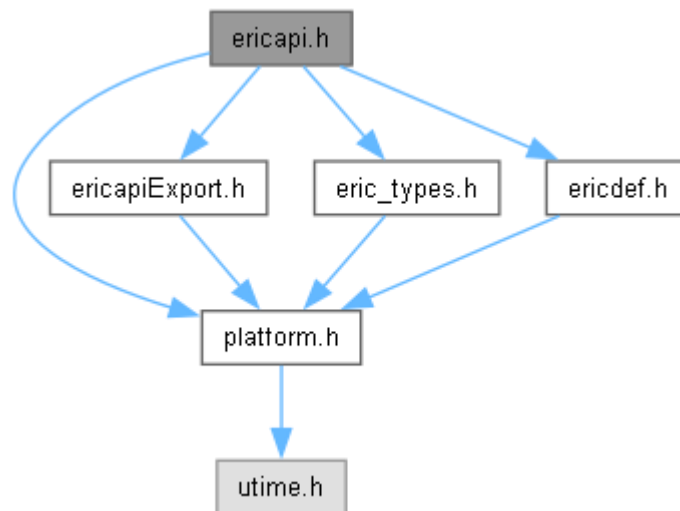
ERIC_PRINT_STEUFALL_NICHT_UNTERSTUETZT	[610501011] Die Kombination aus Unterfallart und Veranlagungszeitraum wird nicht unterstützt.
ERIC_PRINT_FUSSTEXT_ZU_LANG	[610501012] Der übergebene Fußtext ist zu lang.
ERIC_PRINT_PDFCALLBACK	[610501015] Bei der PDF-Erstellung hat die benutzerdefinierte Callback-Funktion einen Fehler gemeldet.

ericapi.h-Dateireferenz

Deklaration der ERiC API-Funktionen für die Singlethreading-API.

```
#include "platform.h"
#include "ericapiExport.h"
#include "eric_types.h"
#include "ericdef.h"
```

Include-Abhängigkeitsdiagramm für ericapi.h:



Funktionen

- [ERICAPI_IMPORT](#) int [EricBearbeiteVorgang](#) (const char *datenpuffer, const char *datenartVersion, [uint32_t](#) bearbeitungsFlags, const [eric_druck_parameter_t](#) *druckParameter, const [eric_verschluesselungs_parameter_t](#) *cryptoParameter, [EricTransferHandle](#) *transferHandle, [EricRueckgabepufferHandle](#) rueckgabeXmlPuffer, [EricRueckgabepufferHandle](#) serverantwortXmlPuffer)

Diese API-Funktion ist die zentrale Schnittstellenfunktion zur Kommunikation mit dem ELSTER-Annahmeserver.

- [ERICAPI_IMPORT](#) int [EricBeende](#) (void)
Beendet den Singlethreading-ERiC.
- [ERICAPI_IMPORT](#) int [EricChangePassword](#) (const [byteChar](#) *psePath, const [byteChar](#) *oldPin, const [byteChar](#) *newPin)

Die PIN für ein clientseitig erzeugtes Zertifikat (CEZ) wird geändert.

- [ERICAPI_IMPORT](#) int [EricPruefeBuFaNummer](#) (const [byteChar](#) *steuernummer)
Die Bundesfinanzamtsnummer wird überprüft.
- [ERICAPI_IMPORT](#) int [EricCheckXML](#) (const char *xml, const char *datenartVersion, [EricRueckgabepufferHandle](#) fehlerTextPuffer)
Das xml wird gegen das Schema der datenartVersion validiert.
- [ERICAPI_IMPORT](#) int [EricCloseHandleToCertificate](#) ([EricZertifikatHandle](#) hToken)
Das Zertifikat-Handle hToken wird freigegeben.
- [ERICAPI_IMPORT](#) int [EricCreateKey](#) (const [byteChar](#) *pin, const [byteChar](#) *pfad, const [eric_zertifikat_parameter_t](#) *zertifikatInfo)
Es werden die Kryptomittel für ein clientseitig erzeugtes Zertifikat (CEZ) in einem Verzeichnis des Dateisystems erstellt.
- [ERICAPI_IMPORT](#) int [EricCreateTH](#) (const char *xml, const char *verfahren, const char *datenart, const char *vorgang, const char *testmerker, const char *herstellerId, const char *datenLieferant, const char *versionClient, const [byteChar](#) *publicKey, [EricRueckgabepufferHandle](#) xmlRueckgabePuffer)
Diese Funktion erzeugt einen TransferHeader.
- [ERICAPI_IMPORT](#) int [EricCreateUUID](#) ([EricRueckgabepufferHandle](#) uuidRueckgabePuffer)
Erzeugt einen Version 4 Universally Unique Identifier (UUID) gemäß RFC 4122.
- [ERICAPI_IMPORT](#) int [EricDekodiereDaten](#) ([EricZertifikatHandle](#) zertifikatHandle, const [byteChar](#) *pin, const [byteChar](#) *base64Eingabe, [EricRueckgabepufferHandle](#) rueckgabePuffer)
Es werden die mit der Datenabholung abgeholten und verschlüsselten Daten entschlüsselt.
- [ERICAPI_IMPORT](#) int [EricEinstellungAlleZuruecksetzen](#) (void)
Alle Einstellungen werden auf den jeweiligen Standardwert zurückgesetzt.

- [ERICAPI_IMPORT](#) int [EricEinstellungLesen](#) (const char *name, [EricRueckgabepufferHandle](#) rueckgabePuffer)

Der Wert der API-Einstellung `name` wird im `rueckgabePuffer` zurückgeliefert.
- [ERICAPI_IMPORT](#) int [EricEinstellungSetzen](#) (const char *name, const char *wert)

Die API-Einstellung `name` wird auf den `wert` gesetzt.
- [ERICAPI_IMPORT](#) int [EricEinstellungZuruecksetzen](#) (const char *name)

Der Wert der API-Einstellung `name` wird auf den Standardwert zurückgesetzt.
- [ERICAPI_IMPORT](#) int [EricEntladePlugins](#) (void)

Alle verwendeten Plugin-Bibliotheken werden entladen und deren Speicher wird freigegeben.
- [ERICAPI_IMPORT](#) int [EricFormatEWaz](#) (const [byteChar](#) *ewAzElster, [EricRueckgabepufferHandle](#) ewAzBescheidPuffer)

Konvertiert ein Einheitswert-Aktenzeichen im ELSTER-Format in ein landesspezifisches Bescheidformat.
- [ERICAPI_IMPORT](#) int [EricFormatStNr](#) (const [byteChar](#) *eingabeSteuernummer, [EricRueckgabepufferHandle](#) rueckgabePuffer)

Die Steuernummer `eingabeSteuernummer` wird in das Bescheid-Format des jeweiligen Bundeslandes umgewandelt.
- [ERICAPI_IMPORT](#) int [EricGetAuswahlListen](#) (const char *datenartVersion, const char *feldkennung, [EricRueckgabepufferHandle](#) rueckgabeXmlPuffer)

Die Auswahlliste(n) für `datenartVersion` oder `feldkennung` wird zurückgeliefert.
- [ERICAPI_IMPORT](#) int [EricGetErrorMessageFromXMLAnswer](#) (const char *xml, [EricRueckgabepufferHandle](#) transferticketPuffer, [EricRueckgabepufferHandle](#) returncodeTHPuffer, [EricRueckgabepufferHandle](#) fehlertextTHPuffer, [EricRueckgabepufferHandle](#) returncodesUndFehlertexteNDHXmlPuffer)

Aus dem Antwort-XML des Finanzamtsservers wird das Transferticket und Returncodes/Fehlermeldungen zurückgegeben.

- [ERICAPI_IMPORT](#) int [EricGetHandleToCertificate](#) ([EricZertifikatHandle](#) *hToken, [uint32_t](#) *iInfoPinSupport, const [byteChar](#) *pathToKeystore)

Für das übergebene Zertifikat in `pathToKeystore` wird das Handle `hToken` und die unterstützten PIN-Werte `iInfoPinSupport` zurückgeliefert.
- [ERICAPI_IMPORT](#) int [EricZertifikatOeffnenAusBytes](#) ([EricZertifikatHandle](#) *hToken, const [byteChar](#) *pkcs12Container, [uint32_t](#) containerGroesse, const [byteChar](#) *pkcs12Passwort)

Für das im Speicher übergebene Zertifikat wird das Handle `hToken` zurückgeliefert.
- [ERICAPI_IMPORT](#) int [EricGetPinStatus](#) ([EricZertifikatHandle](#) hToken, [uint32_t](#) *pinStatus, [uint32_t](#) keyType)

Der PIN-Status wird für ein passwortgeschütztes Kryptomittel abgefragt und in `pinStatus` zurückgegeben.
- [ERICAPI_IMPORT](#) int [EricGetPublicKey](#) (const [eric_verschluesselungs_parameter_t](#) *cryptoParameter, [EricRueckgabepufferHandle](#) rueckgabePuffer)

Es wird der öffentliche Schlüssel als base64-kodierte Zeichenkette für das übergebene Zertifikat-Handle in `cryptoParameter` zurückgeliefert. Von den beiden öffentlichen Schlüsseln, die in ELSTER-Zertifikaten enthalten sind, wird der Verschlüsselungsschlüssel (Encryption Key) zurückgegeben.
- [ERICAPI_IMPORT](#) int [EricHoleFehlerText](#) (int fehlerkode, [EricRueckgabepufferHandle](#) rueckgabePuffer)

Es wird die Klartextfehlermeldung zu dem `fehlerkode` ermittelt.
- [ERICAPI_IMPORT](#) int [EricHoleFinanzaemter](#) (const [byteChar](#) *finanzamtLandNummer, [EricRueckgabepufferHandle](#) rueckgabeXmlPuffer)

Es wird die Finanzamtliste für eine bestimmte `finanzamtLandNummer` zurückgegeben.
- [ERICAPI_IMPORT](#) int [EricHoleFinanzamtLandNummern](#) ([EricRueckgabepufferHandle](#) rueckgabeXmlPuffer)

Die Liste aller Finanzamtlandnummern wird zurückgegeben.

- [ERICAPI_IMPORT](#) int [EricHoleFinanzamtsdaten](#) (const [byteChar](#) bufaNr[5], [EricRueckgabepufferHandle](#) rueckgabeXmlPuffer)

Die finanzamtsdaten werden für eine Bundesfinanzamtsnummer zurückgegeben.
- [ERICAPI_IMPORT](#) int [EricHoleTestfinanzaemter](#) ([EricRueckgabepufferHandle](#) rueckgabeXmlPuffer)

Die Testfinanzamtliste wird in rueckgabeXmlPuffer zurückgegeben.
- [ERICAPI_IMPORT](#) int [EricHoleZertifikatEigenschaften](#) ([EricZertifikatHandle](#) hToken, const [byteChar](#) *pin, [EricRueckgabepufferHandle](#) rueckgabeXmlPuffer)

Die Eigenschaften des übergebenen Zertifikats werden im rueckgabeXmlPuffer zurückgegeben.
- [ERICAPI_IMPORT](#) int [EricHoleZertifikatFingerabdruck](#) (const [eric_verschluesselungs_parameter_t](#) *cryptoParameter, [EricRueckgabepufferHandle](#) fingerabdruckPuffer, [EricRueckgabepufferHandle](#) signaturPuffer)

Der Fingerabdruck und dessen Signatur wird für das übergebene Zertifikat zurückgegeben.
- [ERICAPI_IMPORT](#) int [EricInitialisiere](#) (const [byteChar](#) *pluginPfad, const [byteChar](#) *logPfad)

Initialisiert den Singlethreading-ERiC.
- [ERICAPI_IMPORT](#) int [EricMakeElsterStnr](#) (const [byteChar](#) *steuernrBescheid, const [byteChar](#) landesnr[2+1], const [byteChar](#) bundesfinanzamtsnr[4+1], [EricRueckgabepufferHandle](#) steuernrPuffer)

Es wird eine Steuernummer im ELSTER-Steuernummerformat erzeugt.
- [ERICAPI_IMPORT](#) int [EricMakeElsterEWaz](#) (const [byteChar](#) *ewAzBescheid, const [byteChar](#) *landeskuerzel, [EricRueckgabepufferHandle](#) ewAzElsterPuffer)

Konvertiert ein Einheitswert-Aktenzeichen in das ELSTER-Format.
- [ERICAPI_IMPORT](#) int [EricPruefeBIC](#) (const [byteChar](#) *bic)

Die bic wird auf Gültigkeit überprüft.
- [ERICAPI_IMPORT](#) int [EricPruefeIBAN](#) (const [byteChar](#) *iban)

Die iban wird auf Gültigkeit überprüft.

- [ERICAPI_IMPORT](#) int [EricPruefeEWAZ](#) (const [byteChar](#) *einheitswertAz)
Überprüft ein Einheitswert-Aktenzeichen im ELSTER-Format auf Gültigkeit.
- [ERICAPI_IMPORT](#) int [EricPruefeIdentifikationsMerkmal](#) (const [byteChar](#) *steuerId)
Die steuerId wird auf Gültigkeit überprüft.
- [ERICAPI_IMPORT](#) int [EricPruefeSteuernummer](#) (const [byteChar](#) *steuernummer)
Die steuernummer wird einschließlich Bundesfinanzamtsnummer auf formale Richtigkeit geprüft.
- [ERICAPI_IMPORT](#) int [EricPruefeWIdNr](#) (const [byteChar](#) *wIdNr)
Die Wirtschafts-Identifikationsnummer (W-IdNr.) wird auf formale Gültigkeit überprüft.
- [ERICAPI_IMPORT](#) int [EricPruefeZertifikatPin](#) (const [byteChar](#) *pathToKeystore, const [byteChar](#) *pin, [uint32_t](#) keyType)
Prüft, ob die pin zum Zertifikat pathToKeystore passt. Nicht anwendbar auf Ad Hoc-Zertifikate (AHZ), die für einen neuen Personalausweis (nPA) ausgestellt sind.
- [ERICAPI_IMPORT](#) int [EricRegistriereFortschrittCallback](#) ([EricFortschrittCallback](#) funktion, void *benutzerdaten)
Die funktion wird als Callback-Funktion für [EricBearbeiteVorgang\(\)](#) registriert.
- [ERICAPI_IMPORT](#) int [EricRegistriereGlobalenFortschrittCallback](#) ([EricFortschrittCallback](#) funktion, void *benutzerdaten)
Die registrierte funktion wird als Callback-Funktion von [EricBearbeiteVorgang\(\)](#) aufgerufen und zeigt den Gesamtfortschritt der Verarbeitung an.
- [ERICAPI_IMPORT](#) int [EricRegistriereLogCallback](#) ([EricLogCallback](#) funktion, [uint32_t](#) schreibeEricLogDatei, void *benutzerdaten)

Die registrierte *funktion* wird als Callback-Funktion für jede Lognachricht aufgerufen. Die Ausgabe entspricht einer Zeile im *eric.log*.

- [ERICAPI_IMPORT](#) [EricRueckgabepufferHandle](#) [EricRueckgabepufferErzeugen](#) (void)

Diese API-Funktion erzeugt einen Rückgabepuffer und gibt ein Handle darauf zurück.

- [ERICAPI_IMPORT](#) int [EricRueckgabepufferFreigeben](#) ([EricRueckgabepufferHandle](#) handle)

Der durch das *handle* bezeichnete Rückgabepuffer wird freigegeben.

- [ERICAPI_IMPORT](#) const char * [EricRueckgabepufferInhalt](#) ([EricRueckgabepufferHandle](#) handle)

Der durch das *handle* bezeichnete Inhalt des Rückgabepuffers wird zurückgegeben.

- [ERICAPI_IMPORT](#) uint32_t [EricRueckgabepufferLaenge](#) ([EricRueckgabepufferHandle](#) handle)

Die Länge des Rückgabepufferinhalts wird zurückgegeben.

- [ERICAPI_IMPORT](#) int [EricSystemCheck](#) (void)

Es werden Plattform-, Betriebssystem- und ERiC-Informationen ausgegeben.

- [ERICAPI_IMPORT](#) int [EricVersion](#) ([EricRueckgabepufferHandle](#) rueckgabeXmlPuffer)

Es wird eine Liste sämtlicher Produkt- und Dateiversionen der verwendeten ERiC-Bibliotheken als XML-Daten zurückgegeben.

Ausführliche Beschreibung

Deklaration der ERiC API-Funktionen für die Singlethreading-API.

Dokumentation der Funktionen

[ERICAPI_IMPORT](#) int EricBearbeiteVorgang (const char * datenpuffer, const char * datenartVersion, uint32_t bearbeitungsFlags, const [eric_druck_parameter_t](#) * druckParameter, const [eric_verschluesselungs_parameter_t](#) * cryptoParameter, [EricTransferHandle](#) * transferHandle, [EricRueckgabepufferHandle](#) rueckgabeXmlPuffer, [EricRueckgabepufferHandle](#) serverantwortXmlPuffer)

Diese API-Funktion ist die zentrale Schnittstellenfunktion zur Kommunikation mit dem ELSTER-Annahmeserver.

Als Austauschformat wird XML verwendet, siehe Kap. "Datenverarbeitung mit ERiC" im [ERiC-Entwicklerhandbuch.pdf](#). Dort sind die Arbeitsabläufe von Einzel- und Sammellieferung beschrieben.

Die Funktion kann Steuerdaten plausibilisieren, an den ELSTER-Annahmeserver übertragen und ausdrucken, sowie Protokolle der Übertragung erzeugen. Die ProcessingFlags im Parameter bearbeitungsFlags definieren, welche der Schritte wie ausgeführt werden.

Je nach Anwendungsfall können die Daten authentifiziert übertragen werden und es kann ein PDF-Druck der Daten erfolgen. In diesen Fällen sind die Parameter cryptoParameter und druckParameter entsprechend zu befüllen. Die möglichen Parameterkombinationen und Druckkennzeichnungen können im [ERiC-Entwicklerhandbuch.pdf](#) nachgelesen werden.

Sind für einen Anwendungsfall mehrere voneinander abhängige Aufrufe von [EricBearbeiteVorgang\(\)](#) nötig, so ist der Parameter transferHandle zu übergeben. Dies ist derzeit nur für die Datenabholung der Fall.

Es werden an bestimmten Punkten der Verarbeitung benutzerdefinierte Callback Funktionen aufgerufen. Siehe hierzu [Fortschrittcallbacks](#).

Zur Erzeugung, Verwendung und Freigabe von Rückgabepuffern siehe Dokumentation zu [EricRueckgabepufferHandle](#).

Parameter:

in	<i>datenpuffer</i>	Enthält die zu verarbeitenden XML-Daten.
in	<i>datenartVersion</i>	Die datenartVersion ist der Datenartversionmatrix.xml zu entnehmen. Dieser Parameter darf nicht NULL sein und muss zu den XML-Eingangsdaten passen. Siehe auch ERiC-Entwicklerhandbuch.pdf .

in	<i>bearbeitungsFlags</i>	<p>Oder-Verknüpfung von Bearbeitungsvorgaben. Anhand dieser Vorgaben werden die übergebenen Daten verarbeitet. Der Parameter darf nicht 0 sein, zu gültigen Werten siehe eric_bearbeitung_flag_t.</p> <p>Bei welchen Anwendungsfällen welche Flags möglich oder notwendig sind, ist im ERiC-Entwicklerhandbuch.pdf nachzulesen.</p>
in	<i>druckParameter</i>	<p>Parameter, der für den PDF-Druck benötigt wird, siehe eric_druck_parameter_t.</p> <p>Bei welchen Anwendungsfällen der Druckparameter möglich oder notwendig ist, ist im ERiC-Entwicklerhandbuch.pdf nachzulesen.</p> <p>Soll kein PDF-Druck erfolgen, so ist NULL zu übergeben.</p>
in	<i>cryptoParameter</i>	<p>Enthält die für den authentifizierten Versand benötigten Informationen und darf nur dann übergeben werden, siehe eric_verschluesselungs_parameter_t.</p> <p>Erfolgt kein authentifizierter Versand, so ist NULL zu übergeben.</p>
in,out	<i>transferHandle</i>	<p>Bei der Datenabholung ist ein Zeiger auf ein vom Aufrufer verwaltetes und anfangs mit 0 befülltes EricTransferHandle zu übergeben, über das die zusammenhängenden Versandvorgänge einer Datenabholung gebündelt werden (Bündelung der Versandvorgänge "Anforderung", "Abholung" und optional "Quittierung").</p> <p>Wenn bei der Datenabholung kein Versandflag gesetzt ist (nur Validierung), darf dem transferHandle auch ein Nullzeiger (NULL) übergeben werden.</p> <p>Bei allen anderen Anwendungsfällen ist immer NULL zu übergeben.</p>
out	<i>rueckgabeXmlPuffer</i>	<p>Handle auf einen Rückgabepuffer, in den beim Versand Telenummer und Ordnungsbegriff, Hinweise oder Fehler bei der Regelprüfung geschrieben werden.</p> <p>Siehe Inhalt des Rückgabepuffers und des Serverantwortpuffers und EricRueckgabepufferHandle.</p>
out	<i>serverantwortXmlPuffer</i>	<p>Handle auf einen Rückgabepuffer, in den beim Versand die Antwort des Empfangsservers geschrieben wird.</p> <p>Siehe Inhalt des Rückgabepuffers und des Serverantwortpuffers und EricRueckgabepufferHandle.</p>

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_UNGUELTIGER_PARAMETER](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_DATENARTVERSION_UNBEKANNT](#)
- [ERIC_GLOBAL_VERSCHLUESSELUNGS_PARAMETER_NICHT_ANGEGEBEN](#)
- [ERIC_GLOBAL_PRUEF_FEHLER](#) Plausibilitätsfehler in den Eingabedaten, die Fehlermeldungen werden im Rückgabepuffer `rueckgabeXmlPuffer` zurückgegeben. Siehe Abschnitt [Plausibilitätsfehler](#).
- [ERIC_GLOBAL_HINWEISE](#) Kann nur zurückgegeben werden, falls das Bearbeitungsflag [ERIC_PRUEFE_HINWEISE](#) angegeben wurde. Es wurden ausschließlich Hinweise zu den Eingabedaten gemeldet, die Hinweise werden im Rückgabepuffer `rueckgabeXmlPuffer` zurückgegeben. Siehe Abschnitt [Hinweise](#).
- [ERIC_GLOBAL_DATENSATZ_ZU_GROSS](#) Die maximal zulässige Größe des XML-Eingangsdatensatzes oder des zu übermittelnden, komprimierten, verschlüsselten und base64-kodierten Datenteils, siehe [ERIC-Entwicklerhandbuch.pdf](#), Kap. "Größenbegrenzung der Eingangsdaten", ist überschritten.
- [ERIC_TRANSFER_ERR_XML_HEADER](#), [ERIC_TRANSFER_ERR_XML_NHEADER](#) Die Serverantwort enthält Fehlermeldungen. Zur Auswertung kann entweder die Serverantwort selbst ausgewertet werden oder es wird [EricGetErrorMessageFromXMLAnswer\(\)](#) aufgerufen.
- [ERIC_IO_READER_SCHEMA_VALIDIERUNGSFEHLER](#)
- [ERIC_IO_PARSE_FEHLER](#)
- [ERIC_GLOBAL_COMMONDATA_NICHT_VERFUEGBAR](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- weitere, siehe [eric_fehlercodes.h](#)

Inhalt des Rückgabepuffers und des Serverantwortpuffers

Der Inhalt der Pufferspeicher kann mit [EricRueckgabepufferInhalt\(\)](#) abgefragt und ausgewertet werden. `rueckgabeXmlPuffer` gibt im [Erfolgsfall](#) oder bei [Plausibilitätsfehler](#) XML-Daten nach Schema [Dokumentation\API-Rueckgabe-Schemata\EricBearbeiteVorgang.xsd](#) zurück. `serverantwortXmlPuffer` gibt bei Sendevorgängen die Antwort des ELSTER- Annahmeservers zurück.

Nach dem Aufruf der Funktion müssen programmatisch folgende Fälle aufgrund des Rückgabewerts unterschieden werden.

Erfolgsfall

Sind alle Bearbeitungsschritte fehlerfrei durchlaufen worden, dann ist der Rückgabewert [ERIC_OK](#) und der Text im Pufferspeicher `rueckgabeXmlPuffer` enthält beim Versand XML-Daten mit generierter Telenummer und bei Neuaufnahmen den Ordnungsbegriff.

Beispiel:

```
<?xml version="1.0" encoding="UTF-8"?>
<EricBearbeiteVorgang
xmlns="http://www.elster.de/EricXML/1.1/EricBearbeiteVorgang">
  <Erfolg>
    <Telenummer>N55</Telenummer>
  </Erfolg>
</EricBearbeiteVorgang>
```

Beim Versand befindet sich zusätzlich im Pufferspeicher `serverantwortXmlPuffer` die Antwort des ELSTER-Annahmeservers. Bei einer Datenabholung kann diese ausgewertet werden. Details hierzu befinden sich im [ERiC-Entwicklerhandbuch.pdf](#)

Hinweise

Falls das Bearbeitungsflag [ERIC_PRUEFE_HINWEISE](#) angegeben worden ist, kann der Rückgabewert [ERIC_GLOBAL_HINWEISE](#) zurückgegeben werden. Der Rückgabepuffer enthält dann die gemeldeten Hinweise.

Beispiel:

```
<?xml version="1.0" encoding="UTF-8"?>
<EricBearbeiteVorgang
xmlns="http://www.elster.de/EricXML/1.1/EricBearbeiteVorgang">
  <Hinweis>
    <Nutzdatenticket>1075</Nutzdatenticket>
    <Feldidentifikator>100001</Feldidentifikator>
    <Mehrfachzeilenindex>1</Mehrfachzeilenindex>
    <LfdNrVordruck>1</LfdNrVordruck>
    <VordruckZeilennummer>4</VordruckZeilennummer>
    <SemantischerIndex>PersonA</SemantischerIndex>
    <Untersachsbereich>5</Untersachsbereich>
    <RegelName>testRegelName</RegelName>
    <FachlicheHinweisId>9995</FachlicheHinweisId>
    <Text>Weitere Angaben können erforderlich sein</Text>
```

```
</Hinweis>
</EricBearbeiteVorgang>
```

Die einzelnen Elemente sind in der Schemadefinition

[Dokumentation\API-Rueckgabe-Schemata\EricBearbeiteVorgang.xsd](#)

dokumentiert. Wenn die Bearbeitungsflags [ERIC_PRUEFE_HINWEISE](#) und [ERIC_VALIDIERE](#) übergeben worden sind, wurden bei der Plausibilisierung keine Fehler gefunden. Es sind keine Fehler im Rückgabepuffer enthalten.

Plausibilitätsfehler

Bei fehlgeschlagener Plausibilitätsprüfung ist der Rückgabewert

[ERIC_GLOBAL_PRUEF_FEHLER](#), und die Fehler werden im Rückgabepuffer als XML-Daten zurückgeliefert.

Beispiel:

```
<?xml version="1.0" encoding="UTF-8"?>
<EricBearbeiteVorgang
xmlns="http://www.elster.de/EricXML/1.1/EricBearbeiteVorgang">
  <FehlerRegelpruefung>
    <Nutzdatenticket>1075</Nutzdatenticket>
    <Feldidentifikator>100001</Feldidentifikator>
    <Mehrfachzeilenindex>1</Mehrfachzeilenindex>
    <LfdNrVordruck>1</LfdNrVordruck>
    <VordruckZeilennummer>4</VordruckZeilennummer>
    <SemantischerIndex>PersonA</SemantischerIndex>
    <Untersachsbereich>5</Untersachsbereich>
    <RegelName>testRegelName</RegelName>
    <FachlicheFehlerId>9995</FachlicheFehlerId>
    <Text>Beim Ankreuzfeld muss der Wert 'X' angegeben werden.</Text>
  </FehlerRegelpruefung>
</EricBearbeiteVorgang>
```

Die einzelnen Elemente sind in der Schemadefinition

[Dokumentation\API-Rueckgabe-Schemata\EricBearbeiteVorgang.xsd](#)

dokumentiert. Wenn die Bearbeitungsflags [ERIC_PRUEFE_HINWEISE](#) und [ERIC_VALIDIERE](#) übergeben worden sind, kann der Rückgabepuffer auch Hinweise enthalten.

Fehler in der Serverantwort

Ist der Rückgabewert [ERIC_TRANSFER_ERR_XML_THEADER](#) oder

[ERIC_TRANSFER_ERR_XML_NHEADER](#) so enthält der Serverantwortpuffer Fehlermeldungen. Zur Auswertung kann entweder die Serverantwort selbst

ausgewertet werden oder es wird [EricGetErrormessagesFromXMLAnswer\(\)](#) aufgerufen.

Sonstige Fehler

Bei sonstigen Fehlern ist der Inhalt der Rückgabepuffer undefiniert. Um nähere Informationen über die Fehlerursache herauszufinden, kann [EricHoleFehlerText\(\)](#) mit dem Rückgabewert aufgerufen werden.

Fortschrittcallbacks

Während der Verarbeitung eines Anwendungsfalls werden die durch die Funktionen [EricRegistriereFortschrittCallback\(\)](#) und [EricRegistriereGlobalenFortschrittCallback\(\)](#) registrierten Callbacks aufgerufen.

Siehe auch:

- [ERiC-Entwicklerhandbuch.pdf](#), Kap. "Anwendungsfälle von EricBearbeiteVorgang()"
- [ERiC-Entwicklerhandbuch.pdf](#), Kap. der jeweiligen Datenart
- [ERiC-Entwicklerhandbuch.pdf](#), Kap. "ElsterDatenabholung"
- [ERiC-Entwicklerhandbuch.pdf](#), Kap. "Größenbegrenzung der Eingangsdaten"
- [ERiC-Entwicklerhandbuch.pdf](#), Kap. "Funktionen für Fortschrittcallbacks"
- [EricHoleFehlerText\(\)](#)
- [EricGetErrormessagesFromXMLAnswer\(\)](#)
- [EricRegistriereFortschrittCallback\(\)](#)
- [EricRegistriereGlobalenFortschrittCallback\(\)](#)

ERICAPI_IMPORT int EricBeende (void)

Beendet den Singlethreading-ERiC.

Die Verarbeitung mit der ERiC Singlethread-API ist beendet, als letztes muss [EricBeende\(\)](#) aufgerufen werden.

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)

Siehe auch:

- [EricInitialisiere\(\)](#)

[ERICAPI_IMPORT](#) int EricChangePassword (const [byteChar](#) * psePath, const [byteChar](#) * oldPin, const [byteChar](#) * newPin)

Die PIN für ein clientseitig erzeugtes Zertifikat (CEZ) wird geändert.

Die Funktion ändert die bei der Funktion [EricCreateKey\(\)](#) angegebene PIN und entsprechend hierfür die Prüfsumme in der Datei `eric.sfv`. Falls die Datei `eric.sfv` nicht vorhanden ist, wird sie, wie bei [EricCreateKey\(\)](#), erstellt. Eine PIN-Änderung von einem Portalzertifikat (POZ) ist nicht möglich.

Pfade müssen auf Windows in der für Datei-Funktionen benutzten ANSI-Codepage, auf Linux, AIX und Linux Power in der für das Dateisystem benutzten Locale und auf macOS in der "decomposed form" von UTF-8 übergeben werden. Bitte weitere Betriebssystemspezifika bzgl. nicht erlaubter Zeichen in Pfaden und Pfadtrennzeichen beachten.

Für Details zu Pfaden im ERiC siehe [ERiC-Entwicklerhandbuch.pdf](#), Kapitel "Übergabe von Pfaden an ERiC API-Funktionen"

Parameter:

in	<i>psePath</i>	In dem angegebenen Pfad liegt das Schlüsselpaar <code>eric_private.p12</code> und <code>eric_public.cer</code>
in	<i>oldPin</i>	Bisherige PIN.
in	<i>newPin</i>	Neue PIN. Die Mindestlänge beträgt 4 Stellen. Zulässige Zeichen sind alle ASCII-Zeichen ohne die Steuerzeichen.

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_UNGUELTIGER_PARAMETER](#)
- [ERIC_GLOBAL_UNGUELTIGER_PARAMETER](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)
- [ERIC_CRYPT_PIN_STAERKE_NICHT_AUSREICHEND](#)
- [ERIC_CRYPT_PIN_ENTHAELT_UNGUELTIGE_ZEICHEN](#)
- [ERIC_CRYPT_E_PSE_PATH](#)
- [ERIC_CRYPT_NICHT_UNTERSTUETZTES_PSE_FORMAT](#)
- [ERIC_CRYPT_ERROR_CREATE_KEY](#)

Siehe auch:

- [EricCreateKey\(\)](#)
- [ERiC-Entwicklerhandbuch.pdf](#), Kap. "Zuordnung der API-Funktionen zur Verwendung von POZ, CEZ und AHZ"

[ERICAPI_IMPORT](#) int EricCheckXML (const char * xml, const char * datenartVersion, [EricRueckgabepufferHandle](#) fehlertextPuffer)

Das xml wird gegen das Schema der datenartVersion validiert.

Das verwendete Schema kann nachgeschlagen werden unter [Dokumentation\Schnittstellenbeschreibungen\](#)

Nicht unterstützte Datenartversionen:

- ElsterKMV
- alle Bilanz Datenartversionen

Parameter:

in	<i>xml</i>	XML-Zeichenfolge
in	<i>datenartVersion</i>	Die datenartVersion ist der Datenartversionmatrix.xml zu entnehmen. Dieser Parameter darf nicht NULL sein und muss zu den XML-Eingangsdaten passen. Siehe auch ERIC-Entwicklerhandbuch.pdf .
out	<i>fehlertextPuffer</i>	Handle auf einen Rückgabepuffer, in den Fehlertexte geschrieben werden. Zur Erzeugung, Verwendung und Freigabe von Rückgabepuffern siehe Dokumentation zu EricRueckgabepufferHandle .

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_FUNKTION_NICHT_UNTERSTUETZT](#):
Schemavalidierung wird für die übergebene datenartVersion nicht unterstützt.
- [ERIC_GLOBAL_COMMONDATA_NICHT_VERFUEGBAR](#)
- [ERIC_GLOBAL_DATENARTVERSION_UNBEKANNT](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_IO_READER_SCHEMA_VALIDIERUNGSFEHLER](#):
Die Fehlerbeschreibung steht im fehlertextPuffer .
- [ERIC_IO_PARSE_FEHLER](#):
Die Fehlerbeschreibung steht im fehlertextPuffer .
- weitere, siehe [eric_fehlercodes.h](#)

ERICAPI_IMPORT int EricCloseHandleToCertificate (EricZertifikatHandle hToken)

Das Zertifikat-Handle `hToken` wird freigegeben.

Diese Funktion gibt das übergebene Zertifikat-Handle frei. Zertifikat-Handles sollten möglichst frühzeitig, d.h. wenn sie nicht mehr benötigt werden, mit [EricCloseHandleToCertificate\(\)](#) freigegeben werden, spätestens jedoch zum Programmende bzw. vor dem Entladen der ericapi Bibliothek.

Das Ad Hoc-Zertifikat eines neuen Personalausweises sollte immer genau dann freigegeben werden, wenn es nicht mehr benötigt wird, jedoch spätestens vor Ablauf der 24 Stunden, die das Ad Hoc-Zertifikat gültig ist.

Tritt ein Fehler auf, kann die Fehlermeldung mit [EricHoleFehlerText\(\)](#) ausgelesen werden.

Parameter:

in	<i>hToken</i>	Zertifikat-Handle wie von der Funktion EricGetHandleToCertificate() zurückgeliefert.
----	---------------	--

Rückgabe:

- [ERIC_OK](#)
- [ERIC_CRYPT_E_INVALID_HANDLE](#)
- [ERIC_GLOBAL_UNGUELTIGER_PARAMETER](#)
- [ERIC_GLOBAL_NICHT_GENUEGENDE_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)

Zu beachten:

Die folgenden Rückgabewerte gelten nur bei Verwendung des neuen Personalausweises.

Rückgabe:

- [ERIC_TRANSFER_EID_CLIENTFEHLER](#)
- [ERIC_TRANSFER_EID_FEHLENDEFELDER](#)
- [ERIC_TRANSFER_EID_IDENTIFIKATIONABGEBROCHEN](#)
- [ERIC_TRANSFER_EID_NPABLOCKIERT](#)
- [ERIC_TRANSFER_EID_IDNRNICHTEINDEUTIG](#)
- [ERIC_TRANSFER_EID_KEINCLIENT](#)
- [ERIC_TRANSFER_EID_KEINKONTO](#)
- [ERIC_TRANSFER_EID_SERVERFEHLER](#)
- [ERIC_TRANSFER_ERR_CONNECTSERVER](#)
- [ERIC_TRANSFER_ERR_NORESPONSE](#)
- [ERIC_TRANSFER_ERR_PROXYAUTH](#)
- [ERIC_TRANSFER_ERR_PROXYCONNECT](#)

- [ERIC_TRANSFER_ERR_SEND](#)
- [ERIC_TRANSFER_ERR_SEND_INIT](#)
- [ERIC_TRANSFER_ERR_TIMEOUT](#)

Siehe auch:

- [EricGetHandleToCertificate\(\)](#)
- [EricZertifikatOeffnenAusBytes\(\)](#)
- [EricGetPinStatus\(\)](#)
- [ERiC-Entwicklerhandbuch.pdf](#), Kap. "Authentifizierung mit dem neuen Personalausweis (nPA)"

ERICAPI_IMPORT int EricCreateKey (const byteChar * pin, const byteChar * pfad, const eric_zertifikat_parameter_t * zertifikatInfo)

Es werden die Kryptomittel für ein clientseitig erzeugtes Zertifikat (CEZ) in einem Verzeichnis des Dateisystems erstellt.

Im angegebenen Verzeichnis `pfad` sind nach Ausführung der Funktion [EricCreateKey\(\)](#) drei Dateien erstellt worden:

- `eric_public.cer`:
Enthält das Zertifikat mit den Daten aus `zertifikatInfo` und darin den öffentlichen Schlüssel.
- `eric_private.pl2`:
Enthält den privaten Schlüssel. Der Zugriff ist über die `pin` geschützt.
- `eric.sfv`:
Enthält die Prüfsumme der Dateien `eric_public.cer` und `eric_private.pl2`. Die Integrität dieser beiden Dateien kann damit jederzeit überprüft werden.

Ein CEZ kann unter anderem für die Bescheiddaten-Rückübermittlung verwendet werden. Weitere Informationen zur Datenabholung lesen Sie bitte im [ERiC-Entwicklerhandbuch.pdf](#) nach.

Parameter:

in	<i>pin</i>	PIN (Passwort), mit der auf den privaten Schlüssel zugegriffen werden kann. Die Mindestlänge beträgt 4 Stellen. Zulässige Zeichen sind alle ASCII-Zeichen ohne die Steuerzeichen.
in	<i>pfad</i>	Pfad (1), in dem die Kryptomittel erzeugt werden sollen. Das durch den angegebenen Pfad bezeichnete Verzeichnis muss im Dateisystem bereits existieren und beschreibbar sein. Es gibt folgende Möglichkeiten: <ul style="list-style-type: none"> ◦ Absoluter Pfad: Empfehlung. ◦ Relativer Pfad: Wird an das Arbeitsverzeichnis angehängt. ◦ Leere Zeichenkette: In diesem Fall wird das Arbeitsverzeichnis verwendet.
in	<i>zertifikatInfo</i>	Daten, die zur Identifikation des Schlüsselinhabers im Zertifikat abgelegt werden.

(1) Pfade müssen auf Windows in der für Datei-Funktionen benutzten ANSI-Codepage, auf Linux, AIX und Linux Power in der für das Dateisystem benutzten Locale und auf macOS in der "decomposed form" von UTF-8

übergeben werden. Bitte weitere Betriebssystemspezifika bzgl. nicht erlaubter Zeichen in Pfaden und Pfadtrennzeichen beachten.

Für Details zu Pfaden im ERiC siehe [ERiC-Entwicklerhandbuch.pdf](#), Kapitel "Übergabe von Pfaden an ERiC API-Funktionen".

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_UNGUELTIGE_PARAMETER_VERSION](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)
- [ERIC_CRYPT_ZERTIFIKATSPFAD_KEIN_VERZEICHNIS](#)
- [ERIC_CRYPT_ZERTIFIKATSDATEI_EXISTIERT_BEREITS](#)
- [ERIC_CRYPT_PIN_STAERKE_NICHT_AUSREICHEND](#)
- [ERIC_CRYPT_PIN_ENTHAELT_UNGUELTIGE_ZEICHEN](#)
- [ERIC_CRYPT_ERROR_CREATE_KEY](#)

Siehe auch:

- [EricChangePassword\(\)](#)
- [ERiC-Entwicklerhandbuch.pdf](#), Kap. "Zertifikate und Authentifizierungsverfahren"
- [ERiC-Entwicklerhandbuch.pdf](#), Kap. "Übergabe von Pfaden an ERiC API-Funktionen"

[ERICAPI_IMPORT](#) int EricCreateTH (const char * xml, const char * verfahren, const char * datenart, const char * vorgang, const char * testmerker, const char * herstellerId, const char * datenLieferant, const char * versionClient, const [byteChar](#) * publicKey, [EricRueckgabepufferHandle](#) xmlRueckgabePuffer)

Diese Funktion erzeugt einen TransferHeader.

Dieser ist der oberste Header in der Datenstruktur. Er enthält Felder für die Kommunikation zwischen Server und Client. Es wird nur die Kombination NutzdatenHeader-Version "11" und TransferHeader-Version "11" unterstützt.

Zur Erzeugung, Verwendung und Freigabe von Rückgabepuffern siehe Dokumentation zu [EricRueckgabepufferHandle](#).

Parameter:

in	<i>xml</i>	<p>XML-Datensatz, für den der <TransferHeader>-Block erzeugt werden soll.</p> <p>Es kann entweder ein komplettes Elster-XML oder nur der Datenteil übergeben werden.</p> <p>ERIC nimmt bei diesem Parameter keine Konvertierung von Sonderzeichen in Entitätenreferenzen vor.</p> <p>Attribute, die in den Start-Tags der Elemente <Elster> bzw. <DatenTeil> im übergebenen XML-Datensatz definiert werden, werden nicht in das Rückgabe-XML übernommen.</p> <p>Namespace-Definitionen, die in den Start-Tags der Elemente <Elster> bzw. <DatenTeil> im übergebenen XML-Datensatz definiert werden, führen zu einem ERIC IO PARSE FEHLER.</p> <p>Im Rückgabe-XML werden im Start-Tag des Elements <Elster> die URI "http://www.elster.de/elsterxml/schema/v11" als Default-Namensraum definiert.</p> <p>Die dem Element <DatenTeil> untergeordneten Elemente aus dem übergebenen XML-Datensatz werden unverändert übernommen.</p> <p>Der allgemeine Aufbau des Elster-XMLs wird im ERIC-Entwicklerhandbuch.pdf, Kap. "Datenverarbeitung mit ERIC" beschrieben.</p>
in	<i>verfahren</i>	<p>Name des Verfahrens, z.B: 'ElsterAnmeldung', siehe ERIC-Entwicklerhandbuch.pdf, Tabelle "Eigenschaften der Datenart" im jeweiligen Kapitel zur Datenart.</p>

in	<i>datenart</i>	Name der Datenart, z.B.: 'LStB' oder 'UStVA', siehe ERiC-Entwicklerhandbuch.pdf , Tabelle "Eigenschaften der Datenart" im jeweiligen Kapitel zur Datenart.
in	<i>vorgang</i>	Name der Übertragungsart, z.B. 'send-NoSig', siehe ERiC-Entwicklerhandbuch.pdf , Tabelle "Eigenschaften der Datenart" im jeweiligen Kapitel zur Datenart.
in	<i>testmerker</i>	Für eine Testübertragung muss der entsprechende Testmerker angegeben werden, siehe ERiC-Entwicklerhandbuch.pdf , Kap. "Test Unterstützung bei der ERiC-Anbindung". Falls ein Echtfall übertragen werden soll, muss der Wert NULL angegeben werden.
in	<i>herstellerId</i>	Hersteller-ID des Softwareproduktes.
in	<i>datenLieferant</i>	Der Wert entspricht dem XML-Element <DatenLieferant>, wie es im Schema des Transferheaders der ElsterBasis-XML-Schnittstelle definiert ist. ERiC konvertiert bei diesem Parameter Sonderzeichen in Entitätenreferenzen.
in	<i>versionClient</i>	Angabe von Versionsinformation, die in der Serverantwort auch zurückgegeben wird und ausgewertet werden kann. Der Wert NULL entspricht "keine Angabe von Versionsinformation", d.h. es wird kein XML-Element <VersionClient> im <TransferHeader>-Block erzeugt. ERiC konvertiert bei diesem Parameter Sonderzeichen in Entitätenreferenzen.
in	<i>publicKey</i>	Öffentlicher Schlüssel für die Transportverschlüsselung beim Verfahren ElsterLohn. Bei anderen Verfahren sollte NULL übergeben werden. Dieser Wert kann mit dem Rückgabewert von EricGetPublicKey() befüllt werden. Der Inhalt dieses Parameters wird in das <TransportSchluessel>-Element der Rückgabe-XML geschrieben.

out	<i>xmlRueckgabePuffer</i>	Handle auf einen Rückgabepuffer, in den das Elster-XML mit dem erzeugten TransportHeader geschrieben wird, siehe EricRueckgabepufferHandle . Es wird immer ein vollständiger Elster-XML-Datensatz mit dem <Elster>-Element als Wurzel-Element zurückgeliefert. Bzgl. der darin enthaltenen XML-Namespace-Definitionen sind die bei der Beschreibung des Parameters "xml" genannten Einschränkungen zu berücksichtigen.
-----	---------------------------	---

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_TRANSFER_ERR_XML_ENCODING](#): Die übergebenen XML-Daten sind nicht UTF-8 kodiert.
- [ERIC_IO_PARSE_FEHLER](#)
- [ERIC_IO_DATENTEILNOTFOUND](#)
- [ERIC_IO_DATENTEILENDNOTFOUND](#)
- weitere, siehe [eric fehlercodes.h](#)

Siehe auch:

- [ERiC-Entwicklerhandbuch.pdf](#), Kap. "Datenverarbeitung mit ERiC"
- [ERiC-Entwicklerhandbuch.pdf](#), Kap. "Anwendungsfälle von EricBearbeiteVorgang()"
- ERiC-Returncodes und Fehlertexte sind in [eric fehlercodes.h](#) zu finden.

[ERICAPI_IMPORT](#) int EricCreateUUID ([EricRueckgabepufferHandle](#)
[uuidRueckgabePuffer](#))

Erzeugt einen Version 4 Universally Unique Identifier (UUID) gemäß RFC 4122.

Parameter:

out	<i>uuidRueckgabe Puffer</i>	Handle auf einen Rückgabepuffer, in den die erzeugte UUID geschrieben wird.
-----	---------------------------------	--

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)

ERICAPI_IMPORT int EricDekodiereDaten ([EricZertifikatHandle](#) zertifikatHandle, const [byteChar](#) * pin, const [byteChar](#) * base64Eingabe, [EricRueckgabepufferHandle](#) rueckgabePuffer)

Es werden die mit der Datenabholung abgeholten und verschlüsselten Daten entschlüsselt.

Falls während der Bearbeitung ein Fehler auftritt, liefert die Funktion [EricHoleFehlerText\(\)](#) den dazugehörigen Fehlertext.

Parameter:

in	<i>zertifikatHandle</i>	Handle auf das zum Entschlüsseln zu verwendende Zertifikat.
in	<i>pin</i>	PIN zum Zugriff auf das Zertifikat.
in	<i>base64Eingabe</i>	Base64-kodierte verschlüsselte Daten oder Anhänge, welche mit dem Verfahren ElsterDatenabholung abgeholt wurden. Die Abholdaten befinden sich im Element /Elster[1]/DatenTeil[1]/Nutzdatenblock/Nutzdaten[1]/Datenabholung[1]/Abholung[1]/Datenpaket Die optionalen Anhänge befinden sich im Element /Elster[1]/DatenTeil[1]/Nutzdatenblock/Nutzdaten[1]/Datenabholung[1]/Abholung[1]/Anhaenge[1]/Anhang[1]/Dateiinhalt
out	<i>rueckgabePuffer</i>	Handle auf einen Rückgabepuffer, in den die entschlüsselten Daten geschrieben werden. Im Fehlerfall ist der Inhalt des Rückgabepuffers undefiniert. Zur Erzeugung, Verwendung und Freigabe von Rückgabepuffern siehe EricRueckgabepufferHandle .

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_ERR_DEKODIEREN](#)
- [ERIC_GLOBAL_UNKNOWN](#)
- Ein Zertifikatsfehler aus dem Statuscodebereich von [ERIC_CRYPT_E_INVALID_HANDLE](#) = 610201101 bis 610201212

Siehe auch:

- [EricHoleFehlerText\(\)](#)

- [ERiC-Entwicklerhandbuch.pdf](#), Kap. "ElsterDatenabholung"

ERICAPI_IMPORT int EricEinstellungAlleZuruecksetzen (void)

Alle Einstellungen werden auf den jeweiligen Standardwert zurückgesetzt.

Die Standardwerte sind im Dokument [ERiC-Entwicklerhandbuch.pdf](#), Kap. "Vorbelegung der ERiC-Einstellungen" zu finden.

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_UNKNOWN](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)

Siehe auch:

- [EricEinstellungSetzen\(\)](#)
- [EricEinstellungLesen\(\)](#)
- [EricEinstellungZuruecksetzen\(\)](#)
- [ERiC-Entwicklerhandbuch.pdf](#), Kap. "Bedeutung der ERiC-Einstellungen"

[ERICAPI_IMPORT](#) int EricEinstellungLesen (const char * name, [EricRueckgabepufferHandle](#) rueckgabePuffer)

Der Wert der API-Einstellung `name` wird im `rueckgabePuffer` zurückgeliefert.

Parameter:

in	<i>name</i>	Name der API-Einstellung, NULL-terminierte Zeichenfolge.
out	<i>rueckgabePuffer</i>	Handle auf einen Rückgabepuffer, in den der Wert der API-Einstellung geschrieben wird. Zur Erzeugung, Verwendung und Freigabe von Rückgabepuffern siehe EricRueckgabepufferHandle .

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_EINSTELLUNG_NAME_UNGUELTIG](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)

Siehe auch:

- [EricEinstellungSetzen\(\)](#)
- [EricEinstellungZuruecksetzen\(\)](#)
- [EricEinstellungAlleZuruecksetzen\(\)](#)
- [ERiC-Entwicklerhandbuch.pdf](#), Kap. "Bedeutung der ERiC-Einstellungen"

ERICAPI_IMPORT int EricEinstellungSetzen (const char * name, const char * wert)

Die API-Einstellung `name` wird auf den `wert` gesetzt.

Nach dem Laden der ERiC-Bibliotheken hat jede API-Einstellung ihren Standardwert. Mit dieser Funktion kann der Wert verändert werden. Der Wertebereich der jeweiligen API-Einstellung ist zu beachten.

Bei Pfad-Einstellungen muss auf Windows der Wert in der für Datei-Funktionen benutzten ANSI-Codepage, auf Linux, AIX und Linux Power in der für das Dateisystem benutzten Locale und auf macOS in der "decomposed form" von UTF-8 übergeben werden. Bitte weitere Betriebssystemspezifika bzgl. nicht erlaubter Zeichen in Pfaden und Pfadtrennzeichen beachten.

Für Details zu Pfaden im ERiC siehe [ERiC-Entwicklerhandbuch.pdf](#), Kapitel "Übergabe von Pfaden an ERiC API-Funktionen"

Parameter:

in	<i>name</i>	Name der API-Einstellung, NULL-terminierte Zeichenfolge.
in	<i>wert</i>	Wert der API-Einstellung, NULL-terminierte Zeichenfolge.

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_EINSTELLUNG_NAME_UNGUELTIG](#)
- [ERIC_GLOBAL_EINSTELLUNG_WERT_UNGUELTIG](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)

Siehe auch:

- [EricEinstellungLesen\(\)](#)
- [EricEinstellungZuruecksetzen\(\)](#)
- [EricEinstellungAlleZuruecksetzen\(\)](#)
- [ERiC-Entwicklerhandbuch.pdf](#), Kap. "Bedeutung der ERiC-Einstellungen"

ERICAPI_IMPORT int EricEinstellungZuruecksetzen (const char * name)

Der Wert der API-Einstellung `name` wird auf den Standardwert zurückgesetzt.

Die Standardwerte sind im Dokument [ERiC-Entwicklerhandbuch.pdf](#), Kap. "Vorbelegung der ERiC-Einstellungen" zu finden.

Parameter:

in	<i>name</i>	Name der API-Einstellung, NULL-terminierte Zeichenfolge.
----	-------------	--

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_EINSTELLUNG_NAME_UNGUELTIG](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)

Siehe auch:

- [EricEinstellungSetzen\(\)](#)
- [EricEinstellungLesen\(\)](#)
- [EricEinstellungAlleZuruecksetzen\(\)](#)
- [ERiC-Entwicklerhandbuch.pdf](#), Kap. "Bedeutung der ERiC-Einstellungen"

ERICAPI_IMPORT int EricEntladePlugins (void)

Alle verwendeten Plugin-Bibliotheken werden entladen und deren Speicher wird freigegeben.

Der ERiC lädt die für die Bearbeitung notwendigen Plugin-Bibliotheken permanent in den Speicher und gibt diese erst mit dem Aufruf dieser Funktion wieder frei.

Zu beachten:

[EricEntladePlugins\(\)](#) sollte erst dann aufgerufen werden, wenn die Plugin-Bibliotheken definitiv nicht mehr benötigt werden. Ein erneutes Laden der Bibliotheken ist verhältnismäßig zeitintensiv.

Falls eine Plugin-Bibliothek nicht entladen werden kann, wird dies in eric.log protokolliert. Der Returncode ist immer [ERIC_OK](#).

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)

Siehe auch:

- [ERiC-Entwicklerhandbuch.pdf](#), Kap. "Verwendung von EricEntladePlugins()"

[ERICAPI_IMPORT](#) int EricFormatEWaz (const [byteChar](#) * ewAzElster, [EricRueckgabepufferHandle](#) ewAzBescheidPuffer)

Konvertiert ein Einheitswert-Aktenzeichen im ELSTER-Format in ein landesspezifisches Bescheidformat.

Konvertiert ein Einheitswert-Aktenzeichen im ELSTER-Format (z.B. 2831400190001250002) in ein landesspezifisches Einheitswert-Aktenzeichen im Bescheidformat (z.B. 3100190001250002).

Parameter:

in	<i>ewAzElster</i>	Zeiger auf ein Einheitswert-Aktenzeichen im ELSTER-Format (z.B. 2831400190001250002)
out	<i>ewAzBescheidPuffer</i>	Handle auf einen Rückgabepuffer, in den das Einheitswert-Aktenzeichen im Bescheidformat (z.B. 3100190001250002) geschrieben wird. Zur Erzeugung, Verwendung und Freigabe von Rückgabepuffern siehe Dokumentation zu EricRueckgabepufferHandle .

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_EWAZ_UNGUELTIG](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)

[ERICAPI_IMPORT](#) int EricFormatStNr (const [byteChar](#) * eingabeSteuernummer, [EricRueckgabepufferHandle](#) rueckgabePuffer)

Die Steuernummer `eingabeSteuernummer` wird in das Bescheid-Format des jeweiligen Bundeslandes umgewandelt.

Parameter:

in	<i>eingabeSteuernummer</i>	Gültige, zu formatierende Steuernummer im ELSTER-Steuernummernformat.
out	<i>rueckgabePuffer</i>	Handle auf einen Rückgabepuffer, in den die formatierte Steuernummer im Bescheid-Format des jeweiligen Bundeslandes geschrieben wird. Zur Erzeugung, Verwendung und Freigabe von Rückgabepuffern siehe Dokumentation zu EricRueckgabepufferHandle .

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_STEUERNUMMER_UNGUELTIG](#)
- [ERIC_GLOBAL_COMMONDATA_NICHT_VERFUEGBAR](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)

Siehe auch:

- [Pruefung_der_Steuer_und_Steueridentifikatsnummer.pdf](#), siehe [Entwicklerbereich](#) bei [ELSTER](#).

[ERICAPI IMPORT](#) int EricGetAuswahlListen (const char * datenartVersion, const char * feldkennung, [EricRueckgabepufferHandle](#) rueckgabeXmlPuffer)

Die Auswahlliste(n) für `datenartVersion` oder `feldkennung` wird zurückgeliefert.

Anwendungsfälle:

1. Parameter `feldkennung` ist nicht NULL:
Die Funktion liefert die zur `feldkennung` und `datenartVersion` gehörige Auswahlliste.
2. Parameter `feldkennung` ist NULL:
Die Funktion liefert alle zur `datenartVersion` gehörigen Feldkennungen mit hinterlegten Auswahllisten.

Für die Ermittlung der Auswahllisten vieler Feldkennungen wird aus Performanzgründen Anwendungsfall 2 empfohlen. Die Funktion liefert Auswahllisten zu Feldkennungen vom Format "NichtAbgeschlosseneEnumeration" zurück. Diese Auswahllisten werden auch in der Jahres-/Deltadokumentation dokumentiert.

Parameter:

in	<i>datenartVersion</i>	Dieser Parameter darf nicht NULL sein. Die gültigen Datenartversionen sind in der Datenartversionmatrix.xml enthalten.
in	<i>feldkennung</i>	Feldkennung, für welche die Auswahlliste zu ermitteln ist.
out	<i>rueckgabeXmlPuffer</i>	Handle auf einen Rückgabepuffer, in den die angeforderten Auswahlliste(n) als XML-Daten geschrieben werden. Die XML-Daten folgen der XML Schema Definition in Dokumentation\API-Rueckgabe-Schemata\EricGetAuswahlListen.xsd . Zur Erzeugung, Verwendung und Freigabe von Rückgabepuffern siehe EricRueckgabepufferHandle .

Beispiel:

```
<?xml version="1.0" encoding="UTF-8"?>
<EricGetAuswahlListen
xmlns="http://www.elster.de/EricXML/1.0/EricGetAuswahlListen">
  <AuswahlListe>
    <Feldkennung>0104110</Feldkennung>
    <ListenElement>Arbeitslosengeld</ListenElement>
    <ListenElement>Elterngeld</ListenElement>
    <ListenElement>Insolvenzgeld</ListenElement>
    <ListenElement>Krankengeld</ListenElement>
    <ListenElement>Mutterschaftsgeld</ListenElement>
  </AuswahlListe>
</EricGetAuswahlListen>
```

```
</AuswahlListe>  
</EricGetAuswahlListe>
```

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_KEINE_DATEN_VORHANDEN](#)
- [ERIC_GLOBAL_DATENARTVERSION_UNBEKANNT](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)

[ERICAPI_IMPORT](#) int [EricGetErrormessagesFromXMLAnswer](#) (const char * xml, [EricRueckgabepufferHandle](#) transferticketPuffer, [EricRueckgabepufferHandle](#) returncodeTHPuffer, [EricRueckgabepufferHandle](#) fehlertextTHPuffer, [EricRueckgabepufferHandle](#) returncodesUndFehlertexteNDHXmlPuffer)

Aus dem Antwort-XML des Finanzamtsservers wird das Transferticket und Returncodes/Fehlermeldungen zurückgegeben.

Die Funktion liefert bei erfolgreicher Ausführung:

- Das Transferticket aus dem Antwort-XML in dem Parameter `transferticketPuffer`.
- Den Returncode und die Fehlermeldung aus dem Transferheader in den Parametern `returncodeTHPuffer` und `fehlertextTHPuffer`.
- Für jeden Nutzdatenheader dessen Returncode und Fehlermeldung als XML-Daten im Parameter `returncodesUndFehlertexteNDHXmlPuffer` nach XML Schema Definition [Dokumentation\API-Rueckgabe-Schemata\EricGetErrormessagesFromXMLAnswer.xsd](#). Enthält das Antwort-XML keine Nutzdaten, wird kein <Fehler> Element zurückgegeben.

Zur Erzeugung, Verwendung und Freigabe von Rückgabepuffern siehe Dokumentation zu [EricRueckgabepufferHandle](#).

Parameter:

in	<i>xml</i>	Antwort-XML des ELSTER-Servers, das ausgewertet werden soll. Der originale XML-Server-Datenstrom sollte unverändert übergeben werden und darf insbesondere keine Zeilenumbruchzeichen enthalten.
out	<i>transferticketPuffer</i>	Handle auf einen Rückgabepuffer, in den das Transferticket geschrieben wird, siehe EricRueckgabepufferHandle .
out	<i>returncodeTHPuffer</i>	Handle auf einen Rückgabepuffer, in den der Returncode aus dem Transferheader geschrieben wird. Siehe EricRueckgabepufferHandle .
out	<i>fehlertextTHPuffer</i>	Handle auf einen Rückgabepuffer, in den die Fehlermeldung aus dem Transferheader geschrieben wird, siehe EricRueckgabepufferHandle .
out	<i>returncodesUndFehlertexteNDHXmlPuffer</i>	Handle auf einen Rückgabepuffer, in den die Liste der Returncodes nach XML-Schema Dokumentation\API-Rueckgabe-Schemata\EricGetErrormessagesFromXMLAnswer.xsd geschrieben werden, siehe EricRueckgabepufferHandle .

Beispiel:

```
<?xml version="1.0" encoding="UTF-8"?>
<EricGetErrorMessageFromXMLAnswer
xmlns="http://www.elster.de/EricXML/1.0/EricGetErrorMessageFromXMLAnswer">
  <Fehler>
    <Code>1</Code>
    <Meldung>Fehlermeldung 1</Meldung>
  </Fehler>
  <Fehler>
    <Code>2</Code>
    <Meldung>Fehlermeldung 2</Meldung>
  </Fehler>
  (...)
</EricGetErrorMessageFromXMLAnswer>
```

Rückgabe:

- [ERIC_OK](#)
- [ERIC_IO_PARSE_FEHLER](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_PUFFER_ZUGRIFFSKONFLIKT](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)

Zu beachten:

Diese Funktion kann nicht dafür verwendet werden, die Antwort im Datenteil aus einer dekodierten Serverantwort für Lohnsteuerbescheinigungen auszuwerten.

Siehe auch:

- XML-Schema des Transferheaders:
[Dokumentation\Schnittstellenbeschreibungen\ElsterBasisSchema\Schema\th000011_extern.xsd](#)
- XML-Schema des Nutzdatenheaders:
[Dokumentation\Schnittstellenbeschreibungen\ElsterBasisSchema\Schema\ndh000011.xsd](#)
- [ERIC-Entwicklerhandbuch.pdf](#), Kap. "Schnittstellenbeschreibungen", Tabelle "Ergänzende Softwarepakete und Dateien – Schnittstellenbeschreibungen"

[ERICAPI_IMPORT](#) int EricGetHandleToCertificate ([EricZertifikatHandle](#) * hToken, [uint32_t](#) * iInfoPinSupport, const [byteChar](#) * pathToKeystore)

Für das übergebene Zertifikat in `pathToKeystore` wird das Handle `hToken` und die unterstützten PIN-Werte `iInfoPinSupport` zurückgeliefert.

Die ERiC API benötigt Zertifikat-Handles typischerweise bei kryptografischen Operationen.

Zertifikat-Handles sollten möglichst frühzeitig, d.h. wenn sie nicht mehr benötigt werden, mit [EricCloseHandleToCertificate\(\)](#) freigegeben werden, spätestens jedoch zum Programmende bzw. vor dem Entladen der ericapi Bibliothek.

Parameter:

out	<i>hToken</i>	Handle zu einem der folgenden Zertifikate: <ul style="list-style-type: none"> ◦ Portalzertifikat ◦ clientseitig erzeugtes Zertifikat ◦ Ad Hoc-Zertifikat für den neuen Personalausweis
-----	---------------	---

out	<i>iInfoPinSupport</i>	<p>Wird in <code>iInfoPinSupport</code> ein Zeiger ungleich NULL übergeben und die Funktion mit ERIC_OK beendet, dann enthält <code>iInfoPinSupport</code> einen vorzeichenlosen Integer-Wert.</p> <p>In diesem Wert ist kodiert abgelegt, ob eine PIN-Eingabe erforderlich ist und welche PIN-Statusinformationen unterstützt werden.</p> <p>Die kodierten Werte (nachfolgend in hexadezimaler Form angegeben) können durch ein binäres ODER kombiniert werden und bedeuten im Einzelnen:</p> <ul style="list-style-type: none"> ◦ <code>0x00</code>: Keine PIN-Angabe erforderlich, kein PIN-Status unterstützt. ◦ <code>0x01</code>: PIN-Angabe für Signatur erforderlich. ◦ <code>0x02</code>: PIN-Angabe für Entschlüsselung erforderlich. ◦ <code>0x04</code>: PIN-Angabe für Verschlüsselung des Zertifikats erforderlich. ◦ <code>0x08</code>: reserviert (wird derzeit nicht verwendet) ◦ <code>0x10</code>: PIN-Status "Pin Ok" wird unterstützt. ◦ <code>0x20</code>: PIN-Status "Der letzte Versuch der Pin-Eingabe schlug fehl" wird unterstützt. ◦ <code>0x40</code>: PIN-Status "Beim nächsten fehlerhaften Versuch wird die Pin gesperrt" wird unterstützt. ◦ <code>0x80</code>: PIN-Status "Pin ist gesperrt" wird unterstützt. <p>Falls vom Aufrufer NULL übergeben wird, gibt die Funktion nichts zurück.</p>
-----	------------------------	---

in	<i>pathToKeystore</i>	<p>Folgende Zertifikatstypen werden unterstützt:</p> <ol style="list-style-type: none"> 1. Clientseitig erzeugtes Zertifikat: Pfad zum Verzeichnis, in dem sich die Zertifikats-Datei (.cer) und die Datei mit dem privaten Schlüssel (.p12) befinden. Diese Kryptomittel wurden mit EricCreateKey() erzeugt. Der Pfad zum Verzeichnis ist bei clientseitig erzeugten Zertifikaten relativ zum aktuellen Arbeitsverzeichnis oder absolut anzugeben. 2. Software-Portalzertifikat: Pfad zur Software-Zertifikatsdatei (i.d.R. mit der Endung .pfx). Der Pfad zur Datei ist bei Software-Zertifikaten relativ zum aktuellen Arbeitsverzeichnis oder absolut anzugeben. 3. Sicherheitsstick: Pfad zur Treiberdatei, siehe (1). Bitte beachten, dass der Treiber betriebssystemabhängig sein kann. Weitere Informationen in der Anleitung zum Sicherheitsstick oder unter https://www.sicherheitsstick.de. 4. Signaturkarte: Pfad zur Treiberdatei, welcher einen Zugriff auf die Signaturkarte ermöglicht, siehe (1). Weitere Informationen in der Anleitung zur Signaturkarte. 5. Neuer Personalausweis (nPA): URL des eID-Clients wie zum Beispiel der AusweisApp 2 In den meisten Fällen lautet diese URL: http://127.0.0.1:24727/eID-Client Optional kann auf die folgende Weise noch ein Testmerker angehängt werden: http://127.0.0.1:24727/eID-Client?testmerker=520000000. Zu den verfügbaren Testmerkern siehe ERiC-Entwicklerhandbuch.pdf, Kap. "Test Unterstützung bei der ERiC-Anbindung". <p>Wichtig: Das Ad Hoc-Zertifikat, das in diesem Fall für den neuen Personalausweis erzeugt wird, ist nur 24 Stunden gültig.</p>
----	-----------------------	---

(1) Bei Sicherheitssticks und Signaturkarten ist bei der Angabe des Treibers der Suchmechanismus nach dynamischen Modulen des jeweiligen Betriebssystems

zu berücksichtigen. Weitere Informationen sind z.B. unter Windows der Dokumentation der `LoadLibrary()` oder unter Linux und macOS der Dokumentation der `dlopen()` zu entnehmen.

Pfade müssen auf Windows in der für Datei-Funktionen benutzten ANSI-Codepage, auf Linux, AIX und Linux Power in der für das Dateisystem benutzten Locale und auf macOS in der "decomposed form" von UTF-8 übergeben werden. Bitte weitere Betriebssystemspezifika bzgl. nicht erlaubter Zeichen in Pfaden und Pfadtrennzeichen beachten.

Für Details zu Pfaden im ERiC siehe [ERiC-Entwicklerhandbuch.pdf](#), Kapitel "Übergabe von Pfaden an ERiC API-Funktionen".

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)
- [ERIC_CRYPT_NICHT_UNTERSTUETZTES_PSE_FORMAT](#)
- [ERIC_CRYPT_E_MAX_SESSION](#)
- [ERIC_CRYPT_E_PSE_PATH](#)
- [ERIC_CRYPT_E_BUSY](#)
- [ERIC_CRYPT_E_P11_SLOT_EMPTY](#)
- [ERIC_CRYPT_E_NO_SIG_ENC_KEY](#)
- [ERIC_CRYPT_E_LOAD_DLL](#)
- [ERIC_CRYPT_E_NO_SERVICE](#)
- [ERIC_CRYPT_E_ESICL_EXCEPTION](#)

Zu beachten:

Die folgenden Rückgabewerte gelten nur bei Verwendung des neuen Personalausweises.

Rückgabe:

- [ERIC_TRANSFER_EID_CLIENTFEHLER](#)
- [ERIC_TRANSFER_EID_FEHLENDEFELDER](#)
- [ERIC_TRANSFER_EID_IDENTIFIKATIONABGEBROCHEN](#)
- [ERIC_TRANSFER_EID_NPABLOCKIERT](#)
- [ERIC_TRANSFER_EID_IDNRNICHTEINDEUTIG](#)
- [ERIC_TRANSFER_EID_KEINCLIENT](#)
- [ERIC_TRANSFER_EID_KEINKONTO](#)
- [ERIC_TRANSFER_EID_SERVERFEHLER](#)

- [ERIC_TRANSFER_ERR_CONNECTSERVER](#)
- [ERIC_TRANSFER_ERR_NORESPONSE](#)
- [ERIC_TRANSFER_ERR_PROXYAUTH](#)
- [ERIC_TRANSFER_ERR_PROXYCONNECT](#)
- [ERIC_TRANSFER_ERR_SEND](#)
- [ERIC_TRANSFER_ERR_SEND_INIT](#)
- [ERIC_TRANSFER_ERR_TIMEOUT](#)

Siehe auch:

- [EricCloseHandleToCertificate\(\)](#)
- [EricGetPinStatus\(\)](#)

ERICAPI_IMPORT int EricGetPinStatus ([EricZertifikatHandle](#) hToken, [uint32_t](#) *pinStatus, [uint32_t](#) keyType)

Der PIN-Status wird für ein passwortgeschütztes Kryptomittel abgefragt und in `pinStatus` zurückgegeben.

Der PIN-Status wird für einen passwortgeschützten Bereich ermittelt, der durch das übergebene Zertifikat-Handle im Parameter `hToken` referenziert wird. Da bei Sicherheitssticks und Signaturkarten durch ein einziges Zertifikat-Handle zwei Schlüsselpaare referenziert werden können (eines für die Signatur und eines für die Verschlüsselung von Daten), muss grundsätzlich der Parameter `keyType` gesetzt werden.

Mit dem Rückgabewert der Funktion kann der Endanwender rechtzeitig informiert werden, falls bei einer weiteren falschen PIN-Eingabe das Kryptomittel gesperrt wird. Im Fehlerfall ist `pinStatus` nicht definiert.

Der Karten- bzw. Stickhersteller ist verantwortlich, dass seine Implementierung den korrekten PIN-Status zurückgibt, siehe auch Tabelle "PIN-Statusabfrage für POZ" im Unterkap. "Das Portalzertifikat (POZ)" im Dokument [ERIC-Entwicklerhandbuch.pdf](#).

Parameter:

in	<i>hToken</i>	Zertifikat-Handle, für dessen passwortgeschützten Bereich der PIN-Status ermittelt werden soll. Wird von der Funktion EricGetHandleToCertificate() zurückgeliefert.
out	<i>pinStatus</i>	Mögliche Rückgabewerte: <ul style="list-style-type: none"> ◦ 0: StatusPinOk: Kein Fehlversuch oder keine Informationen verfügbar ◦ 1: StatusPinLocked: PIN gesperrt ◦ 2: StatusPreviousPinError: Die letzte PIN-Eingabe war fehlerhaft ◦ 3: StatusLockedIfPinError: Beim nächsten fehlerhaften Versuch wird die PIN gesperrt
in	<i>keyType</i>	Mögliche Eingabewerte: <ul style="list-style-type: none"> ◦ 0: eSignatureKey: Schlüssel für die Signatur von Daten ◦ 1: eEncryptionKey: Schlüssel für die Verschlüsselung von Daten

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- weitere, siehe [eric_fehlercodes.h](#)

Siehe auch:

- [EricGetHandleToCertificate\(\)](#)
- [ERiC-Entwicklerhandbuch.pdf](#), Kap. "Zertifikate und Authentifizierungsverfahren"

ERICAPI_IMPORT int EricGetPublicKey (const eric_verschluesselungs_parameter_t * cryptoParameter, EricRueckgabepufferHandle rueckgabePuffer)

Es wird der öffentliche Schlüssel als base64-kodierte Zeichenkette für das übergebene Zertifikat-Handle in `cryptoParameter` zurückgeliefert. Von den beiden öffentlichen Schlüsseln, die in ELSTER-Zertifikaten enthalten sind, wird der Verschlüsselungsschlüssel (Encryption Key) zurückgegeben.

Parameter:

in	<i>cryptoParameter</i>	Die Struktur enthält das Zertifikat-Handle und die PIN. Falls der Zugriff auf den öffentlichen Schlüssel keine PIN erfordert, ist PIN=NULL anzugeben.
out	<i>rueckgabePuffer</i>	Handle auf den Rückgabepuffer. Bei Erfolg enthält der Rückgabepuffer den öffentlichen Schlüssel als base64-kodierte Zeichenkette. Zur Erzeugung, Verwendung und Freigabe von Rückgabepuffern siehe Dokumentation zu EricRueckgabepufferHandle .

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_CRYPT_E_INVALID_HANDLE](#)
- [ERIC_CRYPT_E_P12_ENC_KEY](#)
- [ERIC_CRYPT_E_PIN_WRONG](#)
- [ERIC_CRYPT_E_PIN_LOCKED](#)
- weitere, siehe [eric_fehlercodes.h](#)

ERICAPI_IMPORT int EricHoleFehlerText (int fehlerkode, EricRueckgabepufferHandle rueckgabePuffer)

Es wird die Klartextfehlermeldung zu dem `fehlerkode` ermittelt.

Die Funktion liefert die Klartextfehlermeldung zu einem ERiC Fehlercode - definiert in [eric_fehlercodes.h](#)

Parameter:

in	<i>fehlerkode</i>	Eingabe-Fehlercode, definiert in eric_fehlercodes.h .
out	<i>rueckgabePuffer</i>	Handle auf einen Rückgabepuffer, in den die Klartextfehlermeldung geschrieben wird. Zur Erzeugung, Verwendung und Freigabe von Rückgabepuffern siehe Dokumentation zu EricRueckgabepufferHandle . Die Klartextfehlermeldung ist gemäß UTF-8 kodiert.

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_FEHLERMELDUNG_NICHT_VORHANDEN](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)

ERICAPI_IMPORT int EricHoleFinanzaemter (const [byteChar](#) * finanzamtLandNummer, [EricRueckgabepufferHandle](#) rueckgabeXmlPuffer)

Es wird die Finanzamtliste für eine bestimmte finanzamtLandNummer zurückgegeben.

Parameter:

in	<i>finanzamtLandNummer</i>	Die Finanzamtlandnummer besteht aus den ersten zwei Stellen der Bundesfinanzamtsnummer. Eine Liste aller Finanzamtlandnummern wird von EricHoleFinanzamtLandNummern() zurückgegeben.
out	<i>rueckgabeXmlPuffer</i>	Handle auf einen Rückgabepuffer, in den die Ergebnis XML-Daten geschrieben werden. Die XML-Daten folgen der XML Schema Definition Dokumentation\API-Rueckgabe-Schemata\EricHoleFinanzaemter.xsd . Zur Erzeugung, Verwendung und Freigabe von Rückgabepuffern siehe Dokumentation zu EricRueckgabepufferHandle .

Beispiel:

```
<?xml version="1.0" encoding="UTF-8"?>
<EricHoleFinanzaemter
xmlns="http://www.elster.de/EricXML/1.0/EricHoleFinanzaemter">
  <Finanzamt>
    <BuFaNummer>2801</BuFaNummer>
    <Name>Finanzamt Offenburg Außenstelle Achern</Name>
  </Finanzamt>
  <Finanzamt>
    <BuFaNummer>2804</BuFaNummer>
    <Name>Finanzamt Villingen-Schwenningen Außenstelle Donaueschingen</Name>
  </Finanzamt>
  (...)
</EricHoleFinanzaemter>
```

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_UTI_COUNTRY_NOT_SUPPORTED](#)
- [ERIC_GLOBAL_COMMONDATA_NICHT_VERFUEGBAR](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)

ERICAPI IMPORT int EricHoleFinanzamtLandNummern (EricRueckgabepufferHandle rueckgabeXmlPuffer)

Die Liste aller Finanzamtlandnummern wird zurückgegeben.

Parameter:

out	<i>rueckgabeXmlPuffer</i>	Handle auf einen Rückgabepuffer, in den die Ergebnis XML-Daten geschrieben werden. Die XML-Daten folgen der XML Schema Definition Dokumentation\API-Rueckgabe-Schemata\EricHoleFinanzamtLandNummern.xsd . Zur Erzeugung, Verwendung und Freigabe von Rückgabepuffern siehe Dokumentation zu EricRueckgabepufferHandle .
-----	---------------------------	--

Beispiel:

```
<?xml version="1.0" encoding="UTF-8"?>
<EricHoleFinanzamtLandNummern
xmlns="http://www.elster.de/EricXML/1.0/EricHoleFinanzamtLandNummern">
  <FinanzamtLand>
    <FinanzamtLandNummer>28</FinanzamtLandNummer>
    <Name>Baden-Württemberg</Name>
  </FinanzamtLand>
  <FinanzamtLand>
    <FinanzamtLandNummer>91</FinanzamtLandNummer>
    <Name>Bayern (Zuständigkeit LfSt - München)</Name>
  </FinanzamtLand>
  (...)
</EricHoleFinanzamtLandNummern>
```

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_COMMONDATA_NICHT_VERFUEGBAR](#)
- [ERIC_GLOBAL_NICHT_GENUEGENDE_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)

[ERICAPI_IMPORT](#) int [EricHoleFinanzamtsdaten](#) (const [byteChar](#) [bufaNr](#)[5], [EricRueckgabepufferHandle](#) [rueckgabeXmlPuffer](#))

Die `finanzamtsdaten` werden für eine Bundesfinanzamtsnummer zurückgegeben.

Die Bundesfinanzamtsnummer kann über die Kombination der Funktionen [EricHoleFinanzamtLandNummern\(\)](#) und [EricHoleFinanzaemter\(\)](#) ermittelt werden.

Parameter:

in	<i>bufaNr</i>	Übergabe der 4-stelligen Bundesfinanzamtsnummer.
out	<i>rueckgabeXmlPuffer</i>	Handle auf einen Rückgabepuffer, in den die Ergebnis XML-Daten geschrieben werden. Die XML-Daten folgen der XML Schema Definition Dokumentation\API-Rueckgabe-Schemata\EricHoleFinanzamtsdaten.xsd . Zur Erzeugung, Verwendung und Freigabe von Rückgabepuffern siehe Dokumentation zu EricRueckgabepufferHandle .

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#): Parameter `bufaNr` ist NULL.
- [ERIC_GLOBAL_PRUEF_FEHLER](#): Die übergebene Bundesfinanzamtsnummer ist keine Ganzzahl.
- [ERIC_GLOBAL_KEINE_DATEN_VORHANDEN](#): Immer bei Testfinanzämtern.
- [ERIC_GLOBAL_COMMONDATA_NICHT_VERFUEGBAR](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)

Siehe auch:

- [EricHoleFinanzamtLandNummern\(\)](#)
- [EricHoleFinanzaemter\(\)](#)

ERICAPI IMPORT int EricHoleTestfinanzaemter (EricRueckgabepufferHandle rueckgabeXmlPuffer)

Die Testfinanzamtliste wird in `rueckgabeXmlPuffer` zurückgegeben.

Parameter:

out	<i>rueckgabeXmlPuffer</i>	Handle auf einen Rückgabepuffer, in den die Ergebnis XML-Daten geschrieben werden. Die XML-Daten folgen der XML Schema Definition Dokumentation\API-Rueckgabe-Schemata\EricHoleTestFinanzaemter.xsd . Zur Erzeugung, Verwendung und Freigabe von Rückgabepuffern siehe Dokumentation zu EricRueckgabepufferHandle .
-----	---------------------------	--

Beispiel:

```
<?xml version="1.0" encoding="UTF-8"?>
<EricHoleTestFinanzaemter
xmlns="http://www.elster.de/EricXML/1.0/EricHoleTestFinanzaemter">
  <Finanzamt>
    <BuFaNummer>1096</BuFaNummer>
    <Name>Testfinanzamt Saarland</Name>
  </Finanzamt>
  <Finanzamt>
    <BuFaNummer>1097</BuFaNummer>
    <Name>Finanzschule (Edenkoben)</Name>
  </Finanzamt>
  (...)
</EricHoleTestFinanzaemter>
```

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_COMMONDATA_NICHT_VERFUEGBAR](#)
- [ERIC_GLOBAL_NICHT_GENUEGENDE_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)

ERICAPI_IMPORT int **EricHoleZertifikatEigenschaften** ([EricZertifikatHandle](#) hToken, const [byteChar](#) * pin, [EricRueckgabepufferHandle](#) rueckgabeXmlPuffer)

Die Eigenschaften des übergebenen Zertifikats werden im `rueckgabeXmlPuffer` zurückgegeben.

Parameter:

in	<i>hToken</i>	Handle des Zertifikats, dessen Eigenschaften geholt werden sollen. Wird von der Funktion EricGetHandleToCertificate() zurückgeliefert.
in	<i>pin</i>	PIN zum Öffnen des Zertifikats. Wird bei Software-Portalzertifikaten benötigt.
out	<i>rueckgabeXmlPuffer</i>	Handle auf einen Rückgabepuffer, in den die Zertifikateigenschaften im XML-Format geschrieben werden. Das Format ist im XML Schema Dokumentation\API-Rueckgabe-Schemata\EricHoleZertifikatEigenschaften.xsd definiert. Zur Erzeugung, Verwendung und Freigabe von Rückgabepuffern siehe Dokumentation zu EricRueckgabepufferHandle .

Zu beachten:

Bei einem ELSTER-Softwarezertifikat (.pfx) steht im Common Name (CN) die ID des ELSTER-Kontos, für das das Zertifikat ausgestellt wurde. Die Konto-ID kann beispielsweise dafür genutzt werden, bei einer Zertifikatsverlängerung das verlängerte Zertifikat dem alten Zertifikat zuzuordnen.

Beispiel:

```
<EricHoleZertifikatEigenschaften
xmlns="http://www.elster.de/EricXML/2.0/EricHoleZertifikatEigenschaften">
  <Signaturzertifikateigenschaften>
    <AusgestelltAm>220817152116Z</AusgestelltAm>
    <GueltingBis>230817152116Z</GueltingBis>

<Signaturalgorithmus>sha1WithRSAEncryption(1.2.840.113549.1.1.5)</Signaturalgorithmus>

    <PublicKeyMD5>6b8b191936677957fe74103198e77f4e</PublicKeyMD5>
    <PublicKeySHA1>884b0dfe2e10221a2aedd28c986cf34db0f1d932</PublicKeySHA1>
    <PublicKeyBitLength>2048</PublicKeyBitLength>
    <Issuer>
      <Info><Name>CN</Name><Wert>ElsterSoftCA</Wert></Info>
      <Info><Name>OU</Name><Wert>CA</Wert></Info>
      (...)
    </Issuer>
    <Subjekt>
      <Info><Name>CN</Name><Wert>1000872896</Wert></Info>
```

```
</Subjekt>
<Identifikationsmerkmaltyp>Steuernummer</Identifikationsmerkmaltyp>
<Registrierertyp>Person</Registrierertyp>
<Verifikationsart>Postweg</Verifikationsart>
<TokenTyp>Software</TokenTyp>
<Testzertifikat>true</Testzertifikat>
</Signaturzertifikateigenschaften>
<Verschluesselungszertifikateigenschaften>
  (...)
</Verschluesselungszertifikateigenschaften>
</EricHoleZertifikatEigenschaften>
```

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)
- ERIC_CRYPT_E_*: Ein Zertifikatsfehler aus dem Statuscodebereich von [ERIC_CRYPT_E_INVALID_HANDLE](#) = 610201101 bis 610201212

Siehe auch:

- [ERiC-Entwicklerhandbuch.pdf](#), Kap. "Verwendung von EricHoleZertifikatEigenschaften()"
- [Dokumentation\API-Rueckgabe-Schemata\EricHoleZertifikatEigenschaften.xsd](#)

ERICAPI_IMPORT int EricHoleZertifikatFingerabdruck (const **eric_verschluesselungs_parameter_t** * cryptoParameter, **EricRueckgabepufferHandle** fingerabdruckPuffer, **EricRueckgabepufferHandle** signaturPuffer)

Der Fingerabdruck und dessen Signatur wird für das übergebene Zertifikat zurückgegeben.

Zur Erzeugung, Verwendung und Freigabe von Rückgabepuffern siehe Dokumentation zu [EricRueckgabepufferHandle](#).

Parameter:

in	<i>cryptoParameter</i>	Zertifikatsdaten, siehe eric_verschluesselungs_parameter_t . Das in der übergebenen Struktur referenzierte Zertifikat muss ein clientseitig erzeugtes Zertifikat (CEZ) sein.
out	<i>fingerabdruckPuffer</i>	Handle auf einen Rückgabepuffer, in den der Fingerabdruck geschrieben wird, siehe EricRueckgabepufferHandle .
out	<i>signaturPuffer</i>	Handle auf einen Rückgabepuffer, in den die Signatur des Fingerabdrucks geschrieben wird, siehe EricRueckgabepufferHandle .

Zu beachten:

Die Erzeugung eines Fingerabdrucks mit dieser Funktion ist nur in Zusammenhang mit clientseitig erzeugten Zertifikaten definiert.

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_PUFFER_ZUGRIFFSKONFLIKT](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)
- [ERIC_CRYPT_E_P12_READ](#)
- [ERIC_CRYPT_E_P12_DECODE](#)
- [ERIC_CRYPT_E_PIN_WRONG](#)
- [ERIC_CRYPT_E_P12_SIG_KEY](#)
- [ERIC_CRYPT_E_P12_ENC_KEY](#)
- [ERIC_CRYPT_ZERTIFIKAT](#)
- [ERIC_CRYPT_EIDKARTE_NICHT_UNTERSTUETZT](#)
- [ERIC_CRYPT_SIGNATUR](#)
- [ERIC_CRYPT_CORRUPTED](#)

[ERICAPI_IMPORT](#) int EricInitialisiere (const [byteChar](#) * pluginPfad, const [byteChar](#) * logPfad)

Initialisiert den Singlethreading-ERiC.

Vor der Verwendung der Singlethreading-API muss [EricInitialisiere\(\)](#) aufgerufen werden.

Mehrfache Aufrufe dieser Funktion, ohne das zwischendurch [EricBeende\(\)](#) aufgerufen worden ist, führen dazu, dass der Fehlercode [ERIC_GLOBAL_MEHRFACHE_INITIALISIERUNG](#) zurückgegeben wird. Der zuvor initialisierte Singlethreading-ERiC bleibt davon aber unberührt und ist weiterhin in einem gültigen Zustand.

Parameter:

in	<i>pluginPfad</i>	Pfad, in dem die Plugins rekursiv gesucht werden. Ist der Zeiger gleich NULL, wird der Pfad zur Bibliothek ericapi verwendet.
in	<i>logPfad</i>	Optionaler Pfad zur Log-Datei eric.log. Ist der Wert gleich NULL, wird das betriebssystemspezifische Verzeichnis für temporäre Dateien verwendet.

Zu beachten:

Kann kein eric.log angelegt werden, wird eine entsprechende Fehlermeldung auf die Konsole (stderr) geschrieben und an den Windows-Ereignisdienst bzw. den syslogd-Dienst (Linux, AIX, macOS) geschickt.

Für Linux, AIX und macOS ist zu beachten, dass der syslogd-Dienst gegebenenfalls erst noch zu aktivieren und für die Protokollierung von Meldungen der Facility "User" zu konfigurieren ist.

Suchkriterien für ERiC-Meldungen in der Windows-Ereignisansicht sind "ERiC (Elster Rich Client)" als Quelle und "Anwendung" als Protokoll.

Suchkriterien für ERiC-Meldungen in den Systemlogdateien unter Linux, AIX und macOS sind die Facility "User" und der Ident "ERiC (Elster Rich Client)".

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_MEHRFACHE_INITIALISIERUNG](#)
- [ERIC_GLOBAL_FEHLER_INITIALISIERUNG](#)
- [ERIC_GLOBAL_LOG_EXCEPTION](#)

Siehe auch:

- [EricBeende\(\)](#)

[ERICAPI_IMPORT](#) int EricMakeElsterEWAZ (const [byteChar](#) * ewAzBescheid, const [byteChar](#) * landeskuerzel, [EricRueckgabepufferHandle](#) ewAzElsterPuffer)

Konvertiert ein Einheitswert-Aktenzeichen in das ELSTER-Format.

Konvertiert ein gültiges Einheitswert-Aktenzeichen in einem landesspezifischen Bescheidformat (z.B. 208/035-3-03889.3) unter Angabe des Landeskürzels in ein Einheitswert-Aktenzeichen im ELSTER-Format (z.B. 520840353038893).

Parameter:

in	<i>ewAzBescheid</i>	Zeiger auf das Einheitswert-Aktenzeichen in einem landesspezifischen Bescheidformat.
in	<i>landeskuerzel</i>	Zeiger auf das Landeskürzel (zum Beispiel BY für Bayern)
out	<i>ewAzElsterPuffer</i>	Handle auf einen Rückgabepuffer, in den das erzeugte Einheitswert-Aktenzeichen im ELSTER-Format geschrieben wird.

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_EWAZ_UNGUELTIG](#)
- [ERIC_GLOBAL_EWAZ_LANDESKUERZEL_UNBEKANNT](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_UNGUELTIGER_PARAMETER](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)

Siehe auch:

- Landeskürzel siehe ISO-3166-2

[ERICAPI_IMPORT](#) int EricMakeElsterStnr (const [byteChar](#) * steuernrBescheid, const [byteChar](#) landesnr[2+1], const [byteChar](#) bundesfinanzamtsnr[4+1], [EricRueckgabepufferHandle](#) steuernrPuffer)

Es wird eine Steuernummer im ELSTER-Steuernummerformat erzeugt.

Die Funktion erzeugt aus einer angegebenen Steuernummer im Format des Steuerbescheides eine 13-stellige Steuernummer im ELSTER-Steuernummerformat.

Die sich ergebende 13-stellige Steuernummer im ELSTER-Steuernummerformat wird von der Funktion [EricMakeElsterStnr\(\)](#) auch auf Gültigkeit geprüft.

Einer der beiden Parameter `landesnr` oder `bundesfinanzamtsnr` muss korrekt angegeben werden. Der jeweils andere Parameter darf NULL oder leer sein. Bei bayerischen und berliner Steuernummern im Format BBB/UUUUP ist die Angabe der Bundesfinanzamtsnummer zwingend erforderlich.

Parameter:

in	<i>steuernrBescheid</i>	Format der Steuernummer wie auch auf amtlichen Schreiben angegeben.
in	<i>landesnr</i>	2-stellige Landesnummer (entspricht den ersten zwei Stellen der Bundesfinanzamtsnummer).
in	<i>bundesfinanzamtsnr</i>	4-stellige Bundesfinanzamtsnummer.
out	<i>steuernrPuffer</i>	Handle auf einen Rückgabepuffer, in den die Steuernummer im ELSTER-Steuernummerformat geschrieben wird. Zur Erzeugung, Verwendung und Freigabe von Rückgabepuffern siehe Dokumentation zu EricRueckgabepufferHandle .

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_STEUERNUMMER_UNGUELTIG](#)
- [ERIC_GLOBAL_LANDESNUMMER_UNBEKANNT](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_UNGUELTIGER_PARAMETER](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)

ERICAPI_IMPORT int EricPruefeBIC (const byteChar * bic)

Die `bic` wird auf Gültigkeit überprüft.

Die Prüfung erfolgt in zwei Schritten:

1. Formale Prüfung auf gültige Zeichen und richtige Länge.
2. Prüfung, ob das Länderkennzeichen für BIC gültig ist.

Falls die BIC ungültig ist liefert die Funktion [EricHoleFehlerText\(\)](#) den zugehörigen Fehlertext.

Parameter:

in	<i>bic</i>	Zeiger auf eine NULL-terminierte Zeichenkette.
----	------------	--

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_BIC_FORMALER_FEHLER](#): Ungültige Zeichen, falsche Länge.
- [ERIC_GLOBAL_BIC_LAENDERCODE_FEHLER](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#): Parameter `bic` ist NULL.
- [ERIC_GLOBAL_COMMONDATA_NICHT_VERFUEGBAR](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)

Siehe auch:

- [ERiC-Entwicklerhandbuch.pdf](#), Kap. "BIC ISO-Ländercodes"
- [ERiC-Entwicklerhandbuch.pdf](#), Kap. "BIC-Prüfung"

ERICAPI_IMPORT int EricPruefeBuFaNummer (const byteChar * steuernummer)

Die Bundesfinanzanzamtsnummer wird überprüft.

Wird eine 13-stellige Steuernummer im ELSTER-Steuernummernformat angegeben, so wird nur die Bundesfinanzanzamtsnummer (= die ersten 4 Stellen der 13-stelligen Steuernummer) geprüft.

Eine Prüfung der Steuernummer selbst findet nicht statt (hierfür [EricPruefeSteuernummer\(\)](#) verwenden).

Parameter:

in	<i>steuernummer</i>	13-stellige Steuernummer im ELSTER Steuernummernformat bzw. 4-stellige Bundesfinanzanzamtsnummer.
----	---------------------	---

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_BUFANR_UNBEKANNT](#): Die Bundesfinanzanzamtsnummer ist unbekannt oder ungültig.
- [ERIC_GLOBAL_NULL_PARAMETER](#): Es wurde keine Bundesfinanzanzamtsnummer übergeben (Parameter ist NULL).
- [ERIC_GLOBAL_COMMONDATA_NICHT_VERFUEGBAR](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)

Siehe auch:

- [EricPruefeSteuernummer\(\)](#)
- Pruefung_der_Steuer_und_Steueridentifikatsnummer.pdf, siehe [Entwicklerbereich](#) bei [ELSTER](#).

ERICAPI_IMPORT int EricPruefeEWaz (const byteChar * einheitswertAz)

Überprüft ein `Einheitswert-Aktenzeichen` im ELSTER-Format auf Gültigkeit.

Parameter:

in	<i>einheitswertAz</i>	Zeiger auf ein Einheitswert-Aktenzeichen im ELSTER-Format
----	-----------------------	---

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_EWAZ_UNGUELTIG](#)
- [ERIC_GLOBAL_COMMONDATA_NICHT_VERFUEGBAR](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)

ERICAPI_IMPORT int EricPruefelIBAN (const byteChar * iban)

Die `iban` wird auf Gültigkeit überprüft.

Die Prüfung erfolgt in vier Schritten:

1. Formale Prüfung auf gültige Zeichen und richtige Länge.
2. Prüfung, ob das Länderkennzeichen für IBAN gültig ist.
3. Prüfung, ob das länderspezifische Format gültig ist.
4. Prüfung, ob die Prüfziffer der IBAN gültig ist.

Falls die IBAN ungültig ist, liefert die Funktion EricHoleFehlerText() den zugehörigen Fehlertext.

Parameter:

in	<i>iban</i>	Zeiger auf eine NULL-terminierte Zeichenkette.
----	-------------	--

Rückgabe:

- ERIC_OK
- ERIC_GLOBAL_IBAN_FORMALER_FEHLER: Ungültige Zeichen, falsche Länge.
- ERIC_GLOBAL_IBAN_LAENDERCODE_FEHLER
- ERIC_GLOBAL_IBAN_LANDESFORMAT_FEHLER
- ERIC_GLOBAL_IBAN_PRUEFZIFFER_FEHLER
- ERIC_GLOBAL_NULL_PARAMETER: Parameter `iban` ist NULL.
- ERIC_GLOBAL_COMMONDATA_NICHT_VERFUEGBAR
- ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER
- ERIC_GLOBAL_UNKNOWN

Siehe auch:

- ERiC-Entwicklerhandbuch.pdf, Kap. "IBAN - länderspezifische Formate"
- ERiC-Entwicklerhandbuch.pdf, Kap. "IBAN-Prüfung"

[ERICAPI_IMPORT](#) int EricPruefeldentifikationsMerkmal (const [byteChar](#) * steuerId)

Die `steuerId` wird auf Gültigkeit überprüft.

Parameter:

in	<i>steuerId</i>	Steuer-Identifikationsnummer (IdNr)
----	-----------------	-------------------------------------

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_IDNUMMER_UNGUELTIG](#)
- [ERIC_GLOBAL_COMMONDATA_NICHT_VERFUEGBAR](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)

Siehe auch:

- [EricPruefeSteuernummer\(\)](#)
- [ERiC-Entwicklerhandbuch.pdf](#), Kap. "Prüfung der Steueridentifikationsnummer (IdNr)"
- [ERiC-Entwicklerhandbuch.pdf](#), Kap. "Test-Steueridentifikationsnummer"

ERICAPI_IMPORT int EricPruefeSteuernummer (const byteChar * steuernummer)

Die `steuernummer` wird einschließlich Bundesfinanzamtsnummer auf formale Richtigkeit geprüft.

Zur Prüfung der Bundesfinanzamtsnummer wird [EricPruefeBuFaNummer\(\)](#) verwendet.

Zur Erzeugung, Verwendung und Freigabe von Rückgabepuffern siehe Dokumentation zu [EricRueckgabepufferHandle](#).

Parameter:

in	<i>steuernummer</i>	NULL-terminierte 13-stellige Steuernummer im ELSTER-Steuernummernformat.
----	---------------------	--

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_STEUERNUMMER_UNGUELTIG](#)
- [ERIC_GLOBAL_COMMONDATA_NICHT_VERFUEGBAR](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)

Siehe auch:

- [EricPruefeBuFaNummer\(\)](#)
- [Pruefung_der_Steuer_und_Steueridentifikatsnummer.pdf](#), siehe [Entwicklerbereich](#) bei [ELSTER](#).

ERICAPI_IMPORT int EricPruefeWldNr (const byteChar * wldNr)

Die Wirtschafts-Identifikationsnummer (W-IdNr .) wird auf formale Gültigkeit überprüft.

Parameter:

in	<i>wldNr</i>	NULL-terminierte Wirtschafts-Identifikationsnummer mit oder ohne Unterscheidungsmerkmal.
----	--------------	--

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_IDNUMMER_UNGUELTIG](#)
- [ERIC_GLOBAL_COMMONDATA_NICHT_VERFUEGBAR](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)

Siehe auch:

- [ERiC-Entwicklerhandbuch.pdf](#), Kap. "Prüfung der Wirtschafts-Identifikationsnummer (W-IdNr.)"

ERICAPI_IMPORT int EricPruefeZertifikatPin (const [byteChar](#) * pathToKeystore, const [byteChar](#) * pin, [uint32_t](#) keyType)

Prüft, ob die `pin` zum Zertifikat `pathToKeystore` passt. Nicht anwendbar auf Ad Hoc-Zertifikate (AHZ), die für einen neuen Personalausweis (nPA) ausgestellt sind.

Parameter:

in	<i>pathToKeystore</i>	<p>Folgende Zertifikatstypen werden unterstützt:</p> <ol style="list-style-type: none"> 1. Clientseitig erzeugtes Zertifikat: Pfad zum Verzeichnis, in dem sich die Zertifikats-Datei (.cer) und die Datei mit dem privaten Schlüssel (.p12) befinden. Diese Kryptomittel wurden mit EricCreateKey() erzeugt. Der Pfad zum Verzeichnis ist bei clientseitig erzeugten Zertifikaten relativ zum aktuellen Arbeitsverzeichnis oder absolut anzugeben. 2. Software-Portalzertifikat: Pfad zur Software-Zertifikatsdatei (i.d.R. mit der Endung .pfx). Der Pfad zur Datei ist bei Software-Zertifikaten relativ zum aktuellen Arbeitsverzeichnis oder absolut anzugeben. 3. Sicherheitsstick: Pfad zur Treiberdatei, siehe (2). Bitte beachten, dass der Treiber betriebssystemabhängig sein kann. Weitere Informationen in der Anleitung zum Sicherheitsstick oder unter https://www.sicherheitsstick.de. 4. Signaturkarte: Pfad zur Treiberdatei, welcher einen Zugriff auf die Signaturkarte ermöglicht, siehe (2). Weitere Informationen in der Anleitung zur Signaturkarte.
in	<i>pin</i>	PIN für den Zugriff auf den privaten Schlüssel des Zertifikats.
in	<i>keyType</i>	<p>Mögliche Eingabewerte:</p> <ul style="list-style-type: none"> ◦ 0: eSignatureKey: Schlüssel für die Signatur von Daten, siehe (1). ◦ 1: eEncryptionKey: Schlüssel für die Verschlüsselung von Daten, siehe (1).

(1) Bei einem Zertifikat wie dem mit [EricCreateKey\(\)](#) clientseitig erzeugten Zertifikat (CEZ), das nur einen einzigen, gemeinsamen Schlüssel für Signatur und Verschlüsselung besitzt, sind beide Eingabewerte erlaubt. Die Werte beziehen sich dann beide auf denselben Schlüssel.

(2) Bei Sicherheitssticks und Signaturkarten ist bei der Angabe des Treibers der Suchmechanismus nach dynamischen Modulen des jeweiligen Betriebssystems zu berücksichtigen. Weitere Informationen sind z.B. unter Windows der Dokumentation der `LoadLibrary()` oder unter Linux und macOS der Dokumentation der `dlopen()` zu entnehmen.

Pfade müssen auf Windows in der für Datei-Funktionen benutzten ANSI-Codepage, auf Linux, AIX und Linux Power in der für das Dateisystem benutzten Locale und auf macOS in der "decomposed form" von UTF-8 übergeben werden. Bitte weitere Betriebssystemspezifika bzgl. nicht erlaubter Zeichen in Pfaden und Pfadtrennzeichen beachten.

Für Details zu Pfaden im ERiC siehe [ERiC-Entwicklerhandbuch.pdf](#), Kapitel "Übergabe von Pfaden an ERiC API-Funktionen".

Es wird empfohlen, geöffnete Zertifikatshandle zu schließen, bevor mit der API-Funktion [EricPruefeZertifikatPin\(\)](#) das gewünschte Zertifikat geprüft wird.

Zu beachten:

Eine falsche PIN-Eingabe erhöht bei Sicherheitsstick und Signaturkarte den Zähler für Fehlversuche. Welche Zertifikatstypen aufgrund von 3 Fehlversuchen gesperrt werden, ist im [ERiC-Entwicklerhandbuch.pdf](#), Kap. "Das Portalzertifikat (POZ)" beschrieben.

Rückgabe:

- [ERIC_OK](#)
- [ERIC_CRYPT_E_PIN_WRONG](#)
- [ERIC_CRYPT_NICHT_UNTERSTUETZTES_PSE_FORMAT](#)
- [ERIC_CRYPT_EIDKARTE_NICHT_UNTERSTUETZT](#)
- [ERIC_CRYPT_E_PSE_PATH](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)

ERICAPI_IMPORT int **EricRegistriereFortschrittCallback** (**EricFortschrittCallback**
funktion, void * benutzerdaten)

Die `funktion` wird als Callback-Funktion für [EricBearbeiteVorgang\(\)](#) registriert.

Die registrierte Callback-Funktion wird von der Funktion [EricBearbeiteVorgang\(\)](#) aufgerufen, um bei der Verarbeitung den Fortschritt der einzelnen Arbeitsbereiche anzuzeigen.

Parameter:

in	<i>funktion</i>	Zeiger auf die zu registrierende Funktion oder <code>NULL</code>
in	<i>benutzerdaten</i>	Zeiger, der der registrierten Funktion immer mitgegeben wird. Die Anwendung kann diesen Parameter dazu verwenden, einen Zeiger auf eigene Daten oder Funktionen an die zu registrierende Funktion übergeben zu lassen.

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_UNGUELTIGER_PARAMETER](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)

Bemerkungen:

- Wenn eine zuvor registrierte Funktion nicht mehr aufgerufen werden soll, ist [EricRegistriereFortschrittCallback\(\)](#) mit dem Wert `NULL` im Parameter `funktion` aufzurufen.
- Es ist nicht erlaubt eine ERiC API-Funktion aus einer Callback-Funktion aufzurufen.
- Die Verarbeitung im Callback findet synchron statt. Deshalb sollte der Callback sehr schnell ausgeführt werden.

Siehe auch:

- [EricFortschrittCallback](#)
- [EricBearbeiteVorgang\(\)](#)
- [ERiC-Entwicklerhandbuch.pdf](#), Kap. "Funktionen für Fortschrittcallbacks"

ERICAPI_IMPORT int EricRegistriereGlobalenFortschrittCallback (EricFortschrittCallback funktion, void * benutzerdaten)

Die registrierte `funktion` wird als Callback-Funktion von [EricBearbeiteVorgang\(\)](#) aufgerufen und zeigt den Gesamtfortschritt der Verarbeitung an.

Parameter:

in	<i>funktion</i>	Zeiger auf die zu registrierende Funktion oder <code>NULL</code>
in	<i>benutzerdaten</i>	Zeiger, der der registrierten Funktion immer mitgegeben wird. Die Anwendung kann diesen Parameter dazu verwenden, einen Zeiger auf eigene Daten oder Funktionen an die zu registrierende Funktion übergeben zu lassen.

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_UNGUELTIGER_PARAMETER](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)

Bemerkungen:

- Wenn eine zuvor registrierte Funktion nicht mehr aufgerufen werden soll, ist [EricRegistriereGlobalenFortschrittCallback\(\)](#) mit dem Wert `NULL` im Parameter `funktion` aufzurufen.
- Es ist nicht erlaubt eine ERiC API-Funktion aus einer Callback-Funktion aufzurufen.
- Die Verarbeitung im Callback findet synchron statt. Deshalb sollte der Callback sehr schnell ausgeführt werden.

Siehe auch:

- [EricBearbeiteVorgang\(\)](#)
- [ERiC-Entwicklerhandbuch.pdf](#), Kap. "Funktionen für Fortschrittcallbacks"

ERICAPI_IMPORT int **EricRegistriereLogCallback** (**EricLogCallback** funktion, **uint32_t** schreibeEricLogDatei, void * benutzerdaten)

Die registrierte `funktion` wird als Callback-Funktion für jede Lognachricht aufgerufen. Die Ausgabe entspricht einer Zeile im `eric.log`.

Parameter:

in	<i>funktion</i>	Zeiger auf die zu registrierende Funktion oder NULL.
in	<i>schreibeEricLogDatei</i>	Log-Nachrichten im <code>eric.log</code> : <ul style="list-style-type: none"> 1 Jede Log-Nachricht wird nach <code>eric.log</code> geschrieben. Der Parameter <code>funktion</code> kann auf eine Funktion zeigen oder NULL sein. 0 Falls <code>funktion != NULL</code> werden keine Log-Nachrichten nach <code>eric.log</code> geschrieben, andernfalls werden die Log-Nachrichten nach <code>eric.log</code> geschrieben.
in	<i>benutzerdaten</i>	Zeiger, welcher der registrierten Funktion immer mitgegeben wird. Die Anwendung kann diesen Parameter dazu verwenden, einen Zeiger auf eigene Daten oder Funktionen an die zu registrierende Funktion übergeben zu lassen.

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_UNGUELTIGER_PARAMETER](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)

Bemerkungen:

- Wenn eine zuvor registrierte Funktion nicht mehr aufgerufen werden soll, ist [EricRegistriereLogCallback\(\)](#) mit dem Wert `NULL` im Parameter `funktion` aufzurufen (=Deregistrierung).
- Vor dem Beenden der Steueranwendung ist eine registrierte Funktion zu deregistrieren, da es sonst zu einem Absturz kommen kann.
- Es ist nicht erlaubt eine ERiC API-Funktion aus einer Callback-Funktion aufzurufen.
- Die Verarbeitung im Callback findet synchron statt. Deshalb sollte der Callback sehr schnell ausgeführt werden.

ERICAPI_IMPORT EricRueckgabepufferHandle EricRueckgabepufferErzeugen (void)

Diese API-Funktion erzeugt einen Rückgabepuffer und gibt ein Handle darauf zurück.

Die von dieser Funktion erzeugten Rückgabepuffer werden verwendet, um die Ausgaben von ERiC-Funktionen (z.B. [EricBearbeiteVorgang\(\)](#)) aufzunehmen. Dazu wird das Rückgabepuffer-Handle für den Schreibvorgang an die ausgebende Funktion übergeben.

Zum Auslesen des von den API-Funktionen beschriebenen Puffers wird das Rückgabepuffer-Handle an [EricRueckgabepufferInhalt\(\)](#) übergeben. Ein einmal erzeugtes Rückgabepuffer-Handle kann für weitere nachfolgende Aufrufe von ERiC API-Funktionen wiederverwendet werden. Bei einer Wiederverwendung eines Handles werden frühere Inhalte überschrieben. Nach Verwendung muss jeder Rückgabepuffer mit [EricRueckgabepufferFreigeben\(\)](#) freigegeben werden.

Rückgabepuffer sind der Singlethreading-API bzw. einer ERiC-Instanz der Multithreading-API fest zugeordnet. Die Funktionen der ERiC API, die einen Rückgabepuffer entgegennehmen, geben den Fehlercode [ERIC_GLOBAL_PUFFER_UNGLEICHER_INSTANZ](#) zurück, wenn der übergebene Rückgabepuffer:

- mit der Singlethreading-API erzeugt worden ist und dann mit der Multithreading-API verwendet wird.
- mit der Multithreading-API erzeugt worden ist und dann mit der Singlethreading-API verwendet wird.
- mit einer ERiC-Instanz erzeugt worden ist und dann mit einer anderen Instanz verwendet wird.

Rückgabe:

- [EricRueckgabepufferHandle](#) im Erfolgsfall.
- NULL im Fehlerfall.

Siehe auch:

- [EricRueckgabepufferLaenge\(\)](#)
- [EricRueckgabepufferInhalt\(\)](#)
- [EricRueckgabepufferFreigeben\(\)](#)

ERICAPI_IMPORT int EricRueckgabepufferFreigeben (EricRueckgabepufferHandle handle)

Der durch das `handle` bezeichnete Rückgabepuffer wird freigegeben.

Das Handle darf danach nicht weiter verwendet werden. Es wird daher empfohlen, Handle-Variablen nach der Freigabe explizit auf NULL zu setzen.

Parameter:

in	<i>handle</i>	Handle auf einen mit EricRueckgabepufferErzeugen() angelegten Rückgabepuffer. Dieser Rückgabepuffer darf nicht bereits freigegeben worden sein.
----	---------------	---

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_UNGUELTIGER_PARAMETER](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)

Siehe auch:

- [EricRueckgabepufferErzeugen\(\)](#)
- [EricRueckgabepufferLaenge\(\)](#)
- [EricRueckgabepufferInhalt\(\)](#)

ERICAPI_IMPORT `const char * EricRueckgabepufferInhalt (EricRueckgabepufferHandle handle)`

Der durch das `handle` bezeichnete Inhalt des Rückgabepuffers wird zurückgegeben.

Der zurückgegebene Zeiger verweist auf ein Byte-Array, das alle in den Rückgabepuffer geschriebenen Bytes sowie eine abschließende NULL-Terminierung enthält. Dieses Array existiert so lange im Speicher, bis der Rückgabepuffer entweder (bei einer Wiederverwendung des Handles) erneut beschrieben oder der Puffer explizit freigegeben wird.

Parameter:

in	<i>handle</i>	Handle auf einen mit EricRueckgabepufferErzeugen() angelegten Rückgabepuffer. Dieser Rückgabepuffer darf nicht bereits freigegeben worden sein.
----	---------------	---

Rückgabe:

- Zeiger auf den NULL-terminierten Rückgabepufferinhalt, wenn ein gültiges Handle übergeben wird.
- NULL: Bei Übergabe des ungültigen Handles NULL.

Siehe auch:

- [EricRueckgabepufferErzeugen\(\)](#)
- [EricRueckgabepufferLaenge\(\)](#)
- [EricRueckgabepufferFreigegeben\(\)](#)

ERICAPI_IMPORT uint32_t EricRueckgabepufferLaenge (EricRueckgabepufferHandle handle)

Die Länge des Rückgabepufferinhalts wird zurückgegeben.

Die zurückgegebene Zahl entspricht der Anzahl von Bytes, die von einer zuvor aufgerufenen ERiC API-Funktion in den Rückgabepuffer geschrieben wurden. Die NULL-Terminierung, die bei Aufruf von [EricRueckgabepufferInhalt\(\)](#) an das zurückgegebene Byte-Array angefügt wird, wird bei dieser Längenangabe nicht berücksichtigt.

Parameter:

in	<i>handle</i>	Handle auf einen mit EricRueckgabepufferErzeugen() angelegten Rückgabepuffer. Dieser Rückgabepuffer darf nicht bereits freigegeben worden sein.
----	---------------	---

Rückgabe:

- Anzahl der in den Rückgabepuffer geschriebenen Bytes, wenn ein gültiges Handle übergeben wird.
- 0: Bei Übergabe des ungültigen Handles NULL.

Siehe auch:

- [EricRueckgabepufferErzeugen\(\)](#)
- [EricRueckgabepufferInhalt\(\)](#)
- [EricRueckgabepufferFreigegeben\(\)](#)

ERICAPI_IMPORT int EricSystemCheck (void)

Es werden Plattform-, Betriebssystem- und ERiC-Informationen ausgegeben.

Diese Funktion liefert Informationen über die verwendeten ERiC-Bibliotheken, ERiC-Druckvorlagen, die eingesetzte Plattform, den Arbeitsspeicher und das verwendete Betriebssystem.

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- weitere, siehe [eric_fehlercodes.h](#)

Siehe auch:

- [EricVersion\(\)](#)

ERICAPI_IMPORT int EricVersion (EricRueckgabepufferHandle rueckgabeXmlPuffer)

Es wird eine Liste sämtlicher Produkt- und Dateiversionen der verwendeten ERiC-Bibliotheken als XML-Daten zurückgegeben.

Diese Funktion kann bei auftretenden Fehlern die Fehlersuche beschleunigen und Supportfälle unterstützen.

Parameter:

out	<i>rueckgabeXmlPuffer</i>	Handle auf einen Rückgabepuffer, in den zu allen ERiC-Bibliotheken die Produkt- und Dateiversionen als XML-Daten nach XML Schema Definition Dokumentation\API-Rueckgabe-Schemata\EricVersion.xsd geschrieben werden. Zur Erzeugung, Verwendung und Freigabe von Rückgabepuffern siehe Dokumentation zu EricRueckgabepufferHandle .
-----	---------------------------	---

Beispiel:

```
<?xml version="1.0" encoding="UTF-8"?>
<EricVersion xmlns="http://www.elster.de/EricXML/1.0/EricVersion">
  <Bibliothek>
    <Name>ericapi.dll</Name>
    <Produktversion>99, 1, 2, 32767</Produktversion>
    <Dateiversion>2008, 3, 5, 0</Dateiversion>
  </Bibliothek>
  <Bibliothek>
    <Name>ericctrl.dll</Name>
    <Produktversion>99, 1, 2, 32767</Produktversion>
    <Dateiversion>2008, 3, 5, 0</Dateiversion>
  </Bibliothek>
  (...)
</EricVersion>
```

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- weitere, siehe [eric_fehlercodes.h](#)

Siehe auch:

- [EricSystemCheck\(\)](#)

ERICAPI_IMPORT int EricZertifikatOeffnenAusBytes ([EricZertifikatHandle](#) * hToken, const [byteChar](#) * pkcs12Container, [uint32_t](#) containerGroesse, const [byteChar](#) * pkcs12Passwort)

Für das im Speicher übergebene Zertifikat wird das Handle `hToken` zurückgeliefert.

Die ERiC API benötigt Zertifikat-Handles typischerweise bei kryptografischen Operationen.

Zertifikat-Handles sollten möglichst frühzeitig, d.h. wenn sie nicht mehr benötigt werden, mit [EricCloseHandleToCertificate\(\)](#) freigegeben werden, spätestens jedoch zum Programmende bzw. vor dem Entladen der Bibliothek `ericapi`.

Parameter:

out	<i>hToken</i>	Handle zu einem der folgenden Zertifikate: <ul style="list-style-type: none"> ◦ Portalzertifikat ◦ clientseitig erzeugtes Zertifikat
in	<i>pkcs12Container</i>	Speicheradresse des Zertifikats im PKCS#12-Format. Es werden nur passwortgeschützte PKCS#12-Container akzeptiert.
in	<i>containerGroesse</i>	Größe des PKCS#12-Containers in Bytes
in	<i>pkcs12Passwort</i>	Das Passwort oder die PIN des PKCS#12-Containers. Dieser Parameter darf nicht NULL sein.

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)
- [ERIC_CRYPT_NICHT_UNTERSTUETZTES_PSE_FORMAT](#)
- [ERIC_CRYPT_E_MAX_SESSION](#)
- [ERIC_CRYPT_E_PSE_PATH](#)
- [ERIC_CRYPT_E_BUSY](#)
- [ERIC_CRYPT_E_P11_SLOT_EMPTY](#)
- [ERIC_CRYPT_E_NO_SIG_ENC_KEY](#)
- [ERIC_CRYPT_E_PIN_WRONG](#)
- [ERIC_CRYPT_E_LOAD_DLL](#)
- [ERIC_CRYPT_E_NO_SERVICE](#)
- [ERIC_CRYPT_E_ESICL_EXCEPTION](#)

Siehe auch:

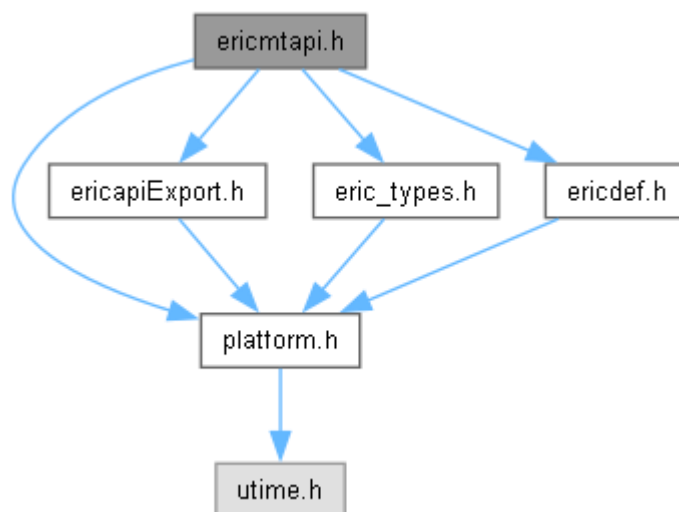
- [EricCloseHandleToCertificate\(\)](#)
- [EricGetPinStatus\(\)](#)

ericmtapi.h-Dateireferenz

Deklaration der ERiC API-Funktionen für die Multithreading-API.

```
#include "platform.h"
#include "ericapiExport.h"
#include "eric_types.h"
#include "ericdef.h"
```

Include-Abhängigkeitsdiagramm für ericmtapi.h:



Funktionen

- [ERICAPI_IMPORT](#) int [EricMtBearbeiteVorgang](#) ([EricInstanzHandle](#) instanz, const char *datenpuffer, const char *datenartVersion, [uint32_t](#) bearbeitungsFlags, const [eric_druck_parameter_t](#) *druckParameter, const [eric_verschlüsselungs_parameter_t](#) *cryptoParameter, [EricTransferHandle](#) *transferHandle, [EricRueckgabepufferHandle](#) rueckgabeXmlPuffer, [EricRueckgabepufferHandle](#) serverantwortXmlPuffer)

Diese API-Funktion ist die zentrale Schnittstellenfunktion zur Kommunikation mit dem ELSTER-Annahmeserver.

- [ERICAPI_IMPORT](#) int [EricMtChangePassword](#) ([EricInstanzHandle](#) instanz, const [byteChar](#) *psePath, const [byteChar](#) *oldPin, const [byteChar](#) *newPin)

Die PIN für ein clientseitig erzeugtes Zertifikat (CEZ) wird geändert.

- [ERICAPI_IMPORT](#) int [EricMtPruefeBuFaNummer](#) ([EricInstanzHandle](#) instanz, const [byteChar](#) *steuernummer)

Die Bundesfinanzamtsnummer wird überprüft.
- [ERICAPI_IMPORT](#) int [EricMtCheckXML](#) ([EricInstanzHandle](#) instanz, const char *xml, const char *datenartVersion, [EricRueckgabepufferHandle](#) fehlertextPuffer)

Das xml wird gegen das Schema der datenartVersion validiert.
- [ERICAPI_IMPORT](#) int [EricMtCloseHandleToCertificate](#) ([EricInstanzHandle](#) instanz, [EricZertifikatHandle](#) hToken)

Das Zertifikat-Handle hToken wird freigegeben.
- [ERICAPI_IMPORT](#) int [EricMtCreateKey](#) ([EricInstanzHandle](#) instanz, const [byteChar](#) *pin, const [byteChar](#) *pfad, const [eric_zertifikat_parameter_t](#) *zertifikatInfo)

Es werden die Kryptomittel für ein clientseitig erzeugtes Zertifikat (CEZ) in einem Verzeichnis des Dateisystems erstellt.
- [ERICAPI_IMPORT](#) int [EricMtCreateTH](#) ([EricInstanzHandle](#) instanz, const char *xml, const char *verfahren, const char *datenart, const char *vorgang, const char *testmerker, const char *herstellerId, const char *datenLieferant, const char *versionClient, const [byteChar](#) *publicKey, [EricRueckgabepufferHandle](#) xmlRueckgabePuffer)

Diese Funktion erzeugt einen TransferHeader.
- [ERICAPI_IMPORT](#) int [EricMtCreateUUID](#) ([EricInstanzHandle](#) instanz, [EricRueckgabepufferHandle](#) uuidRueckgabePuffer)

Erzeugt einen Version 4 Universally Unique Identifier (UUID) gemäß RFC 4122.
- [ERICAPI_IMPORT](#) int [EricMtDekodiereDaten](#) ([EricInstanzHandle](#) instanz, [EricZertifikatHandle](#) zertifikatHandle, const [byteChar](#) *pin, const [byteChar](#) *base64Eingabe, [EricRueckgabepufferHandle](#) rueckgabePuffer)

Es werden die mit der Datenabholung abgeholten und verschlüsselten Daten entschlüsselt.
- [ERICAPI_IMPORT](#) int [EricMtEinstellungAlleZuruecksetzen](#) ([EricInstanzHandle](#) instanz)

Alle Einstellungen, der übergebenen ERiC-Instanz werden auf den jeweiligen Standardwert zurückgesetzt.

- [ERICAPI_IMPORT](#) int [EricMtEinstellungLesen](#) ([EricInstanzHandle](#) instanz, const char *name, [EricRueckgabepufferHandle](#) rueckgabePuffer)
Der Wert der API-Einstellung `name` wird im `rueckgabePuffer` zurückgeliefert.
- [ERICAPI_IMPORT](#) int [EricMtEinstellungSetzen](#) ([EricInstanzHandle](#) instanz, const char *name, const char *wert)
Die API-Einstellung `name` wird auf den `wert` gesetzt.
- [ERICAPI_IMPORT](#) int [EricMtEinstellungZuruecksetzen](#) ([EricInstanzHandle](#) instanz, const char *name)
Der Wert der API-Einstellung `name` wird auf den Standardwert zurückgesetzt.
- [ERICAPI_IMPORT](#) int [EricMtEntladePlugins](#) ([EricInstanzHandle](#) instanz)
Für die übergebene ERiC-Instanz werden alle verwendeten Plugin-Bibliotheken entladen und deren Speicher wird freigegeben.
- [ERICAPI_IMPORT](#) int [EricMtFormatEWaz](#) ([EricInstanzHandle](#) instanz, const [byteChar](#) *ewAzElster, [EricRueckgabepufferHandle](#) ewAzBescheidPuffer)
Konvertiert ein Einheitswert-Aktenzeichen im ELSTER-Format in ein landesspezifisches Bescheidformat.
- [ERICAPI_IMPORT](#) int [EricMtFormatStNr](#) ([EricInstanzHandle](#) instanz, const [byteChar](#) *eingabeSteuernummer, [EricRueckgabepufferHandle](#) rueckgabePuffer)
Die Steuernummer `eingabeSteuernummer` wird in das Bescheid-Format des jeweiligen Bundeslandes umgewandelt.
- [ERICAPI_IMPORT](#) int [EricMtGetAuswahlListen](#) ([EricInstanzHandle](#) instanz, const char *datenartVersion, const char *feldkennung, [EricRueckgabepufferHandle](#) rueckgabeXmlPuffer)
Die Auswahlliste(n) für `datenartVersion` oder `feldkennung` wird zurückgeliefert.
- [ERICAPI_IMPORT](#) int [EricMtGetErrorMessageFromXMLAnswer](#) ([EricInstanzHandle](#) instanz, const char *xml, [EricRueckgabepufferHandle](#) transferticketPuffer, [EricRueckgabepufferHandle](#) returncodeTHPuffer,

[EricRueckgabepufferHandle](#) fehlertextTHPuffer, [EricRueckgabepufferHandle](#) returncodesUndFehlertexteNDHXmlPuffer)

Aus dem Antwort-XML des Finanzamtsservers wird das Transferticket und Returncodes/Fehlermeldungen zurückgegeben.

- [ERICAPI_IMPORT](#) int [EricMtGetHandleToCertificate](#) ([EricInstanzHandle](#) instanz, [EricZertifikatHandle](#) *hToken, [uint32_t](#) *iInfoPinSupport, const [byteChar](#) *pathToKeystore)

Für das übergebene Zertifikat in `pathToKeystore` wird das Handle `hToken` und die unterstützten PIN-Werte `iInfoPinSupport` zurückgeliefert.

- [ERICAPI_IMPORT](#) int [EricMtZertifikatOeffnenAusBytes](#) ([EricInstanzHandle](#) instanz, [EricZertifikatHandle](#) *hToken, const [byteChar](#) *pkcs12Container, [uint32_t](#) containerGroesse, const [byteChar](#) *pkcs12Passwort)

Für das im Speicher übergebene Zertifikat wird das Handle `hToken` zurückgeliefert.

- [ERICAPI_IMPORT](#) int [EricMtGetPinStatus](#) ([EricInstanzHandle](#) instanz, [EricZertifikatHandle](#) hToken, [uint32_t](#) *pinStatus, [uint32_t](#) keyType)

Der PIN-Status wird für ein passwortgeschütztes Kryptomittel abgefragt und in `pinStatus` zurückgegeben.

- [ERICAPI_IMPORT](#) int [EricMtGetPublicKey](#) ([EricInstanzHandle](#) instanz, const [eric_verschluesselungs_parameter_t](#) *cryptoParameter, [EricRueckgabepufferHandle](#) rueckgabePuffer)

Es wird der öffentliche Schlüssel als base64-kodierte Zeichenkette für das übergebene Zertifikat-Handle in `cryptoParameter` zurückgeliefert. Von den beiden öffentlichen Schlüsseln, die in ELSTER-Zertifikaten enthalten sind, wird der Verschlüsselungsschlüssel (Encryption Key) zurückgegeben.

- [ERICAPI_IMPORT](#) int [EricMtHoleFehlerText](#) ([EricInstanzHandle](#) instanz, int fehlerkode, [EricRueckgabepufferHandle](#) rueckgabePuffer)

Es wird die Klartextfehlermeldung zu dem `fehlerkode` ermittelt.

- [ERICAPI_IMPORT](#) int [EricMtHoleFinanzaemter](#) ([EricInstanzHandle](#) instanz, const [byteChar](#) *finanzamtLandNummer, [EricRueckgabepufferHandle](#) rueckgabeXmlPuffer)

Es wird die Finanzamtliste für eine bestimmte `finanzamtLandNummer` zurückgegeben.

- [ERICAPI_IMPORT](#) int [EricMtHoleFinanzamtLandNummern](#) ([EricInstanzHandle](#) instanz, [EricRueckgabepufferHandle](#) rueckgabeXmlPuffer)

Die Liste aller Finanzamtlandnummern wird zurückgegeben.
- [ERICAPI_IMPORT](#) int [EricMtHoleFinanzamtsdaten](#) ([EricInstanzHandle](#) instanz, const [byteChar](#) bufaNr[5], [EricRueckgabepufferHandle](#) rueckgabeXmlPuffer)

Die finanzamtsdaten werden für eine Bundesfinanzamtsnummer zurückgegeben.
- [ERICAPI_IMPORT](#) int [EricMtHoleTestfinanzaemter](#) ([EricInstanzHandle](#) instanz, [EricRueckgabepufferHandle](#) rueckgabeXmlPuffer)

Die Testfinanzamtliste wird in rueckgabeXmlPuffer zurückgegeben.
- [ERICAPI_IMPORT](#) int [EricMtHoleZertifikatEigenschaften](#) ([EricInstanzHandle](#) instanz, [EricZertifikatHandle](#) hToken, const [byteChar](#) *pin, [EricRueckgabepufferHandle](#) rueckgabeXmlPuffer)

Die Eigenschaften des übergebenen Zertifikats werden im rueckgabeXmlPuffer zurückgegeben.
- [ERICAPI_IMPORT](#) int [EricMtHoleZertifikatFingerabdruck](#) ([EricInstanzHandle](#) instanz, const [eric_verschluesselungs_parameter_t](#) *cryptoParameter, [EricRueckgabepufferHandle](#) fingerabdruckPuffer, [EricRueckgabepufferHandle](#) signaturPuffer)

Der Fingerabdruck und dessen Signatur wird für das übergebene Zertifikat zurückgegeben.
- [ERICAPI_IMPORT](#) [EricInstanzHandle](#) [EricMtInstanzErzeugen](#) (const char *pluginPfad, const char *logPfad)

Erstellt und initialisiert eine neue ERiC-Instanz.
- [ERICAPI_IMPORT](#) int [EricMtInstanzFreigeben](#) ([EricInstanzHandle](#) instanz)

Die übergebene ERiC-Instanz wird beendet und deren Speicher freigegeben.
- [ERICAPI_IMPORT](#) int [EricMtMakeElsterStnr](#) ([EricInstanzHandle](#) instanz, const [byteChar](#) *steuernrBescheid, const [byteChar](#) landesnr[2+1], const [byteChar](#) bundesfinanzamtsnr[4+1], [EricRueckgabepufferHandle](#) steuernrPuffer)

Es wird eine Steuernummer im ELSTER-Steuernummerformat erzeugt.

- [ERICAPI_IMPORT](#) int [EricMtMakeElsterEWaz](#) ([EricInstanzHandle](#) instanz, const [byteChar](#) *ewAzBescheid, const [byteChar](#) *landeskuerzel, [EricRueckgabepufferHandle](#) ewAzElsterPuffer)

Konvertiert ein Einheitswert-Aktenzeichen in das ELSTER-Format.

- [ERICAPI_IMPORT](#) int [EricMtPruefeBIC](#) ([EricInstanzHandle](#) instanz, const [byteChar](#) *bic)

Die bic wird auf Gültigkeit überprüft.

- [ERICAPI_IMPORT](#) int [EricMtPruefeIBAN](#) ([EricInstanzHandle](#) instanz, const [byteChar](#) *iban)

Die iban wird auf Gültigkeit überprüft.

- [ERICAPI_IMPORT](#) int [EricMtPruefeEWaz](#) ([EricInstanzHandle](#) instanz, const [byteChar](#) *einheitswertAz)

Überprüft ein Einheitswert-Aktenzeichen im ELSTER-Format auf Gültigkeit.

- [ERICAPI_IMPORT](#) int [EricMtPruefeIdentifikationsMerkmal](#) ([EricInstanzHandle](#) instanz, const [byteChar](#) *steuerId)

Die steuerId wird auf Gültigkeit überprüft.

- [ERICAPI_IMPORT](#) int [EricMtPruefeSteuernummer](#) ([EricInstanzHandle](#) instanz, const [byteChar](#) *steuernummer)

Die steuernummer wird einschließlich Bundesfinanzamtsnummer auf formale Richtigkeit geprüft.

- [ERICAPI_IMPORT](#) int [EricMtPruefeWldNr](#) ([EricInstanzHandle](#) instanz, const [byteChar](#) *wldNr)

Die Wirtschafts-Identifikationsnummer (W-IdNr.) wird auf formale Gültigkeit überprüft.

- [ERICAPI_IMPORT](#) int [EricMtPruefeZertifikatPin](#) ([EricInstanzHandle](#) instanz, const [byteChar](#) *pathToKeystore, const [byteChar](#) *pin, [uint32_t](#) keyType)

Prüft, ob die pin zum Zertifikat pathToKeystore passt. Nicht anwendbar auf Ad Hoc-Zertifikate (AHZ), die für einen neuen Personalausweis (nPA) ausgestellt sind.

- [ERICAPI_IMPORT](#) int [EricMtRegistriereFortschrittCallback](#) ([EricInstanzHandle](#) instanz, [EricFortschrittCallback](#) funktion, void *benutzerdaten)

Die funktion wird als Callback-Funktion für [EricMtBearbeiteVorgang\(\)](#) registriert.
- [ERICAPI_IMPORT](#) int [EricMtRegistriereGlobalenFortschrittCallback](#) ([EricInstanzHandle](#) instanz, [EricFortschrittCallback](#) funktion, void *benutzerdaten)

Die registrierte funktion wird als Callback-Funktion von [EricMtBearbeiteVorgang\(\)](#) aufgerufen und zeigt den Gesamtfortschritt der Verarbeitung an.
- [ERICAPI_IMPORT](#) int [EricMtRegistriereLogCallback](#) ([EricInstanzHandle](#) instanz, [EricLogCallback](#) funktion, [uint32_t](#) schreibeEricLogDatei, void *benutzerdaten)

Die registrierte funktion wird als Callback-Funktion für jede Lognachricht aufgerufen. Die Ausgabe entspricht einer Zeile im eric.log.
- [ERICAPI_IMPORT](#) [EricRueckgabepufferHandle](#) [EricMtRueckgabepufferErzeugen](#) ([EricInstanzHandle](#) instanz)

Diese API-Funktion erzeugt einen Rückgabepuffer und gibt ein Handle darauf zurück.
- [ERICAPI_IMPORT](#) int [EricMtRueckgabepufferFreigeben](#) ([EricInstanzHandle](#) instanz, [EricRueckgabepufferHandle](#) handle)

Der durch das handle bezeichnete Rückgabepuffer wird freigegeben.
- [ERICAPI_IMPORT](#) const char * [EricMtRueckgabepufferInhalt](#) ([EricInstanzHandle](#) instanz, [EricRueckgabepufferHandle](#) handle)

Der durch das handle bezeichnete Inhalt des Rückgabepuffers wird zurückgegeben.
- [ERICAPI_IMPORT](#) [uint32_t](#) [EricMtRueckgabepufferLaenge](#) ([EricInstanzHandle](#) instanz, [EricRueckgabepufferHandle](#) handle)

Die Länge des Rückgabepufferinhalts wird zurückgegeben.
- [ERICAPI_IMPORT](#) int [EricMtSystemCheck](#) ([EricInstanzHandle](#) instanz)

Es werden Plattform-, Betriebssystem- und ERiC-Informationen ausgegeben.

- [ERICAPI_IMPORT](#) int [EricMtVersion](#) ([EricInstanzHandle](#) instanz, [EricRueckgabepufferHandle](#) rueckgabeXmlPuffer)

Es wird eine Liste sämtlicher Produkt- und Dateiversionen der verwendeten ERiC-Bibliotheken als XML-Daten zurückgegeben.

Ausführliche Beschreibung

Deklaration der ERiC API-Funktionen für die Multithreading-API.

Dokumentation der Funktionen

[ERICAPI_IMPORT](#) int [EricMtBearbeiteVorgang](#) ([EricInstanzHandle](#) instanz, const char * datenpuffer, const char * datenartVersion, [uint32_t](#) bearbeitungsFlags, const [eric_druck_parameter_t](#) * druckParameter, const [eric_verschluesselungs_parameter_t](#) * cryptoParameter, [EricTransferHandle](#) * transferHandle, [EricRueckgabepufferHandle](#) rueckgabeXmlPuffer, [EricRueckgabepufferHandle](#) serverantwortXmlPuffer)

Diese API-Funktion ist die zentrale Schnittstellenfunktion zur Kommunikation mit dem ELSTER-Annahmeserver.

Als Austauschformat wird XML verwendet, siehe Kap. "Datenverarbeitung mit ERiC" im [ERiC-Entwicklerhandbuch.pdf](#). Dort sind die Arbeitsabläufe von Einzel- und Sammellieferung beschrieben.

Die Funktion kann Steuerdaten plausibilisieren, an den ELSTER-Annahmeserver übertragen und ausdrucken, sowie Protokolle der Übertragung erzeugen. Die ProcessingFlags im Parameter `bearbeitungsFlags` definieren, welche der Schritte wie ausgeführt werden.

Je nach Anwendungsfall können die Daten authentifiziert übertragen werden und es kann ein PDF-Druck der Daten erfolgen. In diesen Fällen sind die Parameter `cryptoParameter` und `druckParameter` entsprechend zu befüllen. Die möglichen Parameterkombinationen und Druckkennzeichnungen können im [ERiC-Entwicklerhandbuch.pdf](#) nachgelesen werden.

Sind für einen Anwendungsfall mehrere voneinander abhängige Aufrufe von [EricMtBearbeiteVorgang\(\)](#) nötig, so ist der Parameter `transferHandle` zu übergeben. Dies ist derzeit nur für die Datenabholung der Fall.

Es werden an bestimmten Punkten der Verarbeitung benutzerdefinierte Callback Funktionen aufgerufen. Siehe hierzu [Fortschrittcallbacks](#).

Zur Erzeugung, Verwendung und Freigabe von Rückgabepuffern siehe Dokumentation zu [EricRueckgabepufferHandle](#).

Parameter:

in	<i>instanz</i>	Die ERiC-Instanz, auf der diese Funktion ausgeführt werden soll.
in	<i>datenpuffer</i>	Enthält die zu verarbeitenden XML-Daten.
in	<i>datenartVersion</i>	Die <code>datenartVersion</code> ist der Datenartversionmatrix.xml zu entnehmen. Dieser Parameter darf nicht NULL sein und muss zu den XML-Eingangsdaten passen. Siehe auch ERiC-Entwicklerhandbuch.pdf .

in	<i>bearbeitungsFlags</i>	<p>Oder-Verknüpfung von Bearbeitungsvorgaben. Anhand dieser Vorgaben werden die übergebenen Daten verarbeitet. Der Parameter darf nicht 0 sein, zu gültigen Werten siehe eric_bearbeitung_flag_t.</p> <p>Bei welchen Anwendungsfällen welche Flags möglich oder notwendig sind, ist im ERiC-Entwicklerhandbuch.pdf nachzulesen.</p>
in	<i>druckParameter</i>	<p>Parameter, der für den PDF-Druck benötigt wird, siehe eric_druck_parameter_t.</p> <p>Bei welchen Anwendungsfällen der Druckparameter möglich oder notwendig ist, ist im ERiC-Entwicklerhandbuch.pdf nachzulesen.</p> <p>Soll kein PDF-Druck erfolgen, so ist NULL zu übergeben.</p>
in	<i>cryptoParameter</i>	<p>Enthält die für den authentifizierten Versand benötigten Informationen und darf nur dann übergeben werden, siehe eric_verschluesselungs_parameter_t.</p> <p>Erfolgt kein authentifizierter Versand, so ist NULL zu übergeben.</p>
in,out	<i>transferHandle</i>	<p>Bei der Datenabholung ist ein Zeiger auf ein vom Aufrufer verwaltetes und anfangs mit 0 befülltes EricTransferHandle zu übergeben, über das die zusammenhängenden Versandvorgänge einer Datenabholung gebündelt werden (Bündelung der Versandvorgänge "Anforderung", "Abholung" und optional "Quittierung").</p> <p>Wenn bei der Datenabholung kein Versandflag gesetzt ist (nur Validierung), darf dem transferHandle auch ein Nullzeiger (NULL) übergeben werden.</p> <p>Bei allen anderen Anwendungsfällen ist immer NULL zu übergeben.</p>
out	<i>rueckgabeXmlPuffer</i>	<p>Handle auf einen Rückgabepuffer, in den beim Versand Telenummer und Ordnungsbegriff, Hinweise oder Fehler bei der Regelprüfung geschrieben werden.</p> <p>Siehe Inhalt des Rückgabepuffers und des Serverantwortpuffers und EricRueckgabepufferHandle.</p>
out	<i>serverantwortXmlPuffer</i>	<p>Handle auf einen Rückgabepuffer, in den beim Versand die Antwort des Empfangsservers geschrieben wird.</p> <p>Siehe Inhalt des Rückgabepuffers und des Serverantwortpuffers und EricRueckgabepufferHandle.</p>

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_UNGUELTIGER_PARAMETER](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_DATENARTVERSION_UNBEKANNT](#)
- [ERIC_GLOBAL_VERSCHLUESSELUNGS_PARAMETER_NICHT_ANGEGEBEN](#)
- [ERIC_GLOBAL_PRUEF_FEHLER](#) Plausibilitätsfehler in den Eingabedaten, die Fehlermeldungen werden im Rückgabepuffer `rueckgabeXmlPuffer` zurückgegeben. Siehe Abschnitt [Plausibilitätsfehler](#).
- [ERIC_GLOBAL_HINWEISE](#) Kann nur zurückgegeben werden, falls das Bearbeitungsflag [ERIC_PRUEFE_HINWEISE](#) angegeben wurde. Es wurden ausschließlich Hinweise zu den Eingabedaten gemeldet, die Hinweise werden im Rückgabepuffer `rueckgabeXmlPuffer` zurückgegeben. Siehe Abschnitt [Hinweise](#).
- [ERIC_GLOBAL_DATENSATZ_ZU_GROSS](#) Die maximal zulässige Größe des XML-Eingangsdatensatzes oder des zu übermittelnden, komprimierten, verschlüsselten und base64-kodierten Datenteils, siehe [ERiC-Entwicklerhandbuch.pdf](#), Kap. "Größenbegrenzung der Eingangsdaten", ist überschritten.
- [ERIC_TRANSFER_ERR_XML_HEADER](#), [ERIC_TRANSFER_ERR_XML_NHEADER](#) Die Serverantwort enthält Fehlermeldungen. Zur Auswertung kann entweder die Serverantwort selbst ausgewertet werden oder es wird [EricMtGetErrorMessageFromXMLAnswer\(\)](#) aufgerufen.
- [ERIC_IO_READER_SCHEMA_VALIDIERUNGSFEHLER](#)
- [ERIC_IO_PARSE_FEHLER](#)
- [ERIC_GLOBAL_COMMONDATA_NICHT_VERFUEGBAR](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- weitere, siehe [eric_fehlercodes.h](#)

Inhalt des Rückgabepuffers und des Serverantwortpuffers

Der Inhalt der Pufferspeicher kann mit [EricMtRueckgabepufferInhalt\(\)](#) abgefragt und ausgewertet werden. `rueckgabeXmlPuffer` gibt im [Erfolgsfall](#) oder bei [Plausibilitätsfehler](#) XML-Daten nach Schema [Dokumentation\API-Rueckgabe-Schemata\EricBearbeiteVorgang.xsd](#) zurück. `serverantwortXmlPuffer` gibt bei Sendevorgängen die Antwort des ELSTER- Annahmeservers zurück.

Nach dem Aufruf der Funktion müssen programmatisch folgende Fälle aufgrund des Rückgabewerts unterschieden werden.

Erfolgsfall

Sind alle Bearbeitungsschritte fehlerfrei durchlaufen worden, dann ist der Rückgabewert [ERIC_OK](#) und der Text im Pufferspeicher `rueckgabeXmlPuffer` enthält beim Versand XML-Daten mit generierter Telenummer und bei Neuaufnahmen den Ordnungsbegriff.

Beispiel:

```
<?xml version="1.0" encoding="UTF-8"?>
<EricBearbeiteVorgang
xmlns="http://www.elster.de/EricXML/1.1/EricBearbeiteVorgang">
  <Erfolg>
    <Telenummer>N55</Telenummer>
  </Erfolg>
</EricBearbeiteVorgang>
```

Beim Versand befindet sich zusätzlich im Pufferspeicher `serverantwortXmlPuffer` die Antwort des ELSTER-Aannahmeservers. Bei einer Datenabholung kann diese ausgewertet werden. Details hierzu befinden sich im [ERiC-Entwicklerhandbuch.pdf](#)

Hinweise

Falls das Bearbeitungsflag [ERIC_PRUEFE_HINWEISE](#) angegeben worden ist, kann der Rückgabewert [ERIC_GLOBAL_HINWEISE](#) zurückgegeben werden. Der Rückgabepuffer enthält dann die gemeldeten Hinweise.

Beispiel:

```
<?xml version="1.0" encoding="UTF-8"?>
<EricBearbeiteVorgang
xmlns="http://www.elster.de/EricXML/1.1/EricBearbeiteVorgang">
  <Hinweis>
    <Nutzdatenticket>1075</Nutzdatenticket>
    <Feldidentifikator>100001</Feldidentifikator>
    <Mehrfachzeilenindex>1</Mehrfachzeilenindex>
    <LfdNrVordruck>1</LfdNrVordruck>
    <VordruckZeilennummer>4</VordruckZeilennummer>
    <SemantischerIndex>PersonA</SemantischerIndex>
    <Untersachsbereich>5</Untersachsbereich>
    <RegelName>testRegelName</RegelName>
    <FachlicheHinweisId>9995</FachlicheHinweisId>
    <Text>Weitere Angaben können erforderlich sein</Text>
```

```
</Hinweis>
</EricBearbeiteVorgang>
```

Die einzelnen Elemente sind in der Schemadefinition

[Dokumentation\API-Rueckgabe-Schemata\EricBearbeiteVorgang.xsd](#)

dokumentiert. Wenn die Bearbeitungsflags [ERIC_PRUEFE_HINWEISE](#) und [ERIC_VALIDIERE](#) übergeben worden sind, wurden bei der Plausibilisierung keine Fehler gefunden. Es sind keine Fehler im Rückgabepuffer enthalten.

Plausibilitätsfehler

Bei fehlgeschlagener Plausibilitätsprüfung ist der Rückgabewert

[ERIC_GLOBAL_PRUEF_FEHLER](#), und die Fehler werden im Rückgabepuffer als XML-Daten zurückgeliefert.

Beispiel:

```
<?xml version="1.0" encoding="UTF-8"?>
<EricBearbeiteVorgang
xmlns="http://www.elster.de/EricXML/1.1/EricBearbeiteVorgang">
  <FehlerRegelpruefung>
    <Nutzdatenticket>1075</Nutzdatenticket>
    <Feldidentifikator>100001</Feldidentifikator>
    <Mehrfachzeilenindex>1</Mehrfachzeilenindex>
    <LfdNrVordruck>1</LfdNrVordruck>
    <VordruckZeilennummer>4</VordruckZeilennummer>
    <SemantischerIndex>PersonA</SemantischerIndex>
    <Untersachsbereich>5</Untersachsbereich>
    <RegelName>testRegelName</RegelName>
    <FachlicheFehlerId>9995</FachlicheFehlerId>
    <Text>Beim Ankreuzfeld muss der Wert 'X' angegeben werden.</Text>
  </FehlerRegelpruefung>
</EricBearbeiteVorgang>
```

Die einzelnen Elemente sind in der Schemadefinition

[Dokumentation\API-Rueckgabe-Schemata\EricBearbeiteVorgang.xsd](#)

dokumentiert. Wenn die Bearbeitungsflags [ERIC_PRUEFE_HINWEISE](#) und [ERIC_VALIDIERE](#) übergeben worden sind, kann der Rückgabepuffer auch Hinweise enthalten.

Fehler in der Serverantwort

Ist der Rückgabewert [ERIC_TRANSFER_ERR_XML_THEADER](#) oder

[ERIC_TRANSFER_ERR_XML_NHEADER](#) so enthält der Serverantwortpuffer Fehlermeldungen. Zur Auswertung kann entweder die Serverantwort selbst

ausgewertet werden oder es wird [EricMtGetErrormessagesFromXMLAnswer\(\)](#) aufgerufen.

Sonstige Fehler

Bei sonstigen Fehlern ist der Inhalt der Rückgabepuffer undefiniert. Um nähere Informationen über die Fehlerursache herauszufinden, kann [EricMtHoleFehlerText\(\)](#) mit dem Rückgabewert aufgerufen werden.

Fortschrittcallbacks

Während der Verarbeitung eines Anwendungsfalls werden die durch die Funktionen [EricMtRegistriereFortschrittCallback\(\)](#) und [EricMtRegistriereGlobalenFortschrittCallback\(\)](#) registrierten Callbacks aufgerufen.

Siehe auch:

- [ERiC-Entwicklerhandbuch.pdf](#), Kap. "Anwendungsfälle von EricBearbeiteVorgang()"
- [ERiC-Entwicklerhandbuch.pdf](#), Kap. der jeweiligen Datenart
- [ERiC-Entwicklerhandbuch.pdf](#), Kap. "ElsterDatenabholung"
- [ERiC-Entwicklerhandbuch.pdf](#), Kap. "Größenbegrenzung der Eingangsdaten"
- [ERiC-Entwicklerhandbuch.pdf](#), Kap. "Funktionen für Fortschrittcallbacks"
- [EricMtHoleFehlerText\(\)](#)
- [EricMtGetErrormessagesFromXMLAnswer\(\)](#)
- [EricMtRegistriereFortschrittCallback\(\)](#)
- [EricMtRegistriereGlobalenFortschrittCallback\(\)](#)

[ERICAPI_IMPORT](#) int **[EricMtChangePassword](#)** (**[EricInstanzHandle](#)** instanz, const **[byteChar](#)** * psePath, const **[byteChar](#)** * oldPin, const **[byteChar](#)** * newPin)

Die PIN für ein clientseitig erzeugtes Zertifikat (CEZ) wird geändert.

Die Funktion ändert die bei der Funktion [EricMtCreateKey\(\)](#) angegebene PIN und entsprechend hierfür die Prüfsumme in der Datei `eric.sfv`. Falls die Datei `eric.sfv` nicht vorhanden ist, wird sie, wie bei [EricMtCreateKey\(\)](#), erstellt. Eine PIN-Änderung von einem Portalzertifikat (POZ) ist nicht möglich.

Pfade müssen auf Windows in der für Datei-Funktionen benutzten ANSI-Codepage, auf Linux, AIX und Linux Power in der für das Dateisystem benutzten Locale und auf macOS in der "decomposed form" von UTF-8 übergeben werden. Bitte weitere Betriebssystemspezifika bzgl. nicht erlaubter Zeichen in Pfaden und Pfadtrennzeichen beachten.

Für Details zu Pfaden im ERiC siehe [ERiC-Entwicklerhandbuch.pdf](#), Kapitel "Übergabe von Pfaden an ERiC API-Funktionen"

Parameter:

in	<i>instanz</i>	Die ERiC-Instanz, auf der diese Funktion ausgeführt werden soll.
in	<i>psePath</i>	In dem angegebenen Pfad liegt das Schlüsselpaar <code>eric_private.p12</code> und <code>eric_public.cer</code>
in	<i>oldPin</i>	Bisherige PIN.
in	<i>newPin</i>	Neue PIN. Die Mindestlänge beträgt 4 Stellen. Zulässige Zeichen sind alle ASCII-Zeichen ohne die Steuerzeichen.

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_UNGUELTIGER_PARAMETER](#)
- [ERIC_GLOBAL_UNGUELTIGER_PARAMETER](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)
- [ERIC_CRYPT_PIN_STAERKE_NICHT_AUSREICHEND](#)
- [ERIC_CRYPT_PIN_ENTHAELT_UNGUELTIGE_ZEICHEN](#)
- [ERIC_CRYPT_E_PSE_PATH](#)
- [ERIC_CRYPT_NICHT_UNTERSTUETZTES_PSE_FORMAT](#)
- [ERIC_CRYPT_ERROR_CREATE_KEY](#)

Siehe auch:

- [EricMtCreateKey\(\)](#)
- [ERiC-Entwicklerhandbuch.pdf](#), Kap. "Zuordnung der API-Funktionen zur

Verwendung von POZ, CEZ und AHZ"

ERICAPI_IMPORT int EricMtCheckXML ([EricInstanzHandle](#) instanz, const char * xml, const char * datenartVersion, [EricRueckgabepufferHandle](#) fehlertextPuffer)

Das xml wird gegen das Schema der datenartVersion validiert.

Das verwendete Schema kann nachgeschlagen werden unter [Dokumentation\Schnittstellenbeschreibungen\](#)

Nicht unterstützte Datenartversionen:

- ElsterKMV
- alle Bilanz Datenartversionen

Parameter:

in	<i>instanz</i>	Die ERiC-Instanz, auf der diese Funktion ausgeführt werden soll.
in	<i>xml</i>	XML-Zeichenfolge
in	<i>datenartVersion</i>	Die datenartVersion ist der Datenartversionmatrix.xml zu entnehmen. Dieser Parameter darf nicht NULL sein und muss zu den XML-Eingangsdaten passen. Siehe auch ERiC-Entwicklerhandbuch.pdf .
out	<i>fehlertextPuffer</i>	Handle auf einen Rückgabepuffer, in den Fehlertexte geschrieben werden. Zur Erzeugung, Verwendung und Freigabe von Rückgabepuffern siehe Dokumentation zu EricRueckgabepufferHandle .

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_FUNKTION_NICHT_UNTERSTUETZT](#):
Schemavalidierung wird für die übergebene datenartVersion nicht unterstützt.
- [ERIC_GLOBAL_COMMONDATA_NICHT_VERFUEGBAR](#)
- [ERIC_GLOBAL_DATENARTVERSION_UNBEKANNT](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_IO_READER_SCHEMA_VALIDIERUNGSFEHLER](#):
Die Fehlerbeschreibung steht im fehlertextPuffer .
- [ERIC_IO_PARSE_FEHLER](#):
Die Fehlerbeschreibung steht im fehlertextPuffer .
- weitere, siehe [eric_fehlercodes.h](#)

ERICAPI_IMPORT int **EricMtCloseHandleToCertificate** ([EricInstanzHandle](#) instanz, [EricZertifikatHandle](#) hToken)

Das Zertifikat-Handle `hToken` wird freigegeben.

Diese Funktion gibt das übergebene Zertifikat-Handle frei. Zertifikat-Handles sollten möglichst frühzeitig, d.h. wenn sie nicht mehr benötigt werden, mit [EricMtCloseHandleToCertificate\(\)](#) freigegeben werden, spätestens jedoch zum Programmende bzw. vor dem Entladen der ericapi Bibliothek.

Das Ad Hoc-Zertifikat eines neuen Personalausweises sollte immer genau dann freigegeben werden, wenn es nicht mehr benötigt wird, jedoch spätestens vor Ablauf der 24 Stunden, die das Ad Hoc-Zertifikat gültig ist.

Tritt ein Fehler auf, kann die Fehlermeldung mit [EricMtHoleFehlerText\(\)](#) ausgelesen werden.

Parameter:

in	<i>instanz</i>	Die ERiC-Instanz, auf der diese Funktion ausgeführt werden soll.
in	<i>hToken</i>	Zertifikat-Handle wie von der Funktion EricMtGetHandleToCertificate() zurückgeliefert.

Rückgabe:

- [ERIC_OK](#)
- [ERIC_CRYPT_E_INVALID_HANDLE](#)
- [ERIC_GLOBAL_UNGUELTIGER_PARAMETER](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)

Zu beachten:

Die folgenden Rückgabewerte gelten nur bei Verwendung des neuen Personalausweises.

Rückgabe:

- [ERIC_TRANSFER_EID_CLIENTFEHLER](#)
- [ERIC_TRANSFER_EID_FEHLENDEFELDER](#)
- [ERIC_TRANSFER_EID_IDENTIFIKATIONABGEBROCHEN](#)
- [ERIC_TRANSFER_EID_NPABLOCKIERT](#)
- [ERIC_TRANSFER_EID_IDNRNICHTEINDEUTIG](#)
- [ERIC_TRANSFER_EID_KEINCLIENT](#)
- [ERIC_TRANSFER_EID_KEINKONTO](#)
- [ERIC_TRANSFER_EID_SERVERFEHLER](#)
- [ERIC_TRANSFER_ERR_CONNECTSERVER](#)

- [ERIC_TRANSFER_ERR_NORESPONSE](#)
- [ERIC_TRANSFER_ERR_PROXYAUTH](#)
- [ERIC_TRANSFER_ERR_PROXYCONNECT](#)
- [ERIC_TRANSFER_ERR_SEND](#)
- [ERIC_TRANSFER_ERR_SEND_INIT](#)
- [ERIC_TRANSFER_ERR_TIMEOUT](#)

Siehe auch:

- [EricMtGetHandleToCertificate\(\)](#)
- [EricMtZertifikatOeffnenAusBytes\(\)](#)
- [EricMtGetPinStatus\(\)](#)
- [ERiC-Entwicklerhandbuch.pdf](#), Kap. "Authentifizierung mit dem neuen Personalausweis (nPA)"

ERICAPI_IMPORT int EricMtCreateKey ([EricInstanzHandle](#) instanz, const [byteChar](#) * pin, const [byteChar](#) * pfad, const [eric_zertifikat_parameter_t](#) * zertifikatInfo)

Es werden die Kryptomittel für ein clientseitig erzeugtes Zertifikat (CEZ) in einem Verzeichnis des Dateisystems erstellt.

Im angegebenen Verzeichnis `pfad` sind nach Ausführung der Funktion [EricMtCreateKey\(\)](#) drei Dateien erstellt worden:

- `eric_public.cer`:
Enthält das Zertifikat mit den Daten aus `zertifikatInfo` und darin den öffentlichen Schlüssel.
- `eric_private.pl2`:
Enthält den privaten Schlüssel. Der Zugriff ist über die `pin` geschützt.
- `eric.sfv`:
Enthält die Prüfsumme der Dateien `eric_public.cer` und `eric_private.pl2`. Die Integrität dieser beiden Dateien kann damit jederzeit überprüft werden.

Ein CEZ kann unter anderem für die Bescheiddaten-Rückübermittlung verwendet werden. Weitere Informationen zur Datenabholung lesen Sie bitte im [ERiC-Entwicklerhandbuch.pdf](#) nach.

Parameter:

in	<i>instanz</i>	Die ERiC-Instanz, auf der diese Funktion ausgeführt werden soll.
in	<i>pin</i>	PIN (Passwort), mit der auf den privaten Schlüssel zugegriffen werden kann. Die Mindestlänge beträgt 4 Stellen. Zulässige Zeichen sind alle ASCII-Zeichen ohne die Steuerzeichen.
in	<i>pfad</i>	Pfad (1), in dem die Kryptomittel erzeugt werden sollen. Das durch den angegebenen Pfad bezeichnete Verzeichnis muss im Dateisystem bereits existieren und beschreibbar sein. Es gibt folgende Möglichkeiten: <ul style="list-style-type: none"> ◦ Absoluter Pfad: Empfehlung. ◦ Relativer Pfad: Wird an das Arbeitsverzeichnis angehängt. ◦ Leere Zeichenkette: In diesem Fall wird das Arbeitsverzeichnis verwendet.
in	<i>zertifikatInfo</i>	Daten, die zur Identifikation des Schlüsselinhabers im Zertifikat abgelegt werden.

(1) Pfade müssen auf Windows in der für Datei-Funktionen benutzten ANSI-Codepage, auf Linux, AIX und Linux Power in der für das Dateisystem benutzten Locale und auf macOS in der "decomposed form" von UTF-8 übergeben werden. Bitte weitere Betriebssystemspezifika bzgl. nicht erlaubter Zeichen in Pfaden und Pfadtrennzeichen beachten.

Für Details zu Pfaden im ERiC siehe [ERiC-Entwicklerhandbuch.pdf](#), Kapitel "Übergabe von Pfaden an ERiC API-Funktionen".

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_UNGUELTIGE_PARAMETER_VERSION](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)
- [ERIC_CRYPT_ZERTIFIKATSPFAD_KEIN_VERZEICHNIS](#)
- [ERIC_CRYPT_ZERTIFIKATSDATEI_EXISTIERT_BEREITS](#)
- [ERIC_CRYPT_PIN_STAERKE_NICHT_AUSREICHEND](#)
- [ERIC_CRYPT_PIN_ENTHAELT_UNGUELTIGE_ZEICHEN](#)
- [ERIC_CRYPT_ERROR_CREATE_KEY](#)

Siehe auch:

- [EricMtChangePassword\(\)](#)
- [ERiC-Entwicklerhandbuch.pdf](#), Kap. "Zertifikate und Authentifizierungsverfahren"
- [ERiC-Entwicklerhandbuch.pdf](#), Kap. "Übergabe von Pfaden an ERiC API-Funktionen"

[ERICAPI_IMPORT](#) int EricMtCreateTH ([EricInstanzHandle](#) instanz, const char * xml, const char * verfahren, const char * datenart, const char * vorgang, const char * testmerker, const char * herstellerId, const char * datenLieferant, const char * versionClient, const [byteChar](#) * publicKey, [EricRueckgabepufferHandle](#) xmlRueckgabePuffer)

Diese Funktion erzeugt einen TransferHeader.

Dieser ist der oberste Header in der Datenstruktur. Er enthält Felder für die Kommunikation zwischen Server und Client. Es wird nur die Kombination NutzdatenHeader-Version "11" und TransferHeader-Version "11" unterstützt.

Zur Erzeugung, Verwendung und Freigabe von Rückgabepuffern siehe Dokumentation zu [EricRueckgabepufferHandle](#).

Parameter:

in	<i>instanz</i>	Die ERiC-Instanz, auf der diese Funktion ausgeführt werden soll.
in	<i>xml</i>	<p>XML-Datensatz, für den der <TransferHeader>-Block erzeugt werden soll.</p> <p>Es kann entweder ein komplettes Elster-XML oder nur der Datenteil übergeben werden.</p> <p>ERiC nimmt bei diesem Parameter keine Konvertierung von Sonderzeichen in Entitätenreferenzen vor.</p> <p>Attribute, die in den Start-Tags der Elemente <Elster> bzw. <DatenTeil> im übergebenen XML-Datensatz definiert werden, werden nicht in das Rückgabe-XML übernommen.</p> <p>Namespace-Definitionen, die in den Start-Tags der Elemente <Elster> bzw. <DatenTeil> im übergebenen XML-Datensatz definiert werden, führen zu einem ERIC IO PARSE FEHLER.</p> <p>Im Rückgabe-XML werden im Start-Tag des Elements <Elster> die URI "http://www.elster.de/elsterxml/schema/v11" als Default-Namensraum definiert.</p> <p>Die dem Element <DatenTeil> untergeordneten Elemente aus dem übergebenen XML-Datensatz werden unverändert übernommen.</p> <p>Der allgemeine Aufbau des Elster-XMLs wird im ERiC-Entwicklerhandbuch.pdf, Kap. "Datenverarbeitung mit ERiC" beschrieben.</p>

in	<i>verfahren</i>	Name des Verfahrens, z.B: 'ElsterAnmeldung', siehe ERiC-Entwicklerhandbuch.pdf , Tabelle "Eigenschaften der Datenart" im jeweiligen Kapitel zur Datenart.
in	<i>datenart</i>	Name der Datenart, z.B.: 'LStB' oder 'UStVA', siehe ERiC-Entwicklerhandbuch.pdf , Tabelle "Eigenschaften der Datenart" im jeweiligen Kapitel zur Datenart.
in	<i>vorgang</i>	Name der Übertragungsart, z.B. 'send-NoSig', siehe ERiC-Entwicklerhandbuch.pdf , Tabelle "Eigenschaften der Datenart" im jeweiligen Kapitel zur Datenart.
in	<i>testmerker</i>	Für eine Testübertragung muss der entsprechende Testmerker angegeben werden, siehe ERiC-Entwicklerhandbuch.pdf , Kap. "Test Unterstützung bei der ERiC-Anbindung". Falls ein Echtfall übertragen werden soll, muss der Wert NULL angegeben werden.
in	<i>herstellerId</i>	Hersteller-ID des Softwareproduktes.
in	<i>datenLieferant</i>	Der Wert entspricht dem XML-Element <DatenLieferant>, wie es im Schema des Transferheaders der ElsterBasis-XML-Schnittstelle definiert ist. ERiC konvertiert bei diesem Parameter Sonderzeichen in Entitätenreferenzen.
in	<i>versionClient</i>	Angabe von Versionsinformation, die in der Serverantwort auch zurückgegeben wird und ausgewertet werden kann. Der Wert NULL entspricht "keine Angabe von Versionsinformation", d.h. es wird kein XML-Element <VersionClient> im <TransferHeader>-Block erzeugt. ERiC konvertiert bei diesem Parameter Sonderzeichen in Entitätenreferenzen.
in	<i>publicKey</i>	Öffentlicher Schlüssel für die Transportverschlüsselung beim Verfahren ElsterLohn. Bei anderen Verfahren sollte NULL übergeben werden. Dieser Wert kann mit dem Rückgabewert von EricMtGetPublicKey() befüllt werden. Der Inhalt dieses Parameters wird in das <TransportSchluessel>-Element der Rückgabe-XML geschrieben.

out	<i>xmlRueckgabePuffer</i>	Handle auf einen Rückgabepuffer, in den das Elster-XML mit dem erzeugten TransportHeader geschrieben wird, siehe EricRueckgabepufferHandle . Es wird immer ein vollständiger Elster-XML-Datensatz mit dem <Elster>-Element als Wurzel-Element zurückgeliefert. Bzgl. der darin enthaltenen XML-Namespace-Definitionen sind die bei der Beschreibung des Parameters "xml" genannten Einschränkungen zu berücksichtigen.
-----	---------------------------	---

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_TRANSFER_ERR_XML_ENCODING](#): Die übergebenen XML-Daten sind nicht UTF-8 kodiert.
- [ERIC_IO_PARSE_FEHLER](#)
- [ERIC_IO_DATENTEILNOTFOUND](#)
- [ERIC_IO_DATENTEILENDNOTFOUND](#)
- weitere, siehe [eric_fehlercodes.h](#)

Siehe auch:

- [ERiC-Entwicklerhandbuch.pdf](#), Kap. "Datenverarbeitung mit ERiC"
- [ERiC-Entwicklerhandbuch.pdf](#), Kap. "Anwendungsfälle von EricBearbeiteVorgang()"
- ERiC-Returncodes und Fehlertexte sind in [eric_fehlercodes.h](#) zu finden.

[ERICAPI_IMPORT](#) int EricMtCreateUUID ([EricInstanzHandle](#) instanz,
[EricRueckgabepufferHandle](#) uuidRueckgabePuffer)

Erzeugt einen Version 4 Universally Unique Identifier (UUID) gemäß RFC 4122.

Parameter:

in	<i>instanz</i>	Die ERiC-Instanz, auf der diese Funktion ausgeführt werden soll.
out	<i>uuidRueckgabe Puffer</i>	Handle auf einen Rückgabepuffer, in den die erzeugte UUID geschrieben wird.

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)

[ERICAPI_IMPORT](#) int EricMtDekodiereDaten ([EricInstanzHandle](#) instanz, [EricZertifikatHandle](#) zertifikatHandle, const [byteChar](#) * pin, const [byteChar](#) * base64Eingabe, [EricRueckgabepufferHandle](#) rueckgabePuffer)

Es werden die mit der Datenabholung abgeholt und verschlüsselten Daten entschlüsselt.

Falls während der Bearbeitung ein Fehler auftritt, liefert die Funktion [EricMtHoleFehlerText\(\)](#) den dazugehörigen Fehlertext.

Parameter:

in	<i>instanz</i>	Die ERIC-Instanz, auf der diese Funktion ausgeführt werden soll.
in	<i>zertifikatHandle</i>	Handle auf das zum Entschlüsseln zu verwendende Zertifikat.
in	<i>pin</i>	PIN zum Zugriff auf das Zertifikat.
in	<i>base64Eingabe</i>	Base64-kodierte verschlüsselte Daten oder Anhänge, welche mit dem Verfahren ElsterDatenabholung abgeholt wurden. Die Abholdaten befinden sich im Element <code>/Elster[1]/DatenTeil[1]/Nutzdatenblock/Nutzdaten[1]/Datenabholung[1]/Abholung[1]/Datenpaket</code> Die optionalen Anhänge befinden sich im Element <code>/Elster[1]/DatenTeil[1]/Nutzdatenblock/Nutzdaten[1]/Datenabholung[1]/Abholung[1]/Anhaenge[1]/Anhang[1]/Dateiinhalte</code>
out	<i>rueckgabePuffer</i>	Handle auf einen Rückgabepuffer, in den die entschlüsselten Daten geschrieben werden. Im Fehlerfall ist der Inhalt des Rückgabepuffers undefiniert. Zur Erzeugung, Verwendung und Freigabe von Rückgabepuffern siehe EricRueckgabepufferHandle .

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_ERR_DEKODIEREN](#)
- [ERIC_GLOBAL_UNKNOWN](#)
- Ein Zertifikatsfehler aus dem Statuscodebereich von [ERIC_CRYPT_E_INVALID_HANDLE](#) = 610201101 bis 610201212

Siehe auch:

- [EricMthHoleFehlerText\(\)](#)
- [ERiC-Entwicklerhandbuch.pdf](#), Kap. "ElsterDatenabholung"

ERICAPI_IMPORT int EricMtEinstellungAlleZuruecksetzen (EricInstanzHandle instanz)

Alle Einstellungen, der übergebenen ERiC-Instanz werden auf den jeweiligen Standardwert zurückgesetzt.

Die Standardwerte sind im Dokument [ERiC-Entwicklerhandbuch.pdf](#), Kap. "Vorbelegung der ERiC-Einstellungen" zu finden.

Parameter:

in	<i>instanz</i>	Die ERiC-Instanz, auf der diese Funktion ausgeführt werden soll.
----	----------------	--

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_UNKNOWN](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNGUELTIGER_PARAMETER](#)

Siehe auch:

- [EricMtEinstellungSetzen\(\)](#)
- [EricMtEinstellungLesen\(\)](#)
- [EricMtEinstellungZuruecksetzen\(\)](#)
- [ERiC-Entwicklerhandbuch.pdf](#), Kap. "Bedeutung der ERiC-Einstellungen"

[ERICAPI_IMPORT](#) int EricMtEinstellungLesen ([EricInstanzHandle](#) instanz, const char * name, [EricRueckgabepufferHandle](#) rueckgabePuffer)

Der Wert der API-Einstellung `name` wird im `rueckgabePuffer` zurückgeliefert.

Parameter:

in	<i>instanz</i>	Die ERiC-Instanz, auf der diese Funktion ausgeführt werden soll.
in	<i>name</i>	Name der API-Einstellung, NULL-terminierte Zeichenfolge.
out	<i>rueckgabePuffer</i>	Handle auf einen Rückgabepuffer, in den der Wert der API-Einstellung geschrieben wird. Zur Erzeugung, Verwendung und Freigabe von Rückgabepuffern siehe EricRueckgabepufferHandle .

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_EINSTELLUNG_NAME_UNGUELTIG](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)

Siehe auch:

- [EricMtEinstellungSetzen\(\)](#)
- [EricMtEinstellungZuruecksetzen\(\)](#)
- [EricMtEinstellungAlleZuruecksetzen\(\)](#)
- [ERiC-Entwicklerhandbuch.pdf](#), Kap. "Bedeutung der ERiC-Einstellungen"

[ERICAPI_IMPORT](#) int EricMtEinstellungSetzen ([EricInstanzHandle](#) instanz, const char * name, const char * wert)

Die API-Einstellung `name` wird auf den `wert` gesetzt.

Nach dem Laden der ERiC-Bibliotheken hat jede API-Einstellung ihren Standardwert. Mit dieser Funktion kann der Wert verändert werden. Der Wertebereich der jeweiligen API-Einstellung ist zu beachten.

Bei Pfad-Einstellungen muss auf Windows der Wert in der für Datei-Funktionen benutzten ANSI-Codepage, auf Linux, AIX und Linux Power in der für das Dateisystem benutzten Locale und auf macOS in der "decomposed form" von UTF-8 übergeben werden. Bitte weitere Betriebssystemspezifika bzgl. nicht erlaubter Zeichen in Pfaden und Pfadtrennzeichen beachten.

Für Details zu Pfaden im ERiC siehe [ERiC-Entwicklerhandbuch.pdf](#), Kapitel "Übergabe von Pfaden an ERiC API-Funktionen"

Parameter:

in	<i>instanz</i>	Die ERiC-Instanz, auf der diese Funktion ausgeführt werden soll.
in	<i>name</i>	Name der API-Einstellung, NULL-terminierte Zeichenfolge.
in	<i>wert</i>	Wert der API-Einstellung, NULL-terminierte Zeichenfolge.

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_EINSTELLUNG_NAME_UNGUELTIG](#)
- [ERIC_GLOBAL_EINSTELLUNG_WERT_UNGUELTIG](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)

Siehe auch:

- [EricMtEinstellungLesen\(\)](#)
- [EricMtEinstellungZuruecksetzen\(\)](#)
- [EricMtEinstellungAlleZuruecksetzen\(\)](#)
- [ERiC-Entwicklerhandbuch.pdf](#), Kap. "Bedeutung der ERiC-Einstellungen"

[ERICAPI_IMPORT](#) int EricMtEinstellungZuruecksetzen ([EricInstanzHandle](#) instanz, const char * name)

Der Wert der API-Einstellung `name` wird auf den Standardwert zurückgesetzt.

Die Standardwerte sind im Dokument [ERiC-Entwicklerhandbuch.pdf](#), Kap. "Vorbelegung der ERiC-Einstellungen" zu finden.

Parameter:

in	<i>instanz</i>	Die ERiC-Instanz, auf der diese Funktion ausgeführt werden soll.
in	<i>name</i>	Name der API-Einstellung, NULL-terminierte Zeichenfolge.

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_EINSTELLUNG_NAME_UNGUELTIG](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)

Siehe auch:

- [EricMtEinstellungSetzen\(\)](#)
- [EricMtEinstellungLesen\(\)](#)
- [EricMtEinstellungAlleZuruecksetzen\(\)](#)
- [ERiC-Entwicklerhandbuch.pdf](#), Kap. "Bedeutung der ERiC-Einstellungen"

ERICAPI_IMPORT int EricMtEntladePlugins (EricInstanzHandle instanz)

Für die übergebene ERiC-Instanz werden alle verwendeten Plugin-Bibliotheken entladen und deren Speicher wird freigegeben.

Der ERiC lädt die für die Bearbeitung notwendigen Plugin-Bibliotheken permanent in den Speicher und gibt diese erst mit dem Aufruf dieser Funktion wieder frei.

Zu beachten:

EricMtEntladePlugins() sollte erst dann aufgerufen werden, wenn die Plugin-Bibliotheken definitiv nicht mehr benötigt werden. Ein erneutes Laden der Bibliotheken ist verhältnismäßig zeitintensiv.

Falls eine Plugin-Bibliothek nicht entladen werden kann, wird dies in eric.log protokolliert. Der Returncode ist immer ERIC_OK.

Parameter:

in	<i>instanz</i>	Die ERiC-Instanz, auf der diese Funktion ausgeführt werden soll.
----	----------------	--

Rückgabe:

- ERIC_OK
- ERIC_GLOBAL_UNGUELTIGER_PARAMETER
- ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER
- ERIC_GLOBAL_UNKNOWN

Siehe auch:

- ERiC-Entwicklerhandbuch.pdf, Kap. "Verwendung von EricEntladePlugins()"

ERICAPI_IMPORT int EricMtFormatEWAZ (EricInstanzHandle instanz, const byteChar * ewAzElster, EricRueckgabepufferHandle ewAzBescheidPuffer)

Konvertiert ein Einheitswert-Aktenzeichen im ELSTER-Format in ein landesspezifisches Bescheidformat.

Konvertiert ein Einheitswert-Aktenzeichen im ELSTER-Format (z.B. 2831400190001250002) in ein landesspezifisches Einheitswert-Aktenzeichen im Bescheidformat (z.B. 3100190001250002).

Parameter:

in	<i>instanz</i>	Die ERiC-Instanz, auf der diese Funktion ausgeführt werden soll.
in	<i>ewAzElster</i>	Zeiger auf ein Einheitswert-Aktenzeichen im ELSTER-Format (z.B. 2831400190001250002)
out	<i>ewAzBescheidPuffer</i>	Handle auf einen Rückgabepuffer, in den das Einheitswert-Aktenzeichen im Bescheidformat (z.B. 3100190001250002) geschrieben wird. Zur Erzeugung, Verwendung und Freigabe von Rückgabepuffern siehe Dokumentation zu EricRueckgabepufferHandle .

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_EWAZ_UNGUELTIG](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)

[ERICAPI_IMPORT](#) int EricMtFormatStNr ([EricInstanzHandle](#) instanz, const [byteChar](#) *
eingabeSteuernummer, [EricRueckgabepufferHandle](#) rueckgabePuffer)

Die Steuernummer `eingabeSteuernummer` wird in das Bescheid-Format des jeweiligen Bundeslandes umgewandelt.

Parameter:

in	<i>instanz</i>	Die ERiC-Instanz, auf der diese Funktion ausgeführt werden soll.
in	<i>eingabeSteuernummer</i>	Gültige, zu formatierende Steuernummer im ELSTER-Steuernummernformat.
out	<i>rueckgabePuffer</i>	Handle auf einen Rückgabepuffer, in den die formatierte Steuernummer im Bescheid-Format des jeweiligen Bundeslandes geschrieben wird. Zur Erzeugung, Verwendung und Freigabe von Rückgabepuffern siehe Dokumentation zu EricRueckgabepufferHandle .

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_STEUERNUMMER_UNGUELTIG](#)
- [ERIC_GLOBAL_COMMONDATA_NICHT_VERFUEGBAR](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)

Siehe auch:

- [Pruefung_der_Steuer_und_Steueridentifikatsnummer.pdf](#), siehe [Entwicklerbereich](#) bei [ELSTER](#).

ERICAPI_IMPORT int EricMtGetAuswahlListen ([EricInstanzHandle](#) instanz, const char * datenartVersion, const char * feldkennung, [EricRueckgabepufferHandle](#) rueckgabeXmlPuffer)

Die Auswahlliste(n) für datenartVersion oder feldkennung wird zurückgeliefert.

Anwendungsfälle:

1. Parameter feldkennung ist nicht NULL:
Die Funktion liefert die zur feldkennung und datenartVersion gehörige Auswahlliste.
2. Parameter feldkennung ist NULL:
Die Funktion liefert alle zur datenartVersion gehörigen Feldkennungen mit hinterlegten Auswahllisten.

Für die Ermittlung der Auswahllisten vieler Feldkennungen wird aus Performanzgründen Anwendungsfall 2 empfohlen. Die Funktion liefert Auswahllisten zu Feldkennungen vom Format "NichtAbgeschlosseneEnumeration" zurück. Diese Auswahllisten werden auch in der Jahres-/Deltadokumentation dokumentiert.

Parameter:

in	<i>instanz</i>	Die ERiC-Instanz, auf der diese Funktion ausgeführt werden soll.
in	<i>datenartVersion</i>	Dieser Parameter darf nicht NULL sein. Die gültigen Datenartversionen sind in der Datenartversionmatrix.xml enthalten.
in	<i>feldkennung</i>	Feldkennung, für welche die Auswahlliste zu ermitteln ist.
out	<i>rueckgabeXmlPuffer</i>	Handle auf einen Rückgabepuffer, in den die angeforderten Auswahlliste(n) als XML-Daten geschrieben werden. Die XML-Daten folgen der XML Schema Definition in Dokumentation\API-Rueckgabe-Schemata\EricGetAuswahlListen.xsd . Zur Erzeugung, Verwendung und Freigabe von Rückgabepuffern siehe EricRueckgabepufferHandle .

Beispiel:

```
<?xml version="1.0" encoding="UTF-8"?>
<EricGetAuswahlListen
xmlns="http://www.elster.de/EricXML/1.0/EricGetAuswahlListen">
  <AuswahlListe>
    <Feldkennung>0104110</Feldkennung>
    <ListenElement>Arbeitslosengeld</ListenElement>
```

```
<ListenElement>Elterngeld</ListenElement>
<ListenElement>Insolvenzgeld</ListenElement>
<ListenElement>Krankengeld</ListenElement>
<ListenElement>Mutterschaftsgeld</ListenElement>
</AuswahlListe>
</EricGetAuswahlListen>
```

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_KEINE_DATEN_VORHANDEN](#)
- [ERIC_GLOBAL_DATENARTVERSION_UNBEKANNT](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)

[ERICAPI_IMPORT](#) int EricMtGetErrorMessageFromXMLAnswer ([EricInstanzHandle](#) instanz, const char * xml, [EricRueckgabepufferHandle](#) transferticketPuffer, [EricRueckgabepufferHandle](#) returncodeTHPuffer, [EricRueckgabepufferHandle](#) fehlertextTHPuffer, [EricRueckgabepufferHandle](#) returncodesUndFehlertexteNDHXmlPuffer)

Aus dem Antwort-XML des Finanzamtsservers wird das Transferticket und Returncodes/Fehlermeldungen zurückgegeben.

Die Funktion liefert bei erfolgreicher Ausführung:

- Das Transferticket aus dem Antwort-XML in dem Parameter `transferticketPuffer`.
- Den Returncode und die Fehlermeldung aus dem Transferheader in den Parametern `returncodeTHPuffer` und `fehlertextTHPuffer`.
- Für jeden Nutzdatenheader dessen Returncode und Fehlermeldung als XML-Daten im Parameter `returncodesUndFehlertexteNDHXmlPuffer` nach XML Schema Definition [Dokumentation\API-Rueckgabe-Schemata\EricGetErrorMessageFromXMLAnswer.xsd](#). Enthält das Antwort-XML keine Nutzdaten, wird kein <Fehler> Element zurückgegeben.

Zur Erzeugung, Verwendung und Freigabe von Rückgabepuffern siehe Dokumentation zu [EricRueckgabepufferHandle](#).

Parameter:

in	<i>instanz</i>	Die ERiC-Instanz, auf der diese Funktion ausgeführt werden soll.
in	<i>xml</i>	Antwort-XML des ELSTER-Servers, das ausgewertet werden soll. Der originale XML-Server-Datenstrom sollte unverändert übergeben werden und darf insbesondere keine Zeilenumbruchzeichen enthalten.
out	<i>transferticketPuffer</i>	Handle auf einen Rückgabepuffer, in den das Transferticket geschrieben wird, siehe EricRueckgabepufferHandle .
out	<i>returncodeTHPuffer</i>	Handle auf einen Rückgabepuffer, in den der Returncode aus dem Transferheader geschrieben wird. Siehe EricRueckgabepufferHandle .
out	<i>fehlertextTHPuffer</i>	Handle auf einen Rückgabepuffer, in den die Fehlermeldung aus dem Transferheader geschrieben wird, siehe EricRueckgabepufferHandle .

out	<i>returncodesUnd FehlertexteNDH XmlPuffer</i>	Handle auf einen Rückgabepuffer, in den die Liste der Returncodes nach XML-Schema Dokumentation\API-Rueckgabe-Schemata\EricGetErrorMessageFromXMLAnswer.xsd geschrieben werden, siehe EricRueckgabepufferHandle .
-----	--	---

Beispiel:

```
<?xml version="1.0" encoding="UTF-8"?>
<EricGetErrorMessageFromXMLAnswer
xmlns="http://www.elster.de/EricXML/1.0/EricGetErrorMessageFromXMLAnswer">
  <Fehler>
    <Code>1</Code>
    <Meldung>Fehlermeldung 1</Meldung>
  </Fehler>
  <Fehler>
    <Code>2</Code>
    <Meldung>Fehlermeldung 2</Meldung>
  </Fehler>
  (...)
</EricGetErrorMessageFromXMLAnswer>
```

Rückgabe:

- [ERIC_OK](#)
- [ERIC_IO_PARSE_FEHLER](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_PUFFER_ZUGRIFFSKONFLIKT](#)
- [ERIC_GLOBAL_NICHT_GENUEGENDE_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)

Zu beachten:

Diese Funktion kann nicht dafür verwendet werden, die Antwort im Datenteil aus einer dekodierten Serverantwort für Lohnsteuerbescheinigungen auszuwerten.

Siehe auch:

- XML-Schema des Transferheaders:
[Dokumentation\Schnittstellenbeschreibungen\ElsterBasisSchema\Schema\th000011_extern.xsd](#)
- XML-Schema des Nutzdatenheaders:
[Dokumentation\Schnittstellenbeschreibungen\ElsterBasisSchema\Schema\ndh000011.xsd](#)
- [ERiC-Entwicklerhandbuch.pdf](#), Kap. "Schnittstellenbeschreibungen", Tabelle "Ergänzende Softwarepakete und Dateien – Schnittstellenbeschreibungen"

ERICAPI_IMPORT int EricMtGetHandleToCertificate ([EricInstanzHandle](#) instanz, [EricZertifikatHandle](#) * hToken, [uint32_t](#) * iInfoPinSupport, const [byteChar](#) * pathToKeystore)

Für das übergebene Zertifikat in `pathToKeystore` wird das Handle `hToken` und die unterstützten PIN-Werte `iInfoPinSupport` zurückgeliefert.

Die ERiC API benötigt Zertifikat-Handles typischerweise bei kryptografischen Operationen.

Zertifikat-Handles sollten möglichst frühzeitig, d.h. wenn sie nicht mehr benötigt werden, mit [EricMtCloseHandleToCertificate\(\)](#) freigegeben werden, spätestens jedoch zum Programmende bzw. vor dem Entladen der ericapi Bibliothek.

Parameter:

in	<i>instanz</i>	Die ERiC-Instanz, auf der diese Funktion ausgeführt werden soll.
out	<i>hToken</i>	Handle zu einem der folgenden Zertifikate: <ul style="list-style-type: none"> ◦ Portalzertifikat ◦ clientseitig erzeugtes Zertifikat ◦ Ad Hoc-Zertifikat für den neuen Personalausweis

out	<i>iInfoPinSupport</i>	<p>Wird in <code>iInfoPinSupport</code> ein Zeiger ungleich NULL übergeben und die Funktion mit ERIC_OK beendet, dann enthält <code>iInfoPinSupport</code> einen vorzeichenlosen Integer-Wert.</p> <p>In diesem Wert ist kodiert abgelegt, ob eine PIN-Eingabe erforderlich ist und welche PIN-Statusinformationen unterstützt werden.</p> <p>Die kodierten Werte (nachfolgend in hexadezimaler Form angegeben) können durch ein binäres ODER kombiniert werden und bedeuten im Einzelnen:</p> <ul style="list-style-type: none"> ◦ <code>0x00</code>: Keine PIN-Angabe erforderlich, kein PIN-Status unterstützt. ◦ <code>0x01</code>: PIN-Angabe für Signatur erforderlich. ◦ <code>0x02</code>: PIN-Angabe für Entschlüsselung erforderlich. ◦ <code>0x04</code>: PIN-Angabe für Verschlüsselung des Zertifikats erforderlich. ◦ <code>0x08</code>: reserviert (wird derzeit nicht verwendet) ◦ <code>0x10</code>: PIN-Status "Pin Ok" wird unterstützt. ◦ <code>0x20</code>: PIN-Status "Der letzte Versuch der Pin-Eingabe schlug fehl" wird unterstützt. ◦ <code>0x40</code>: PIN-Status "Beim nächsten fehlerhaften Versuch wird die Pin gesperrt" wird unterstützt. ◦ <code>0x80</code>: PIN-Status "Pin ist gesperrt" wird unterstützt. <p>Falls vom Aufrufer NULL übergeben wird, gibt die Funktion nichts zurück.</p>
-----	------------------------	---

in	<i>pathToKeystore</i>	<p>Folgende Zertifikatstypen werden unterstützt:</p> <ol style="list-style-type: none"> 1. Clientseitig erzeugtes Zertifikat: Pfad zum Verzeichnis, in dem sich die Zertifikats-Datei (.cer) und die Datei mit dem privaten Schlüssel (.p12) befinden. Diese Kryptomittel wurden mit EricMtCreateKey() erzeugt. Der Pfad zum Verzeichnis ist bei clientseitig erzeugten Zertifikaten relativ zum aktuellen Arbeitsverzeichnis oder absolut anzugeben. 2. Software-Portalzertifikat: Pfad zur Software-Zertifikatsdatei (i.d.R. mit der Endung .pfx). Der Pfad zur Datei ist bei Software-Zertifikaten relativ zum aktuellen Arbeitsverzeichnis oder absolut anzugeben. 3. Sicherheitsstick: Pfad zur Treiberdatei, siehe (1). Bitte beachten, dass der Treiber betriebssystemabhängig sein kann. Weitere Informationen in der Anleitung zum Sicherheitsstick oder unter https://www.sicherheitsstick.de. 4. Signaturkarte: Pfad zur Treiberdatei, welcher einen Zugriff auf die Signaturkarte ermöglicht, siehe (1). Weitere Informationen in der Anleitung zur Signaturkarte. 5. Neuer Personalausweis (nPA): URL des eID-Clients wie zum Beispiel der AusweisApp 2 In den meisten Fällen lautet diese URL: http://127.0.0.1:24727/eID-Client Optional kann auf die folgende Weise noch ein Testmerker angehängt werden: http://127.0.0.1:24727/eID-Client?testmerker=520000000. Zu den verfügbaren Testmerkern siehe ERiC-Entwicklerhandbuch.pdf, Kap. "Test Unterstützung bei der ERiC-Anbindung". <p>Wichtig: Das Ad Hoc-Zertifikat, das in diesem Fall für den neuen Personalausweis erzeugt wird, ist nur 24 Stunden gültig.</p>
----	-----------------------	---

(1) Bei Sicherheitssticks und Signaturkarten ist bei der Angabe des Treibers der Suchmechanismus nach dynamischen Modulen des jeweiligen Betriebssystems

zu berücksichtigen. Weitere Informationen sind z.B. unter Windows der Dokumentation der `LoadLibrary()` oder unter Linux und macOS der Dokumentation der `dlopen()` zu entnehmen.

Pfade müssen auf Windows in der für Datei-Funktionen benutzten ANSI-Codepage, auf Linux, AIX und Linux Power in der für das Dateisystem benutzten Locale und auf macOS in der "decomposed form" von UTF-8 übergeben werden. Bitte weitere Betriebssystemspezifika bzgl. nicht erlaubter Zeichen in Pfaden und Pfadtrennzeichen beachten.

Für Details zu Pfaden im ERiC siehe [ERiC-Entwicklerhandbuch.pdf](#), Kapitel "Übergabe von Pfaden an ERiC API-Funktionen".

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)
- [ERIC_CRYPT_NICHT_UNTERSTUEZTES_PSE_FORMAT](#)
- [ERIC_CRYPT_E_MAX_SESSION](#)
- [ERIC_CRYPT_E_PSE_PATH](#)
- [ERIC_CRYPT_E_BUSY](#)
- [ERIC_CRYPT_E_P11_SLOT_EMPTY](#)
- [ERIC_CRYPT_E_NO_SIG_ENC_KEY](#)
- [ERIC_CRYPT_E_LOAD_DLL](#)
- [ERIC_CRYPT_E_NO_SERVICE](#)
- [ERIC_CRYPT_E_ESICL_EXCEPTION](#)

Zu beachten:

Die folgenden Rückgabewerte gelten nur bei Verwendung des neuen Personalausweises.

Rückgabe:

- [ERIC_TRANSFER_EID_CLIENTFEHLER](#)
- [ERIC_TRANSFER_EID_FEHLENDEFELDER](#)
- [ERIC_TRANSFER_EID_IDENTIFIKATIONABGEBROCHEN](#)
- [ERIC_TRANSFER_EID_NPABLOCKIERT](#)
- [ERIC_TRANSFER_EID_IDNRNICHTEINDEUTIG](#)
- [ERIC_TRANSFER_EID_KEINCLIENT](#)
- [ERIC_TRANSFER_EID_KEINKONTO](#)
- [ERIC_TRANSFER_EID_SERVERFEHLER](#)

- [ERIC_TRANSFER_ERR_CONNECTSERVER](#)
- [ERIC_TRANSFER_ERR_NORESPONSE](#)
- [ERIC_TRANSFER_ERR_PROXYAUTH](#)
- [ERIC_TRANSFER_ERR_PROXYCONNECT](#)
- [ERIC_TRANSFER_ERR_SEND](#)
- [ERIC_TRANSFER_ERR_SEND_INIT](#)
- [ERIC_TRANSFER_ERR_TIMEOUT](#)

Siehe auch:

- [EricMtCloseHandleToCertificate\(\)](#)
- [EricMtGetPinStatus\(\)](#)

ERICAPI_IMPORT int EricMtGetPinStatus ([EricInstanzHandle](#) instanz, [EricZertifikatHandle](#) hToken, [uint32_t](#) * pinStatus, [uint32_t](#) keyType)

Der PIN-Status wird für ein passwortgeschütztes Kryptomittel abgefragt und in `pinStatus` zurückgegeben.

Der PIN-Status wird für einen passwortgeschützten Bereich ermittelt, der durch das übergebene Zertifikat-Handle im Parameter `hToken` referenziert wird. Da bei Sicherheitssticks und Signaturkarten durch ein einziges Zertifikat-Handle zwei Schlüsselpaare referenziert werden können (eines für die Signatur und eines für die Verschlüsselung von Daten), muss grundsätzlich der Parameter `keyType` gesetzt werden.

Mit dem Rückgabewert der Funktion kann der Endanwender rechtzeitig informiert werden, falls bei einer weiteren falschen PIN-Eingabe das Kryptomittel gesperrt wird. Im Fehlerfall ist `pinStatus` nicht definiert.

Der Karten- bzw. Stickhersteller ist verantwortlich, dass seine Implementierung den korrekten PIN-Status zurückgibt, siehe auch Tabelle "PIN-Statusabfrage für POZ" im Unterkap. "Das Portalzertifikat (POZ)" im Dokument [ERIC-Entwicklerhandbuch.pdf](#).

Parameter:

in	<i>instanz</i>	Die ERIC-Instanz, auf der diese Funktion ausgeführt werden soll.
in	<i>hToken</i>	Zertifikat-Handle, für dessen passwortgeschützten Bereich der PIN-Status ermittelt werden soll. Wird von der Funktion EricMtGetHandleToCertificate() zurückgeliefert.
out	<i>pinStatus</i>	Mögliche Rückgabewerte: <ul style="list-style-type: none"> 0: StatusPinOk: Kein Fehlversuch oder keine Informationen verfügbar 1: StatusPinLocked: PIN gesperrt 2: StatusPreviousPinError: Die letzte PIN-Eingabe war fehlerhaft 3: StatusLockedIfPinError: Beim nächsten fehlerhaften Versuch wird die PIN gesperrt

in	<i>keyType</i>	Mögliche Eingabewerte: <ul style="list-style-type: none"> ◦ 0: eSignatureKey: Schlüssel für die Signatur von Daten ◦ 1: eEncryptionKey: Schlüssel für die Verschlüsselung von Daten
----	----------------	---

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- weitere, siehe [eric_fehlercodes.h](#)

Siehe auch:

- [EricMtGetHandleToCertificate\(\)](#)
- [ERiC-Entwicklerhandbuch.pdf](#), Kap. "Zertifikate und Authentifizierungsverfahren"

ERICAPI_IMPORT int EricMtGetPublicKey ([EricInstanzHandle](#) instanz, const [eric_verschluesselungs_parameter_t](#) * cryptoParameter, [EricRueckgabepufferHandle](#) rueckgabePuffer)

Es wird der öffentliche Schlüssel als base64-kodierte Zeichenkette für das übergebene Zertifikat-Handle in `cryptoParameter` zurückgeliefert. Von den beiden öffentlichen Schlüsseln, die in ELSTER-Zertifikaten enthalten sind, wird der Verschlüsselungsschlüssel (Encryption Key) zurückgegeben.

Parameter:

in	<i>instanz</i>	Die ERIC-Instanz, auf der diese Funktion ausgeführt werden soll.
in	<i>cryptoParameter</i>	Die Struktur enthält das Zertifikat-Handle und die PIN. Falls der Zugriff auf den öffentlichen Schlüssel keine PIN erfordert, ist PIN=NULL anzugeben.
out	<i>rueckgabePuffer</i>	Handle auf den Rückgabepuffer. Bei Erfolg enthält der Rückgabepuffer den öffentlichen Schlüssel als base64-kodierte Zeichenkette. Zur Erzeugung, Verwendung und Freigabe von Rückgabepuffern siehe Dokumentation zu EricRueckgabepufferHandle .

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_CRYPT_E_INVALID_HANDLE](#)
- [ERIC_CRYPT_E_P12_ENC_KEY](#)
- [ERIC_CRYPT_E_PIN_WRONG](#)
- [ERIC_CRYPT_E_PIN_LOCKED](#)
- weitere, siehe [eric_fehlercodes.h](#)

ERICAPI_IMPORT int EricMtHoleFehlerText (EricInstanzHandle instanz, int fehlerkode, EricRueckgabepufferHandle rueckgabePuffer)

Es wird die Klartextfehlermeldung zu dem `fehlerkode` ermittelt.

Die Funktion liefert die Klartextfehlermeldung zu einem ERiC Fehlercode - definiert in [eric_fehlercodes.h](#)

Parameter:

in	<i>instanz</i>	Die ERiC-Instanz, auf der diese Funktion ausgeführt werden soll.
in	<i>fehlerkode</i>	Eingabe-Fehlercode, definiert in eric_fehlercodes.h .
out	<i>rueckgabePuffer</i>	Handle auf einen Rückgabepuffer, in den die Klartextfehlermeldung geschrieben wird. Zur Erzeugung, Verwendung und Freigabe von Rückgabepuffern siehe Dokumentation zu EricRueckgabepufferHandle . Die Klartextfehlermeldung ist gemäß UTF-8 kodiert.

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_FEHLERMELDUNG_NICHT_VORHANDEN](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)

ERICAPI_IMPORT int EricMtHoleFinanzaemter ([EricInstanzHandle](#) instanz, const [byteChar](#) * finanzamtLandNummer, [EricRueckgabepufferHandle](#) rueckgabeXmlPuffer)

Es wird die Finanzamtliste für eine bestimmte `finanzamtLandNummer` zurückgegeben.

Parameter:

in	<i>instanz</i>	Die ERiC-Instanz, auf der diese Funktion ausgeführt werden soll.
in	<i>finanzamtLandNummer</i>	Die Finanzamtlandnummer besteht aus den ersten zwei Stellen der Bundesfinanzamtsnummer. Eine Liste aller Finanzamtlandnummern wird von EricMtHoleFinanzamtLandNummern() zurückgegeben.
out	<i>rueckgabeXmlPuffer</i>	Handle auf einen Rückgabepuffer, in den die Ergebnis XML-Daten geschrieben werden. Die XML-Daten folgen der XML Schema Definition Dokumentation\API-Rueckgabe-Schemata\EricHoleFinanzaemter.xsd . Zur Erzeugung, Verwendung und Freigabe von Rückgabepuffern siehe Dokumentation zu EricRueckgabepufferHandle .

Beispiel:

```
<?xml version="1.0" encoding="UTF-8"?>
<EricHoleFinanzaemter
xmlns="http://www.elster.de/EricXML/1.0/EricHoleFinanzaemter">
  <Finanzamt>
    <BuFaNummer>2801</BuFaNummer>
    <Name>Finanzamt Offenburg Außenstelle Achern</Name>
  </Finanzamt>
  <Finanzamt>
    <BuFaNummer>2804</BuFaNummer>
    <Name>Finanzamt Villingen-Schwenningen Außenstelle Donaueschingen</Name>
  </Finanzamt>
  (...)
</EricHoleFinanzaemter>
```

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_UTI_COUNTRY_NOT_SUPPORTED](#)
- [ERIC_GLOBAL_COMMONDATA_NICHT_VERFUEGBAR](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)

- [ERIC GLOBAL UNKNOWN](#)

ERICAPI_IMPORT int EricMtHoleFinanzamtLandNummern ([EricInstanzHandle](#) instanz, [EricRueckgabepufferHandle](#) rueckgabeXmlPuffer)

Die Liste aller Finanzamtlandnummern wird zurückgegeben.

Parameter:

in	<i>instanz</i>	Die ERiC-Instanz, auf der diese Funktion ausgeführt werden soll.
out	<i>rueckgabeXmlPuffer</i>	Handle auf einen Rückgabepuffer, in den die Ergebnis XML-Daten geschrieben werden. Die XML-Daten folgen der XML Schema Definition Dokumentation\API-Rueckgabe-Schemata\EricHoleFinanzamtLandNummern.xsd . Zur Erzeugung, Verwendung und Freigabe von Rückgabepuffern siehe Dokumentation zu EricRueckgabepufferHandle .

Beispiel:

```
<?xml version="1.0" encoding="UTF-8"?>
<EricHoleFinanzamtLandNummern
xmlns="http://www.elster.de/EricXML/1.0/EricHoleFinanzamtLandNummern">
  <FinanzamtLand>
    <FinanzamtLandNummer>28</FinanzamtLandNummer>
    <Name>Baden-Württemberg</Name>
  </FinanzamtLand>
  <FinanzamtLand>
    <FinanzamtLandNummer>91</FinanzamtLandNummer>
    <Name>Bayern (Zuständigkeit LfSt - München)</Name>
  </FinanzamtLand>
  (...)
</EricHoleFinanzamtLandNummern>
```

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_COMMONDATA_NICHT_VERFUEGBAR](#)
- [ERIC_GLOBAL_NICHT_GENUEGENDE_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)

[ERICAPI_IMPORT](#) int EricMtHoleFinanzamtsdaten ([EricInstanzHandle](#) instanz, const [byteChar](#) bufaNr[5], [EricRueckgabepufferHandle](#) rueckgabeXmlPuffer)

Die `finanzamtsdaten` werden für eine Bundesfinanzamtsnummer zurückgegeben.

Die Bundesfinanzamtsnummer kann über die Kombination der Funktionen [EricMtHoleFinanzamtLandNummern\(\)](#) und [EricMtHoleFinanzaemter\(\)](#) ermittelt werden.

Parameter:

in	<i>instanz</i>	Die ERiC-Instanz, auf der diese Funktion ausgeführt werden soll.
in	<i>bufaNr</i>	Übergabe der 4-stelligen Bundesfinanzamtsnummer.
out	<i>rueckgabeXmlPuffer</i>	Handle auf einen Rückgabepuffer, in den die Ergebnis XML-Daten geschrieben werden. Die XML-Daten folgen der XML Schema Definition Dokumentation\API-Rueckgabe-Schemata\EricHoleFinanzamtsdaten.xsd . Zur Erzeugung, Verwendung und Freigabe von Rückgabepuffern siehe Dokumentation zu EricRueckgabepufferHandle .

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#): Parameter `bufaNr` ist NULL.
- [ERIC_GLOBAL_PRUEF_FEHLER](#): Die übergebene Bundesfinanzamtsnummer ist keine Ganzzahl.
- [ERIC_GLOBAL_KEINE_DATEN_VORHANDEN](#): Immer bei Testfinanzämtern.
- [ERIC_GLOBAL_COMMONDATA_NICHT_VERFUEGBAR](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)

Siehe auch:

- [EricMtHoleFinanzamtLandNummern\(\)](#)
- [EricMtHoleFinanzaemter\(\)](#)

ERICAPI_IMPORT int EricMtHoleTestfinanzaemter ([EricInstanzHandle](#) instanz, [EricRueckgabepufferHandle](#) rueckgabeXmlPuffer)

Die Testfinanzamtliste wird in `rueckgabeXmlPuffer` zurückgegeben.

Parameter:

in	<i>instanz</i>	Die ERiC-Instanz, auf der diese Funktion ausgeführt werden soll.
out	<i>rueckgabeXmlPuffer</i>	Handle auf einen Rückgabepuffer, in den die Ergebnis XML-Daten geschrieben werden. Die XML-Daten folgen der XML Schema Definition Dokumentation\API-Rueckgabe-Schemata\EricHoleTestFinanzaemter.xsd . Zur Erzeugung, Verwendung und Freigabe von Rückgabepuffern siehe Dokumentation zu EricRueckgabepufferHandle .

Beispiel:

```
<?xml version="1.0" encoding="UTF-8"?>
<EricHoleTestFinanzaemter
xmlns="http://www.elster.de/EricXML/1.0/EricHoleTestFinanzaemter">
  <Finanzamt>
    <BuFaNummer>1096</BuFaNummer>
    <Name>Testfinanzamt Saarland</Name>
  </Finanzamt>
  <Finanzamt>
    <BuFaNummer>1097</BuFaNummer>
    <Name>Finanzschule (Edenkoben)</Name>
  </Finanzamt>
  (...)
</EricHoleTestFinanzaemter>
```

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_COMMONDATA_NICHT_VERFUEGBAR](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)

ERICAPI_IMPORT int EricMtHoleZertifikatEigenschaften ([EricInstanzHandle](#) instanz, [EricZertifikatHandle](#) hToken, const [byteChar](#) * pin, [EricRueckgabepufferHandle](#) rueckgabeXmlPuffer)

Die Eigenschaften des übergebenen Zertifikats werden im `rueckgabeXmlPuffer` zurückgegeben.

Parameter:

in	<i>instanz</i>	Die ERiC-Instanz, auf der diese Funktion ausgeführt werden soll.
in	<i>hToken</i>	Handle des Zertifikats, dessen Eigenschaften geholt werden sollen. Wird von der Funktion EricMtGetHandleToCertificate() zurückgeliefert.
in	<i>pin</i>	PIN zum Öffnen des Zertifikats. Wird bei Software-Portalzertifikaten benötigt.
out	<i>rueckgabeXmlPuffer</i>	Handle auf einen Rückgabepuffer, in den die Zertifikateigenschaften im XML-Format geschrieben werden. Das Format ist im XML Schema Dokumentation\API-Rueckgabe-Schemata\EricHoleZertifikatEigenschaften.xsd definiert. Zur Erzeugung, Verwendung und Freigabe von Rückgabepuffern siehe Dokumentation zu EricRueckgabepufferHandle .

Zu beachten:

Bei einem ELSTER-Softwarezertifikat (.pfx) steht im Common Name (CN) die ID des ELSTER-Kontos, für das das Zertifikat ausgestellt wurde. Die Konto-ID kann beispielsweise dafür genutzt werden, bei einer Zertifikatsverlängerung das verlängerte Zertifikat dem alten Zertifikat zuzuordnen.

Beispiel:

```
<EricHoleZertifikatEigenschaften
xmlns="http://www.elster.de/EricXML/2.0/EricHoleZertifikatEigenschaften">
  <Signaturzertifikateigenschaften>
    <AusgestelltAm>220817152116Z</AusgestelltAm>
    <GueltigBis>230817152116Z</GueltigBis>

  <Signaturalgorithmus>sha1WithRSAEncryption(1.2.840.113549.1.1.5)</Signaturalgorithmus>

  <PublicKeyMD5>6b8b191936677957fe74103198e77f4e</PublicKeyMD5>
  <PublicKeySHA1>884b0dfe2e10221a2aedd28c986cf34db0f1d932</PublicKeySHA1>
  <PublicKeyBitLength>2048</PublicKeyBitLength>

  <Issuer>
    <Info><Name>CN</Name><Wert>ElsterSoftCA</Wert></Info>
    <Info><Name>OU</Name><Wert>CA</Wert></Info>

    (...)
  </Issuer>
</EricHoleZertifikatEigenschaften>
```

```
</Issuer>
<Subjekt>
  <Info><Name>CN</Name><Wert>1000872896</Wert></Info>
</Subjekt>
<Identifikationsmerkmaltyp>Steuernummer</Identifikationsmerkmaltyp>
<Registrierertyp>Person</Registrierertyp>
<Verifikationsart>Postweg</Verifikationsart>
<TokenTyp>Software</TokenTyp>
<Testzertifikat>true</Testzertifikat>
</Signaturzertifikateigenschaften>
<Verschlüsselungszertifikateigenschaften>
  (...)
</Verschlüsselungszertifikateigenschaften>
</EricHoleZertifikatEigenschaften>
```

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)
- ERIC_CRYPT_E_*: Ein Zertifikatsfehler aus dem Statuscodebereich von [ERIC_CRYPT_E_INVALID_HANDLE](#) = 610201101 bis 610201212

Siehe auch:

- [ERiC-Entwicklerhandbuch.pdf](#), Kap. "Verwendung von EricHoleZertifikatEigenschaften()"
- [Dokumentation\API-Rueckgabe-Schemata\EricHoleZertifikatEigenschaften.xsd](#)

[ERICAPI_IMPORT](#) int EricMtHoleZertifikatFingerabdruck ([EricInstanzHandle](#) instanz, const [eric_verschluesselungs_parameter_t](#) * cryptoParameter, [EricRueckgabepufferHandle](#) fingerabdruckPuffer, [EricRueckgabepufferHandle](#) signaturPuffer)

Der Fingerabdruck und dessen Signatur wird für das übergebene Zertifikat zurückgegeben.

Zur Erzeugung, Verwendung und Freigabe von Rückgabepuffern siehe Dokumentation zu [EricRueckgabepufferHandle](#).

Parameter:

in	<i>instanz</i>	Die ERiC-Instanz, auf der diese Funktion ausgeführt werden soll.
in	<i>cryptoParameter</i>	Zertifikatsdaten, siehe eric_verschluesselungs_parameter_t . Das in der übergebenen Struktur referenzierte Zertifikat muss ein clientseitig erzeugtes Zertifikat (CEZ) sein.
out	<i>fingerabdruckPuffer</i>	Handle auf einen Rückgabepuffer, in den der Fingerabdruck geschrieben wird, siehe EricRueckgabepufferHandle .
out	<i>signaturPuffer</i>	Handle auf einen Rückgabepuffer, in den die Signatur des Fingerabdrucks geschrieben wird, siehe EricRueckgabepufferHandle .

Zu beachten:

Die Erzeugung eines Fingerabdrucks mit dieser Funktion ist nur in Zusammenhang mit clientseitig erzeugten Zertifikaten definiert.

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_PUFFER_ZUGRIFFSKONFLIKT](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)
- [ERIC_CRYPT_E_P12_READ](#)
- [ERIC_CRYPT_E_P12_DECODE](#)
- [ERIC_CRYPT_E_PIN_WRONG](#)
- [ERIC_CRYPT_E_P12_SIG_KEY](#)
- [ERIC_CRYPT_E_P12_ENC_KEY](#)
- [ERIC_CRYPT_ZERTIFIKAT](#)
- [ERIC_CRYPT_EIDKARTE_NICHT_UNTERSTUETZT](#)

- [ERIC_CRYPT_SIGNATUR](#)
- [ERIC_CRYPT_CORRUPTED](#)

[ERICAPI_IMPORT EricInstanzHandle](#) EricMtInstanzErzeugen (const char * pluginPfad, const char * logPfad)

Erstellt und initialisiert eine neue ERiC-Instanz.

Der erzeugte [EricInstanzHandle](#) ist im Parameter `instanz` der Multithreading-API zu übergeben. Zum Beenden einer ERiC-Instanz ist [EricMtInstanzFreigeben\(\)](#) aufzurufen.

Parameter:

in	<i>pluginPfad</i>	Pfad, in dem die Plugins rekursiv gesucht werden. Ist der Zeiger gleich NULL, wird der Pfad zur Bibliothek ericapi verwendet.
in	<i>logPfad</i>	Optionaler Pfad zur Log-Datei eric.log. Ist der Wert gleich NULL, wird das betriebssystemspezifische Verzeichnis für temporäre Dateien verwendet.

Rückgabe:

- [EricInstanzHandle](#) != NULL: Zeiger auf die erzeugte ERiC-Instanz.
- [EricInstanzHandle](#) == NULL: Fehler, Fehlerursache siehe Protokolldatei eric.log

Zu beachten:

Kann kein eric.log angelegt werden, wird eine entsprechende Fehlermeldung auf die Konsole (stderr) geschrieben und an den Windows-Ereignisdienst bzw. den syslogd-Dienst (Linux, AIX, macOS) geschickt.

Für Linux, AIX und macOS ist zu beachten, dass der syslogd-Dienst gegebenenfalls erst noch zu aktivieren und für die Protokollierung von Meldungen der Facility "User" zu konfigurieren ist. Suchkriterien für ERiC-Meldungen in der Windows-Ereignisansicht sind "ERiC (Elster Rich Client)" als Quelle und "Anwendung" als Protokoll.

Suchkriterien für ERiC-Meldungen in den Systemlogdateien unter Linux, AIX und macOS sind die Facility "User" und der Ident "ERiC (Elster Rich Client)".

Siehe auch:

- [ERiC-Entwicklerhandbuch.pdf](#), Kap. "Hinweise zum optimierten Einsatz von ERiC-Instanzen und Plugins"
- [EricMtInstanzFreigeben\(\)](#)

ERICAPI_IMPORT int EricMtInstanzFreigeben (EricInstanzHandle instanz)

Die übergebene ERiC-Instanz wird beendet und deren Speicher freigegeben.

Die freigegebene ERiC-Instanz kann nicht mehr verwendet werden. Andere ERiC-Instanzen bleiben von der Freigabe unberührt und können weiter verwendet werden.

Parameter:

in	<i>instanz</i>	ERiC-Instanz, die freigegeben werden soll.
----	----------------	--

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_UNGUELTIGE_INSTANZ](#)
- [ERIC_GLOBAL_UNGUELTIGER_PARAMETER](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)

Siehe auch:

- [EricMtInstanzErzeugen\(\)](#)

[ERICAPI_IMPORT](#) int EricMtMakeElsterEWAZ ([EricInstanzHandle](#) instanz, const [byteChar](#) * ewAzBescheid, const [byteChar](#) * landeskuerzel, [EricRueckgabepufferHandle](#) ewAzElsterPuffer)

Konvertiert ein Einheitswert-Aktenzeichen in das ELSTER-Format.

Konvertiert ein gültiges Einheitswert-Aktenzeichen in einem landesspezifischen Bescheidformat (z.B. 208/035-3-03889.3) unter Angabe des Landeskürzels in ein Einheitswert-Aktenzeichen im ELSTER-Format (z.B. 520840353038893).

Parameter:

in	<i>instanz</i>	Die ERiC-Instanz, auf der diese Funktion ausgeführt werden soll.
in	<i>ewAzBescheid</i>	Zeiger auf das Einheitswert-Aktenzeichen in einem landesspezifischen Bescheidformat.
in	<i>landeskuerzel</i>	Zeiger auf das Landeskürzel (zum Beispiel BY für Bayern)
out	<i>ewAzElsterPuffer</i>	Handle auf einen Rückgabepuffer, in den das erzeugte Einheitswert-Aktenzeichen im ELSTER-Format geschrieben wird.

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_EWAZ_UNGUELTIG](#)
- [ERIC_GLOBAL_EWAZ_LANDESKUERZEL_UNBEKANNT](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_UNGUELTIGER_PARAMETER](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)

Siehe auch:

- Landeskürzel siehe ISO-3166-2

**[ERICAPI_IMPORT](#) int EricMtMakeElsterStnr ([EricInstanzHandle](#) instanz, const [byteChar](#) *
steuernrBescheid, const [byteChar](#) landesnr[2+1], const [byteChar](#)
bundesfinanzamtsnr[4+1], [EricRueckgabepufferHandle](#) steuernrPuffer)**

Es wird eine Steuernummer im ELSTER-Steuernummerformat erzeugt.

Die Funktion erzeugt aus einer angegebenen Steuernummer im Format des Steuerbescheides eine 13-stellige Steuernummer im ELSTER-Steuernummerformat.

Die sich ergebende 13-stellige Steuernummer im ELSTER-Steuernummerformat wird von der Funktion [EricMtMakeElsterStnr\(\)](#) auch auf Gültigkeit geprüft.

Einer der beiden Parameter `landesnr` oder `bundesfinanzamtsnr` muss korrekt angegeben werden. Der jeweils andere Parameter darf NULL oder leer sein. Bei bayerischen und berliner Steuernummern im Format BBB/UUUUP ist die Angabe der Bundesfinanzamtsnummer zwingend erforderlich.

Parameter:

in	<i>instanz</i>	Die ERiC-Instanz, auf der diese Funktion ausgeführt werden soll.
in	<i>steuernrBescheid</i>	Format der Steuernummer wie auch auf amtlichen Schreiben angegeben.
in	<i>landesnr</i>	2-stellige Landesnummer (entspricht den ersten zwei Stellen der Bundesfinanzamtsnummer).
in	<i>bundesfinanzamtsnr</i>	4-stellige Bundesfinanzamtsnummer.
out	<i>steuernrPuffer</i>	Handle auf einen Rückgabepuffer, in den die Steuernummer im ELSTER-Steuernummerformat geschrieben wird. Zur Erzeugung, Verwendung und Freigabe von Rückgabepuffern siehe Dokumentation zu EricRueckgabepufferHandle .

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_STEUERNUMMER_UNGUELTIG](#)
- [ERIC_GLOBAL_LANDESNUMMER_UNBEKANNT](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_UNGUELTIGER_PARAMETER](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)

ERICAPI_IMPORT int EricMtPruefeBIC (EricInstanzHandle instanz, const byteChar * bic)

Die `bic` wird auf Gültigkeit überprüft.

Die Prüfung erfolgt in zwei Schritten:

1. Formale Prüfung auf gültige Zeichen und richtige Länge.
2. Prüfung, ob das Länderkennzeichen für BIC gültig ist.

Falls die BIC ungültig ist liefert die Funktion EricMtHoleFehlerText() den zugehörigen Fehlertext.

Parameter:

in	<i>instanz</i>	Die ERiC-Instanz, auf der diese Funktion ausgeführt werden soll.
in	<i>bic</i>	Zeiger auf eine NULL-terminierte Zeichenkette.

Rückgabe:

- ERIC_OK
- ERIC_GLOBAL_BIC_FORMALER_FEHLER: Ungültige Zeichen, falsche Länge.
- ERIC_GLOBAL_BIC_LAENDERCODE_FEHLER
- ERIC_GLOBAL_NULL_PARAMETER: Parameter `bic` ist NULL.
- ERIC_GLOBAL_COMMONDATA_NICHT_VERFUEGBAR
- ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER
- ERIC_GLOBAL_UNKNOWN

Siehe auch:

- ERiC-Entwicklerhandbuch.pdf, Kap. "BIC ISO-Ländercodes"
- ERiC-Entwicklerhandbuch.pdf, Kap. "BIC-Prüfung"

[ERICAPI_IMPORT](#) int EricMtPruefeBuFaNummer ([EricInstanzHandle](#) instanz, const [byteChar](#) * steuernummer)

Die Bundesfinanzamtsnummer wird überprüft.

Wird eine 13-stellige Steuernummer im ELSTER-Steuernummernformat angegeben, so wird nur die Bundesfinanzamtsnummer (= die ersten 4 Stellen der 13-stelligen Steuernummer) geprüft.

Eine Prüfung der Steuernummer selbst findet nicht statt (hierfür [EricMtPruefeSteuernummer\(\)](#) verwenden).

Parameter:

in	<i>instanz</i>	Die ERiC-Instanz, auf der diese Funktion ausgeführt werden soll.
in	<i>steuernummer</i>	13-stellige Steuernummer im ELSTER Steuernummernformat bzw. 4-stellige Bundesfinanzamtsnummer.

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_BUFANR_UNBEKANNT](#): Die Bundesfinanzamtsnummer ist unbekannt oder ungültig.
- [ERIC_GLOBAL_NULL_PARAMETER](#): Es wurde keine Bundesfinanzamtsnummer übergeben (Parameter ist NULL).
- [ERIC_GLOBAL_COMMONDATA_NICHT_VERFUEGBAR](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)

Siehe auch:

- [EricMtPruefeSteuernummer\(\)](#)
- Pruefung_der_Steuer_und_Steueridentifikatsnummer.pdf, siehe [Entwicklerbereich](#) bei [ELSTER](#).

[ERICAPI_IMPORT](#) int EricMtPruefeEWaz ([EricInstanzHandle](#) instanz, const [byteChar](#) * einheitswertAz)

Überprüft ein `Einheitswert-Aktenzeichen` im ELSTER-Format auf Gültigkeit.

Parameter:

in	<i>instanz</i>	Die ERiC-Instanz, auf der diese Funktion ausgeführt werden soll.
in	<i>einheitswertAz</i>	Zeiger auf ein Einheitswert-Aktenzeichen im ELSTER-Format

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_EWAZ_UNGUELTIG](#)
- [ERIC_GLOBAL_COMMONDATA_NICHT_VERFUEGBAR](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)

[ERICAPI_IMPORT](#) int EricMtPruefelIBAN ([EricInstanzHandle](#) instanz, const [byteChar](#) * iban)

Die `iban` wird auf Gültigkeit überprüft.

Die Prüfung erfolgt in vier Schritten:

1. Formale Prüfung auf gültige Zeichen und richtige Länge.
2. Prüfung, ob das Länderkennzeichen für IBAN gültig ist.
3. Prüfung, ob das länderspezifische Format gültig ist.
4. Prüfung, ob die Prüfziffer der IBAN gültig ist.

Falls die IBAN ungültig ist, liefert die Funktion [EricMtHoleFehlerText\(\)](#) den zugehörigen Fehlertext.

Parameter:

in	<i>instanz</i>	Die ERiC-Instanz, auf der diese Funktion ausgeführt werden soll.
in	<i>iban</i>	Zeiger auf eine NULL-terminierte Zeichenkette.

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_IBAN_FORMALER_FEHLER](#): Ungültige Zeichen, falsche Länge.
- [ERIC_GLOBAL_IBAN_LAENDERCODE_FEHLER](#)
- [ERIC_GLOBAL_IBAN_LANDESFORMAT_FEHLER](#)
- [ERIC_GLOBAL_IBAN_PRUEFZIFFER_FEHLER](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#): Parameter `iban` ist NULL.
- [ERIC_GLOBAL_COMMONDATA_NICHT_VERFUEGBAR](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)

Siehe auch:

- [ERIC-Entwicklerhandbuch.pdf](#), Kap. "IBAN - länderspezifische Formate"
- [ERIC-Entwicklerhandbuch.pdf](#), Kap. "IBAN-Prüfung"

[ERICAPI_IMPORT](#) int EricMtPruefeldentifikationsMerkmal ([EricInstanzHandle](#) instanz, const [byteChar](#) * steuerId)

Die `steuerId` wird auf Gültigkeit überprüft.

Parameter:

in	<i>instanz</i>	Die ERiC-Instanz, auf der diese Funktion ausgeführt werden soll.
in	<i>steuerId</i>	Steuer-Identifikationsnummer (IdNr)

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_IDNUMMER_UNGUELTIG](#)
- [ERIC_GLOBAL_COMMONDATA_NICHT_VERFUEGBAR](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)

Siehe auch:

- [EricMtPruefeSteuernummer\(\)](#)
- [ERIC-Entwicklerhandbuch.pdf](#), Kap. "Prüfung der Steueridentifikationsnummer (IdNr)"
- [ERIC-Entwicklerhandbuch.pdf](#), Kap. "Test-Steueridentifikationsnummer"

[ERICAPI_IMPORT](#) int EricMtPruefeSteuernummer ([EricInstanzHandle](#) instanz, const [byteChar](#) * steuernummer)

Die `steuernummer` wird einschließlich Bundesfinanzamtsnummer auf formale Richtigkeit geprüft.

Zur Prüfung der Bundesfinanzamtsnummer wird [EricMtPruefeBuFaNummer\(\)](#) verwendet.

Zur Erzeugung, Verwendung und Freigabe von Rückgabepuffern siehe Dokumentation zu [EricRueckgabepufferHandle](#).

Parameter:

in	<i>instanz</i>	Die ERiC-Instanz, auf der diese Funktion ausgeführt werden soll.
in	<i>steuernummer</i>	NULL-terminierte 13-stellige Steuernummer im ELSTER-Steuernummernformat.

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_STEUERNUMMER_UNGUELTIG](#)
- [ERIC_GLOBAL_COMMONDATA_NICHT_VERFUEGBAR](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)

Siehe auch:

- [EricMtPruefeBuFaNummer\(\)](#)
- [Pruefung_der_Steuer_und_Steueridentifikatsnummer.pdf](#), siehe [Entwicklerbereich](#) bei [ELSTER](#).

ERICAPI_IMPORT int EricMtPruefeWldNr (EricInstanzHandle instanz, const byteChar * wldNr)

Die Wirtschafts-Identifikationsnummer (W-IdNr.) wird auf formale Gültigkeit überprüft.

Parameter:

in	<i>instanz</i>	Die ERiC-Instanz, auf der diese Funktion ausgeführt werden soll.
in	<i>wldNr</i>	NULL-terminierte Wirtschafts-Identifikationsnummer mit oder ohne Unterscheidungsmerkmal.

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_IDNUMMER_UNGUELTIG](#)
- [ERIC_GLOBAL_COMMONDATA_NICHT_VERFUEGBAR](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)

Siehe auch:

- [ERIC-Entwicklerhandbuch.pdf](#), Kap. "Prüfung der Wirtschafts-Identifikationsnummer (W-IdNr.)"

ERICAPI_IMPORT int EricMtPruefeZertifikatPin ([EricInstanzHandle](#) instanz, const [byteChar](#) * pathToKeystore, const [byteChar](#) * pin, [uint32_t](#) keyType)

Prüft, ob die `pin` zum Zertifikat `pathToKeystore` passt. Nicht anwendbar auf Ad Hoc-Zertifikate (AHZ), die für einen neuen Personalausweis (nPA) ausgestellt sind.

Parameter:

in	<i>instanz</i>	Die ERiC-Instanz, auf der diese Funktion ausgeführt werden soll.
in	<i>pathToKeystore</i>	<p>Folgende Zertifikatstypen werden unterstützt:</p> <ol style="list-style-type: none"> 1. Clientseitig erzeugtes Zertifikat: Pfad zum Verzeichnis, in dem sich die Zertifikats-Datei (.cer) und die Datei mit dem privaten Schlüssel (.p12) befinden. Diese Kryptomittel wurden mit EricMtCreateKey() erzeugt. Der Pfad zum Verzeichnis ist bei clientseitig erzeugten Zertifikaten relativ zum aktuellen Arbeitsverzeichnis oder absolut anzugeben. 2. Software-Portalzertifikat: Pfad zur Software-Zertifikatsdatei (i.d.R. mit der Endung .pfx). Der Pfad zur Datei ist bei Software-Zertifikaten relativ zum aktuellen Arbeitsverzeichnis oder absolut anzugeben. 3. Sicherheitsstick: Pfad zur Treiberdatei, siehe (2). Bitte beachten, dass der Treiber betriebssystemabhängig sein kann. Weitere Informationen in der Anleitung zum Sicherheitsstick oder unter https://www.sicherheitsstick.de. 4. Signaturkarte: Pfad zur Treiberdatei, welcher einen Zugriff auf die Signaturkarte ermöglicht, siehe (2). Weitere Informationen in der Anleitung zur Signaturkarte.
in	<i>pin</i>	PIN für den Zugriff auf den privaten Schlüssel des Zertifikats.

in	<i>keyType</i>	Mögliche Eingabewerte: <ul style="list-style-type: none"> ◦ 0: eSignatureKey: Schlüssel für die Signatur von Daten, siehe (1). ◦ 1: eEncryptionKey: Schlüssel für die Verschlüsselung von Daten, siehe (1).
----	----------------	---

(1) Bei einem Zertifikat wie dem mit [EricMtCreateKey\(\)](#) clientseitig erzeugten Zertifikat (CEZ), das nur einen einzigen, gemeinsamen Schlüssel für Signatur und Verschlüsselung besitzt, sind beide Eingabewerte erlaubt. Die Werte beziehen sich dann beide auf denselben Schlüssel.

(2) Bei Sicherheitssticks und Signaturkarten ist bei der Angabe des Treibers der Suchmechanismus nach dynamischen Modulen des jeweiligen Betriebssystems zu berücksichtigen. Weitere Informationen sind z.B. unter Windows der Dokumentation der `LoadLibrary()` oder unter Linux und macOS der Dokumentation der `dlopen()` zu entnehmen.

Pfade müssen auf Windows in der für Datei-Funktionen benutzten ANSI-Codepage, auf Linux, AIX und Linux Power in der für das Dateisystem benutzten Locale und auf macOS in der "decomposed form" von UTF-8 übergeben werden. Bitte weitere Betriebssystemspezifika bzgl. nicht erlaubter Zeichen in Pfaden und Pfadtrennzeichen beachten.

Für Details zu Pfaden im ERiC siehe [ERiC-Entwicklerhandbuch.pdf](#), Kapitel "Übergabe von Pfaden an ERiC API-Funktionen".

Es wird empfohlen, geöffnete Zertifikatshandle zu schließen, bevor mit der API-Funktion [EricMtPruefeZertifikatPin\(\)](#) das gewünschte Zertifikat geprüft wird.

Zu beachten:

Eine falsche PIN-Eingabe erhöht bei Sicherheitsstick und Signaturkarte den Zähler für Fehlversuche. Welche Zertifikatstypen aufgrund von 3 Fehlversuchen gesperrt werden, ist im [ERiC-Entwicklerhandbuch.pdf](#), Kap. "Das Portalzertifikat (POZ)" beschrieben.

Rückgabe:

- [ERIC_OK](#)
- [ERIC_CRYPT_E_PIN_WRONG](#)
- [ERIC_CRYPT_NICHT_UNTERSTUETZTES_PSE_FORMAT](#)
- [ERIC_CRYPT_EIDKARTE_NICHT_UNTERSTUETZT](#)
- [ERIC_CRYPT_E_PSE_PATH](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)

[ERICAPI_IMPORT](#) int [EricMtRegistriereFortschrittCallback](#) ([EricInstanzHandle](#) instanz, [EricFortschrittCallback](#) funktion, void * benutzerdaten)

Die `funktion` wird als Callback-Funktion für [EricMtBearbeiteVorgang\(\)](#) registriert.

Die registrierte Callback-Funktion wird von der Funktion [EricMtBearbeiteVorgang\(\)](#) aufgerufen, um bei der Verarbeitung den Fortschritt der einzelnen Arbeitsbereiche anzuzeigen.

Parameter:

in	<i>instanz</i>	Die ERIC-Instanz, auf der diese Funktion ausgeführt werden soll.
in	<i>funktion</i>	Zeiger auf die zu registrierende Funktion oder <code>NULL</code>
in	<i>benutzerdaten</i>	Zeiger, der der registrierten Funktion immer mitgegeben wird. Die Anwendung kann diesen Parameter dazu verwenden, einen Zeiger auf eigene Daten oder Funktionen an die zu registrierende Funktion übergeben zu lassen.

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_UNGUELTIGER_PARAMETER](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)

Bemerkungen:

- Wenn eine zuvor registrierte Funktion nicht mehr aufgerufen werden soll, ist [EricMtRegistriereFortschrittCallback\(\)](#) mit dem Wert `NULL` im Parameter `funktion` aufzurufen.
- Es ist nicht erlaubt eine ERIC API-Funktion aus einer Callback-Funktion aufzurufen.
- Die Verarbeitung im Callback findet synchron statt. Deshalb sollte der Callback sehr schnell ausgeführt werden.

Siehe auch:

- [EricFortschrittCallback](#)
- [EricMtBearbeiteVorgang\(\)](#)
- [ERIC-Entwicklerhandbuch.pdf](#), Kap. "Funktionen für Fortschrittcallbacks"

[ERICAPI_IMPORT](#) int [EricMtRegistriereGlobalenFortschrittCallback](#) ([EricInstanzHandle](#) instanz, [EricFortschrittCallback](#) funktion, void * benutzerdaten)

Die registrierte `funktion` wird als Callback-Funktion von [EricMtBearbeiteVorgang\(\)](#) aufgerufen und zeigt den Gesamtfortschritt der Verarbeitung an.

Parameter:

in	<i>instanz</i>	Die ERiC-Instanz, auf der diese Funktion ausgeführt werden soll.
in	<i>funktion</i>	Zeiger auf die zu registrierende Funktion oder <code>NULL</code>
in	<i>benutzerdaten</i>	Zeiger, der der registrierten Funktion immer mitgegeben wird. Die Anwendung kann diesen Parameter dazu verwenden, einen Zeiger auf eigene Daten oder Funktionen an die zu registrierende Funktion übergeben zu lassen.

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_UNGUELTIGER_PARAMETER](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)

Bemerkungen:

- Wenn eine zuvor registrierte Funktion nicht mehr aufgerufen werden soll, ist [EricMtRegistriereGlobalenFortschrittCallback\(\)](#) mit dem Wert `NULL` im Parameter `funktion` aufzurufen.
- Es ist nicht erlaubt eine ERiC API-Funktion aus einer Callback-Funktion aufzurufen.
- Die Verarbeitung im Callback findet synchron statt. Deshalb sollte der Callback sehr schnell ausgeführt werden.

Siehe auch:

- [EricMtBearbeiteVorgang\(\)](#)
- [ERIC-Entwicklerhandbuch.pdf](#), Kap. "Funktionen für Fortschrittcallbacks"

ERICAPI_IMPORT int EricMtRegistriereLogCallback ([EricInstanzHandle](#) instanz, [EricLogCallback](#) funktion, [uint32_t](#) schreibeEricLogDatei, void * benutzerdaten)

Die registrierte `funktion` wird als Callback-Funktion für jede Lognachricht aufgerufen. Die Ausgabe entspricht einer Zeile im `eric.log`.

Parameter:

in	<i>instanz</i>	Die ERiC-Instanz, auf der diese Funktion ausgeführt werden soll.
in	<i>funktion</i>	Zeiger auf die zu registrierende Funktion oder NULL.
in	<i>schreibeEricLogDatei</i>	Log-Nachrichten im <code>eric.log</code> : <ul style="list-style-type: none"> ◦ 1 Jede Log-Nachricht wird nach <code>eric.log</code> geschrieben. Der Parameter <code>funktion</code> kann auf eine Funktion zeigen oder NULL sein. ◦ 0 Falls <code>funktion != NULL</code> werden keine Log-Nachrichten nach <code>eric.log</code> geschrieben, andernfalls werden die Log-Nachrichten nach <code>eric.log</code> geschrieben.
in	<i>benutzerdaten</i>	Zeiger, welcher der registrierten Funktion immer mitgegeben wird. Die Anwendung kann diesen Parameter dazu verwenden, einen Zeiger auf eigene Daten oder Funktionen an die zu registrierende Funktion übergeben zu lassen.

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_UNGUELTIGER_PARAMETER](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)

Bemerkungen:

- Wenn eine zuvor registrierte Funktion nicht mehr aufgerufen werden soll, ist [EricMtRegistriereLogCallback\(\)](#) mit dem Wert `NULL` im Parameter `funktion` aufzurufen (=Deregistrierung).
- Vor dem Beenden der Steueranwendung ist eine registrierte Funktion zu deregistrieren, da es sonst zu einem Absturz kommen kann.
- Es ist nicht erlaubt eine ERiC API-Funktion aus einer Callback-Funktion aufzurufen.
- Die Verarbeitung im Callback findet synchron statt. Deshalb sollte der

Callback sehr schnell ausgeführt werden.

[ERICAPI_IMPORT EricRueckgabepufferHandle](#) [EricMtRueckgabepufferErzeugen](#) ([EricInstanzHandle](#) instanz)

Diese API-Funktion erzeugt einen Rückgabepuffer und gibt ein Handle darauf zurück.

Die von dieser Funktion erzeugten Rückgabepuffer werden verwendet, um die Ausgaben von ERiC-Funktionen (z.B. [EricMtBearbeiteVorgang\(\)](#)) aufzunehmen. Dazu wird das Rückgabepuffer-Handle für den Schreibvorgang an die ausgebende Funktion übergeben.

Zum Auslesen des von den API-Funktionen beschriebenen Puffers wird das Rückgabepuffer-Handle an [EricMtRueckgabepufferInhalt\(\)](#) übergeben. Ein einmal erzeugtes Rückgabepuffer-Handle kann für weitere nachfolgende Aufrufe von ERiC API-Funktionen wiederverwendet werden. Bei einer Wiederverwendung eines Handles werden frühere Inhalte überschrieben. Nach Verwendung muss jeder Rückgabepuffer mit [EricMtRueckgabepufferFreigeben\(\)](#) freigegeben werden.

Rückgabepuffer sind der Singlethreading-API bzw. einer ERiC-Instanz der Multithreading-API fest zugeordnet. Die Funktionen der ERiC API, die einen Rückgabepuffer entgegennehmen, geben den Fehlercode [ERIC_GLOBAL_PUFFER_UNGLEICHER_INSTANZ](#) zurück, wenn der übergebene Rückgabepuffer:

- mit der Singlethreading-API erzeugt worden ist und dann mit der Multithreading-API verwendet wird.
- mit der Multithreading-API erzeugt worden ist und dann mit der Singlethreading-API verwendet wird.
- mit einer ERiC-Instanz erzeugt worden ist und dann mit einer anderen Instanz verwendet wird.

Parameter:

in	<i>instanz</i>	Die ERiC-Instanz, auf der diese Funktion ausgeführt werden soll.
----	----------------	--

Rückgabe:

- [EricRueckgabepufferHandle](#) im Erfolgsfall.
- NULL im Fehlerfall.

Siehe auch:

- [EricMtRueckgabepufferLaenge\(\)](#)
- [EricMtRueckgabepufferInhalt\(\)](#)
- [EricMtRueckgabepufferFreigeben\(\)](#)

ERICAPI_IMPORT int EricMtRueckgabepufferFreigeben (**EricInstanzHandle** instanz, **EricRueckgabepufferHandle** handle)

Der durch das `handle` bezeichnete Rückgabepuffer wird freigegeben.

Das Handle darf danach nicht weiter verwendet werden. Es wird daher empfohlen, Handle-Variablen nach der Freigabe explizit auf NULL zu setzen.

Parameter:

in	<i>instanz</i>	Die ERiC-Instanz, auf der diese Funktion ausgeführt werden soll.
in	<i>handle</i>	Handle auf einen mit EricMtRueckgabepufferErzeugen() angelegten Rückgabepuffer. Dieser Rückgabepuffer darf nicht bereits freigegeben worden sein.

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_UNGUELTIGER_PARAMETER](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)

Siehe auch:

- [EricMtRueckgabepufferErzeugen\(\)](#)
- [EricMtRueckgabepufferLaenge\(\)](#)
- [EricMtRueckgabepufferInhalt\(\)](#)

ERICAPI_IMPORT `const char * EricMtRueckgabepufferInhalt (EricInstanzHandle instanz, EricRueckgabepufferHandle handle)`

Der durch das `handle` bezeichnete Inhalt des Rückgabepuffers wird zurückgegeben.

Der zurückgegebene Zeiger verweist auf ein Byte-Array, das alle in den Rückgabepuffer geschriebenen Bytes sowie eine abschließende NULL-Terminierung enthält. Dieses Array existiert so lange im Speicher, bis der Rückgabepuffer entweder (bei einer Wiederverwendung des Handles) erneut beschrieben oder der Puffer explizit freigegeben wird.

Parameter:

in	<i>instanz</i>	Die ERiC-Instanz, auf der diese Funktion ausgeführt werden soll.
in	<i>handle</i>	Handle auf einen mit EricMtRueckgabepufferErzeugen() angelegten Rückgabepuffer. Dieser Rückgabepuffer darf nicht bereits freigegeben worden sein.

Rückgabe:

- Zeiger auf den NULL-terminierten Rückgabepufferinhalt, wenn ein gültiges Handle übergeben wird.
- NULL: Bei Übergabe des ungültigen Handles NULL.

Siehe auch:

- [EricMtRueckgabepufferErzeugen\(\)](#)
- [EricMtRueckgabepufferLaenge\(\)](#)
- [EricMtRueckgabepufferFreigegeben\(\)](#)

ERICAPI_IMPORT uint32_t EricMtRueckgabepufferLaenge (**EricInstanzHandle** instanz, **EricRueckgabepufferHandle** handle)

Die Länge des Rückgabepufferinhalts wird zurückgegeben.

Die zurückgegebene Zahl entspricht der Anzahl von Bytes, die von einer zuvor aufgerufenen ERiC API-Funktion in den Rückgabepuffer geschrieben wurden. Die NULL-Terminierung, die bei Aufruf von [EricMtRueckgabepufferInhalt\(\)](#) an das zurückgegebene Byte-Array angefügt wird, wird bei dieser Längenangabe nicht berücksichtigt.

Parameter:

in	<i>instanz</i>	Die ERiC-Instanz, auf der diese Funktion ausgeführt werden soll.
in	<i>handle</i>	Handle auf einen mit EricMtRueckgabepufferErzeugen() angelegten Rückgabepuffer. Dieser Rückgabepuffer darf nicht bereits freigegeben worden sein.

Rückgabe:

- Anzahl der in den Rückgabepuffer geschriebenen Bytes, wenn ein gültiges Handle übergeben wird.
- 0: Bei Übergabe des ungültigen Handles NULL.

Siehe auch:

- [EricMtRueckgabepufferErzeugen\(\)](#)
- [EricMtRueckgabepufferInhalt\(\)](#)
- [EricMtRueckgabepufferFreigeben\(\)](#)

ERICAPI_IMPORT int EricMtSystemCheck (EricInstanzHandle instanz)

Es werden Plattform-, Betriebssystem- und ERiC-Informationen ausgegeben.

Diese Funktion liefert Informationen über die verwendeten ERiC-Bibliotheken, ERiC-Druckvorlagen, die eingesetzte Plattform, den Arbeitsspeicher und das verwendete Betriebssystem.

Parameter:

in	<i>instanz</i>	Die ERiC-Instanz, auf der diese Funktion ausgeführt werden soll.
----	----------------	--

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_UNGUELTIGER_PARAMETER](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- weitere, siehe [eric_fehlercodes.h](#)

Siehe auch:

- [EricMtVersion\(\)](#)

ERICAPI_IMPORT int EricMtVersion ([EricInstanzHandle](#) instanz, [EricRueckgabepufferHandle](#) rueckgabeXmlPuffer)

Es wird eine Liste sämtlicher Produkt- und Dateiversionen der verwendeten ERiC-Bibliotheken als XML-Daten zurückgegeben.

Diese Funktion kann bei auftretenden Fehlern die Fehlersuche beschleunigen und Supportfälle unterstützen.

Parameter:

in	<i>instanz</i>	Die ERiC-Instanz, auf der diese Funktion ausgeführt werden soll.
out	<i>rueckgabeXmlPuffer</i>	Handle auf einen Rückgabepuffer, in den zu allen ERiC-Bibliotheken die Produkt- und Dateiversionen als XML-Daten nach XML Schema Definition Dokumentation\API-Rueckgabe-Schemata\EricVersion.xsd geschrieben werden. Zur Erzeugung, Verwendung und Freigabe von Rückgabepuffern siehe Dokumentation zu EricRueckgabepufferHandle .

Beispiel:

```
<?xml version="1.0" encoding="UTF-8"?>
<EricVersion xmlns="http://www.elster.de/EricXML/1.0/EricVersion">
  <Bibliothek>
    <Name>ericapi.dll</Name>
    <Produktversion>99, 1, 2, 32767</Produktversion>
    <Dateiversion>2008, 3, 5, 0</Dateiversion>
  </Bibliothek>
  <Bibliothek>
    <Name>ericctrl.dll</Name>
    <Produktversion>99, 1, 2, 32767</Produktversion>
    <Dateiversion>2008, 3, 5, 0</Dateiversion>
  </Bibliothek>
  (...)
</EricVersion>
```

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_NICHT_GENUEGENDE_ARBEITSSPEICHER](#)
- weitere, siehe [eric fehlercodes.h](#)

Siehe auch:

- [EricMtSystemCheck\(\)](#)

[ERICAPI_IMPORT](#) int EricMtZertifikatOeffnenAusBytes ([EricInstanzHandle](#) instanz, [EricZertifikatHandle](#) * hToken, const [byteChar](#) * pkcs12Container, [uint32_t](#) containerGroesse, const [byteChar](#) * pkcs12Passwort)

Für das im Speicher übergebene Zertifikat wird das Handle `hToken` zurückgeliefert.

Die ERiC API benötigt Zertifikat-Handles typischerweise bei kryptografischen Operationen.

Zertifikat-Handles sollten möglichst frühzeitig, d.h. wenn sie nicht mehr benötigt werden, mit [EricMtCloseHandleToCertificate\(\)](#) freigegeben werden, spätestens jedoch zum Programmende bzw. vor dem Entladen der Bibliothek `ericapi`.

Parameter:

in	<i>instanz</i>	Die ERiC-Instanz, auf der diese Funktion ausgeführt werden soll
out	<i>hToken</i>	Handle zu einem der folgenden Zertifikate: <ul style="list-style-type: none"> ◦ Portalzertifikat ◦ clientseitig erzeugtes Zertifikat
in	<i>pkcs12Container</i>	Speicheradresse des Zertifikats im PKCS#12-Format. Es werden nur passwortgeschützte PKCS#12-Container akzeptiert.
in	<i>containerGroesse</i>	Größe des PKCS#12-Containers in Bytes
in	<i>pkcs12Passwort</i>	Das Passwort oder die PIN des PKCS#12-Containers. Dieser Parameter darf nicht NULL sein.

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)
- [ERIC_CRYPT_NICHT_UNTERSTUEZTES_PSE_FORMAT](#)
- [ERIC_CRYPT_E_MAX_SESSION](#)
- [ERIC_CRYPT_E_PSE_PATH](#)
- [ERIC_CRYPT_E_BUSY](#)
- [ERIC_CRYPT_E_P11_SLOT_EMPTY](#)
- [ERIC_CRYPT_E_NO_SIG_ENC_KEY](#)
- [ERIC_CRYPT_E_PIN_WRONG](#)
- [ERIC_CRYPT_E_LOAD_DLL](#)

- [ERIC_CRYPT_E_NO_SERVICE](#)
- [ERIC_CRYPT_E_ESICL_EXCEPTION](#)

Siehe auch:

- [EricMtCloseHandleToCertificate\(\)](#)
- [EricMtGetPinStatus\(\)](#)

ericversion.h-Dateireferenz

Makrodefinitionen

- #define [ERIC_MAJOR_VERSION](#) 43
 - #define [ERIC_MINOR_VERSION](#) 3
 - #define [ERIC_PATCH_VERSION](#) 2
-

Ausführliche Beschreibung

Bereitstellung der ERiC API-Version über C-Präprozessor Makros. Die ERiC API-Version entspricht nicht unbedingt der Version des Setup-Pakets.

Makro-Dokumentation

#define ERIC_MAJOR_VERSION 43

Steht für die Hauptversionsnummer.

#define ERIC_MINOR_VERSION 3

Steht für die untergeordnete Versionsnummer.

#define ERIC_PATCH_VERSION 2

Steht für die Patch-Versionnummer.

erictoolkit.h-Dateireferenz

Bereitstellung von Prüffunktionen ohne Abhängigkeit zu anderen ERiC Bibliotheken.

Makrodefinitionen

- `#define` [ETKAPI_DECL](#)

Funktionen

- [ETKAPI_DECL](#) int [EtkPruefeBuFaNummer](#) (const char *steuernummer)
Die Bundesfinanzamtsnummer wird überprüft.
- [ETKAPI_DECL](#) int [EtkPruefeBIC](#) (const char *bic)
Die `bic` wird auf Gültigkeit überprüft.
- [ETKAPI_DECL](#) int [EtkPruefeEWaz](#) (const char *einheitswertAz)
Überprüft ein `Einheitswert`-Aktenzeichen im ELSTER-Format auf Gültigkeit.
- [ETKAPI_DECL](#) int [EtkPruefeIBAN](#) (const char *iban)
Die `iban` wird auf Gültigkeit überprüft.
- [ETKAPI_DECL](#) int [EtkPruefeIdentifikationsMerkmal](#) (const char *steuerId)
Die `steuerId` wird auf Gültigkeit überprüft. Formal korrekte Test Identifikationsnummern (beginnen mit der Ziffer 0) sind zulässig.
- [ETKAPI_DECL](#) int [EtkPruefeSteuernummer](#) (const char *steuernummer)
Die `steuernummer` wird einschließlich Bundesfinanzamtsnummer auf formale Richtigkeit geprüft.
- [ETKAPI_DECL](#) int [EtkPruefeWldNr](#) (const char *wldNr)
Die `Wirtschafts-Identifikationsnummer (W-IdNr .)` wird auf formale Gültigkeit geprüft.

- [ETKAPI_DECL](#) const char * [EtkHoleProduktVersion](#) ()
Abfragen der Produktversion des ERiCToolKit.
 - [ETKAPI_DECL](#) const char * [EtkHoleDateiVersion](#) ()
Abfragen der Dateiversion des ERiCToolKit.
-

Ausführliche Beschreibung

Bereitstellung von Prüffunktionen ohne Abhängigkeit zu anderen ERiC Bibliotheken.

Makro-Dokumentation

#define ETKAPI_DECL

Dieses Makro ist nur für die Kompilierung unter Windows von Belang. Auf allen anderen Plattformen ist es leer definiert.

Unter Windows gibt es bei Einbindung der [erictoolkit.h](#) in fremden Code an, dass die deklarierten Funktionen zu importieren sind (dllimport).

Siehe auch:

- [ERICAPI_IMPORT](#)
-

Dokumentation der Funktionen

ETKAPI_DECL const char * EtkHoleDateiVersion ()

Abfragen der Dateiversion des ERiCToolKit.

Die Dateiversion wird in den bereitgestellten Speicher als NULL-terminierte C Zeichenkette zurückgegeben. Der Speicher muss/darf von der Anwendung nicht freigegeben werden.

Rückgabe:

- NULL-terminierte C Zeichenkette.

ETKAPI_DECL const char * EtkHoleProduktVersion ()

Abfragen der Produktversion des ERiCToolKit.

Die Produktversion wird in den bereitgestellten Speicher als NULL-terminierte C Zeichenkette zurückgegeben. Der Speicher muss/darf von der Anwendung nicht freigegeben werden.

Rückgabe:

- NULL-terminierte C Zeichenkette.

ETKAPI_DECL int EtkPruefeBIC (const char * bic)

Die `bic` wird auf Gültigkeit überprüft.

Die Prüfung erfolgt in zwei Schritten:

1. Formale Prüfung auf gültige Zeichen und richtige Länge
2. Prüfung, ob das Länderkennzeichen für BIC gültig ist.

Parameter:

in	<i>bic</i>	Zeiger auf eine NULL-terminierte Zeichenkette.
----	------------	--

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_BIC_FORMALER_FEHLER](#): Ungültige Zeichen, falsche Länge.
- [ERIC_GLOBAL_BIC_LAENDERCODE_FEHLER](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#): Parameter `bic` ist NULL.

Siehe auch:

- [ERiC-Entwicklerhandbuch.pdf](#), Kap. "BIC ISO-Ländercodes"
- [ERiC-Entwicklerhandbuch.pdf](#), Kap. "BIC-Prüfung"

ETKAPI_DECL int EtkPruefeBuFaNummer (const char * steuernummer)

Die Bundesfinanzamtsnummer wird überprüft.

Wird eine 13-stellige Steuernummer im ELSTER-Steuernummernformat angegeben, so wird nur die Bundesfinanzamtsnummer (= die ersten 4 Stellen der 13-stelligen Steuernummer) geprüft.

Eine Prüfung der Steuernummer selbst findet nicht statt (hierfür [EtkPruefeSteuernummer\(\)](#) verwenden).

Parameter:

in	<i>steuernummer</i>	13-stellige Steuernummer im ELSTER Steuernummernformat bzw. 4-stellige Bundesfinanzamtsnummer.
----	---------------------	--

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_BUFANR_UNBEKANNT](#): Die Bundesfinanzamtsnummer ist unbekannt oder ungültig.
- [ERIC_GLOBAL_NULL_PARAMETER](#): Es wurde keine Bundesfinanzamtsnummer übergeben (Parameter ist NULL).

Siehe auch:

- [EtkPruefeSteuernummer\(\)](#)
- Pruefung_der_Steuer_und_Steueridentifikatsnummer.pdf, siehe [Entwicklerbereich](#) bei [ELSTER](#).

ETKAPI_DECL int EtkPruefeEWaz (const char * einheitswertAz)

Überprüft ein `Einheitswert-Aktenzeichen` im ELSTER-Format auf Gültigkeit.

Parameter:

in	<i>einheitswertAz</i>	Zeiger auf ein Einheitswert-Aktenzeichen im ELSTER-Format
----	-----------------------	---

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_EWAZ_UNGUELTIG](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)

ETKAPI_DECL int EtkPruefelIBAN (const char * iban)

Die `iban` wird auf Gültigkeit überprüft.

Die Prüfung erfolgt in vier Schritten:

1. Formale Prüfung auf gültige Zeichen und richtige Länge.
2. Prüfung, ob das Länderkennzeichen für IBAN gültig ist.
3. Prüfung, ob das länderspezifische Format gültig ist.
4. Prüfung, ob die Prüfziffer der IBAN gültig ist.

Parameter:

in	<i>iban</i>	Zeiger auf eine NULL-terminierte Zeichenkette.
----	-------------	--

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_IBAN_FORMALER_FEHLER](#): Ungültige Zeichen, falsche Länge.
- [ERIC_GLOBAL_IBAN_LAENDERCODE_FEHLER](#)
- [ERIC_GLOBAL_IBAN_LANDESFORMAT_FEHLER](#)
- [ERIC_GLOBAL_IBAN_PRUEFZIFFER_FEHLER](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#): Parameter `iban` ist NULL.

Siehe auch:

- [ERiC-Entwicklerhandbuch.pdf](#), Kap. "IBAN - länderspezifische Formate"
- [ERiC-Entwicklerhandbuch.pdf](#), Kap. "IBAN-Prüfung"

ETKAPI_DECL int EtkPruefeldentifikationsMerkmal (const char * steuerId)

Die `steuerId` wird auf Gültigkeit überprüft. Formal korrekte Test Identifikationsnummern (beginnen mit der Ziffer 0) sind zulässig.

Parameter:

in	<i>steuerId</i>	Steuer-Identifikationsnummer (IdNr)
----	-----------------	-------------------------------------

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_IDNUMMER_UNGUELTIG](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)

Siehe auch:

- [ERiC-Entwicklerhandbuch.pdf](#), Kap. "Prüfung der Steueridentifikationsnummer (IdNr)"
- [ERiC-Entwicklerhandbuch.pdf](#), Kap. "Test-Steueridentifikationsnummer"
- [EtkPruefeSteuernummer\(\)](#)

ETKAPI_DECL int EtkPruefeSteuernummer (const char * steuernummer)

Die `steuernummer` wird einschließlich Bundesfinanzamtsnummer auf formale Richtigkeit geprüft.

Zur Prüfung der Bundesfinanzamtsnummer wird [EtkPruefeBuFaNummer\(\)](#) verwendet.

Parameter:

in	<i>steuernummer</i>	NULL-terminierte 13-stellige Steuernummer im ELSTER-Steuernummernformat.
----	---------------------	--

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_STEUERNUMMER_UNGUELTIG](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)

Siehe auch:

- [EtkPruefeBuFaNummer\(\)](#)
- [Pruefung_der_Steuer_und_Steueridentifikatsnummer.pdf](#), siehe [Entwicklerbereich](#) bei [ELSTER](#).

ETKAPI_DECL int EtkPruefeWldNr (const char * wldNr)

Die Wirtschafts-Identifikationsnummer (W-IdNr .) wird auf formale Gültigkeit geprüft.

Parameter:

in	<i>wldNr</i>	NULL-terminierte Wirtschafts-Identifikationsnummer mit oder ohne Unterscheidungsmerkmal.
----	--------------	--

Rückgabe:

- [ERIC_OK](#)
- [ERIC_GLOBAL_IDNUMMER_UNGUELTIG](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)

Siehe auch:

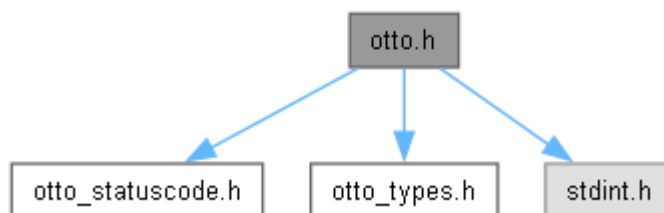
- [ERiC-Entwicklerhandbuch.pdf](#), Kap. "Prüfung der Wirtschafts-Identifikationsnummer (W-IdNr.)"

otto.h-Dateireferenz

Deklaration der Otto-Funktionen.

```
#include "otto_statuscode.h"
#include "otto_types.h"
#include <stdint.h>
```

Include-Abhängigkeitsdiagramm für otto.h:



Funktionen

- [OttoStatusCode](#) [OttoInstanzErzeugen](#) (const [byteChar](#) *logPfad, [OttoLogCallback](#) logCallback, void *logCallbackBenutzerdaten, [OttoInstanzHandle](#) *instanz)

Erstellt und initialisiert eine neue Otto-Instanz.
- [OttoStatusCode](#) [OttoInstanzFreigeben](#) ([OttoInstanzHandle](#) instanz)

Gibt eine Otto-Instanz frei.
- [OttoStatusCode](#) [OttoZertifikatOeffnen](#) ([OttoInstanzHandle](#) instanz, const [byteChar](#) *zertifikatsPfad, const [byteChar](#) *zertifikatsPasswort, [OttoZertifikatHandle](#) *zertifikat)

Erstellt ein Otto-Zertifikatsobjekt für ein Sicherheitstoken.
- [OttoStatusCode](#) [OttoZertifikatOeffnenAusBytes](#) ([OttoInstanzHandle](#) instanz, const [byteChar](#) *pkcs12Container, [uint32_t](#) containerGroesse, const [byteChar](#) *zertifikatsPasswort, [OttoZertifikatHandle](#) *zertifikat)

Erstellt ein Otto-Zertifikatsobjekt für ein PKCS#12-Sicherheitstoken, das im Hauptspeicher übergeben wird.
- [OttoStatusCode](#) [OttoZertifikatSchliessen](#) ([OttoZertifikatHandle](#) zertifikat)

Schließt das Otto-Zertifikatsobjekt zu einem Sicherheitstoken. Anschließend darf das Zertifikatsobjekt nicht mehr verwendet werden.

- [OttoStatusCode](#) [OttoRueckgabepufferErzeugen](#) ([OttoInstanzHandle](#) instanz, [OttoRueckgabepufferHandle](#) *rueckgabepuffer)
Erzeugt einen Rückgabepuffer und gibt ein Handle darauf zurück.
- uint64_t [OttoRueckgabepufferGroesse](#) ([OttoRueckgabepufferHandle](#) rueckgabepuffer)
Gibt die Anzahl der im Rückgabepuffer enthaltenen Bytes zurück. Das abschließende Null-Byte wird nicht mitgezählt.
- const [byteChar](#) * [OttoRueckgabepufferInhalt](#) ([OttoRueckgabepufferHandle](#) rueckgabepuffer)
Gibt den Inhalt eines Rückgabepuffers zurück.
- [OttoStatusCode](#) [OttoRueckgabepufferFreigeben](#) ([OttoRueckgabepufferHandle](#) rueckgabepuffer)
Gibt einen Rückgabepuffer frei.
- [OttoStatusCode](#) [OttoPruefsummeErzeugen](#) ([OttoInstanzHandle](#) instanz, [OttoPruefsummeHandle](#) *pruefsumme)
Erzeugt ein Objekt zur Berechnung einer Datenprüfsumme, die Otto zu Beginn einer Übermittlung an den OTTER-Server senden muss.
- [OttoStatusCode](#) [OttoPruefsummeAktualisieren](#) ([OttoPruefsummeHandle](#) pruefsumme, const [byteChar](#) *datenBlock, uint64_t datenBlockGroesse)
Aktualisiert die Prüfsumme über Daten. Eine Prüfsumme, die bereits signiert wurde, kann nicht mehr aktualisiert werden.
- [OttoStatusCode](#) [OttoPruefsummeSignieren](#) ([OttoPruefsummeHandle](#) pruefsumme, [OttoZertifikatHandle](#) zertifikat, [OttoRueckgabepufferHandle](#) rueckgabepuffer)
Erstellt eine Signatur über eine Prüfsumme.
- [OttoStatusCode](#) [OttoPruefsummeFreigeben](#) ([OttoPruefsummeHandle](#) pruefsumme)
Gibt ein Prüfsummenobjekt frei.

- [OttoStatusCode](#) [OttoVersandBeginnen](#) ([OttoInstanzHandle](#) instanz, const [byteChar](#) *signiertePruefsumme, const [byteChar](#) *herstellerId, [OttoVersandHandle](#) *versand)
Initialisiert einen Datenversand an den OTTER-Server.
- [OttoStatusCode](#) [OttoVersandFortsetzen](#) ([OttoVersandHandle](#) versand, const [byteChar](#) *datenBlock, uint64_t datenBlockGroesse)
Versendet einen Datenblock an den OTTER-Server.
- [OttoStatusCode](#) [OttoVersandAbschliessen](#) ([OttoVersandHandle](#) versand, [OttoRueckgabepufferHandle](#) objektId)
Schließt einen Versand ab und gibt die Objekt-ID zurück.
- [OttoStatusCode](#) [OttoVersandBeenden](#) ([OttoVersandHandle](#) versand)
Gibt ein Versandobjekt frei.
- [OttoStatusCode](#) [OttoEmpfangBeginnen](#) ([OttoInstanzHandle](#) instanz, const [byteChar](#) *objektId, [OttoZertifikatHandle](#) zertifikat, const [byteChar](#) *herstellerId, [OttoEmpfangHandle](#) *empfang)
Initialisiert eine Datenabholung vom OTTER-Server.
- [OttoStatusCode](#) [OttoEmpfangBeginnenAbholzertifikat](#) ([OttoInstanzHandle](#) instanz, const [byteChar](#) *objektId, [OttoZertifikatHandle](#) zertifikat, const [byteChar](#) *herstellerId, const [byteChar](#) *abholzertifikat, [OttoEmpfangHandle](#) *empfang)
Initialisiert eine Datenabholung vom OTTER-Server mit Angabe eines Abholzertifikats.
- [OttoStatusCode](#) [OttoEmpfangFortsetzen](#) ([OttoEmpfangHandle](#) empfang, [OttoRueckgabepufferHandle](#) datenBlock)
Empfängt einen Datenblock vom OTTER-Server.
- [OttoStatusCode](#) [OttoEmpfangBeenden](#) ([OttoEmpfangHandle](#) empfang)
Gibt das Empfangsobjekt wieder frei.
- [OttoStatusCode](#) [OttoDatenAbholen](#) ([OttoInstanzHandle](#) instanz, const [byteChar](#) *objektId, [uint32_t](#) objektGroesse, const [byteChar](#) *zertifikatsPfad, const

[byteChar](#) *zertifikatsPasswort, const [byteChar](#) *herstellerId, const [byteChar](#) *abholzertifikat, [OttoRueckgabepufferHandle](#) abholDaten)

Holt das Datenobjekt zu einer Objekt-ID von OTTER mit einem einzigen Funktionsaufruf vollständig ab.

- const char * [OttoHoleFehlertext](#) ([OttoStatusCode](#) statuscode)
Die Funktion liefert die Klartextfehlermeldung zu einem Otto-StatusCode - definiert in [otto_statuscode.h](#).
- [OttoStatusCode](#) [OttoProxyKonfigurationSetzen](#) ([OttoInstanzHandle](#) instanz, const [OttoProxyKonfiguration](#) *proxyKonfiguration)
Konfiguriert eine Otto-Instanz für einen Proxy.
- [OttoStatusCode](#) [OttoEinstellungSetzen](#) ([OttoInstanzHandle](#) instanz, const char *einstellungName, const char *einstellungWert)
Setzt den Wert einer Otto-Einstellung für die angegebene Instanz.
- [OttoStatusCode](#) [OttoEinstellungLesen](#) ([OttoInstanzHandle](#) instanz, const char *einstellungName, [OttoRueckgabepufferHandle](#) einstellungWert)
Liest den aktuellen Wert einer Otto-Einstellung in der angegebenen Instanz aus.
- [OttoStatusCode](#) [OttoVersion](#) ([OttoRueckgabepufferHandle](#) rueckgabepuffer)
Gibt die Version der Otto-Bibliothek zurück.

Ausführliche Beschreibung

Deklaration der Otto-Funktionen.

Dokumentation der Funktionen

[OttoStatusCode](#) OttoDatenAbholen ([OttoInstanzHandle](#) instanz, const [byteChar](#) * objektId, [uint32_t](#) objektGroesse, const [byteChar](#) * zertifikatsPfad, const [byteChar](#) * zertifikatsPasswort, const [byteChar](#) * herstellerId, const [byteChar](#) * abholzertifikat, [OttoRueckgabepufferHandle](#) abholDaten)

Holt das Datenobjekt zu einer Objekt-ID von OTTER mit einem einzigen Funktionsaufruf vollständig ab.

Diese Funktion ist eine bequemere Alternative zu der blockweisen Datenabholung über die OttoEmpfang-Funktionen. Intern bündelt sie die Aufrufe der OttoEmpfangs-Funktionen, wie sie sonst von der Anwendung selbst durchgeführt werden müßten.

Der Nachteil dieser Funktion gegenüber den OttoEmpfang-Funktionen besteht darin, dass die abgeholten Daten alle im Hauptspeicher von Otto gehalten werden. Sie eignet sich daher nicht für die Abholung sehr großer Datenobjekte oder wenn nur sehr wenig Hauptspeicher zur Verfügung steht.

Zu beachten:

Wurde eine Otto-Instanz vor dem Aufruf dieser Funktion mit [OttoProxyKonfigurationSetzen\(\)](#) für einen Proxy konfiguriert, wird die Abholung über den Proxy durchgeführt.

Parameter:

in	<i>instanz</i>	Handle der Otto-Instanz, auf der diese Funktion ausgeführt werden soll.
in	<i>objektId</i>	ID des Datenobjekts, das vom OTTER-Server abgeholt werden soll.
in	<i>objektGroesse</i>	Die erwartete Größe des Datenobjekts, das vom OTTER-Server abgeholt werden soll, in Bytes. Diesen Wert findet die Anwendung zusammen mit der Objekt-ID im Rückgabe-XML zu einer PostfachAnfrage. Wenn die Größe zu gering angegeben wird, geht dies zwar zu Lasten der Geschwindigkeit und des Hauptspeicherbedarfs, weil dann der Rückgabepuffer von Otto intern sukzessive vergrößert werden muß, aber es führt nicht zu einem Fehler.

in	<i>zertifikatsPfad</i>	<p>Pfad zum Sicherheitstoken, folgende Angaben sind möglich:</p> <ol style="list-style-type: none"> 1. Clientseitig erzeugtes Zertifikat: Pfad zum Verzeichnis, in dem sich die Zertifikats-Datei (.cer) und die Datei mit dem privaten Schlüssel (.p12) befinden. Diese Sicherheitstokens wurden mit EricMtCreateKey() bzw. EricCreateKey() erzeugt. Der Pfad zum Verzeichnis ist bei clientseitig erzeugten Zertifikaten relativ zum aktuellen Arbeitsverzeichnis oder absolut anzugeben. 2. Software-Portalzertifikat: Pfad zur Software-Zertifikatsdatei (i.d.R. mit der Endung .pfx). Der Pfad zur Datei ist bei Software-Zertifikaten relativ zum aktuellen Arbeitsverzeichnis oder absolut anzugeben. 3. Sicherheitsstick: Pfad zur Treiberdatei, siehe (*). Bitte beachten, dass der Treiber betriebssystemabhängig sein kann. Weitere Informationen in der Anleitung zum Sicherheitsstick oder unter https://www.sicherheitsstick.de. 4. Signaturkarte: (*) Pfad zur Treiberdatei, welcher einen Zugriff auf die Signaturkarte ermöglicht. Weitere Informationen in der Anleitung zur Signaturkarte. 5. Elektronischer Personalausweis (nPA) oder Aufenthaltstitel (eAT): Die URL des eID-Clients wie zum Beispiel der AusweisApp 2. In den meisten Fällen lautet diese URL: http://127.0.0.1:24727/eID-Client. Optional kann auf die folgende Weise noch ein Testmerker angehängt werden: http://127.0.0.1:24727/eID-Client?testmerker=520000000. Zu den verfügbaren Testmerkern siehe ERIC-Entwicklerhandbuch.pdf, Kap. "Testunterstützung bei der ERiC-Anbindung". <p>Wichtig: Das Ad-hoc-Zertifikat, das in diesem Fall für den elektronischen Personalausweis erzeugt wird, ist nur 24 Stunden gültig.</p>
----	------------------------	---

(*) Wird der Dateipfad eines Treibers angegeben, ist der Suchmechanismus zu beachten, mit dem das jeweilige Betriebssystem dynamische Bibliotheken lädt. Weitere Informationen sind der Systemdokumentation zu den Betriebssystemfunktionen `LoadLibrary()` (Windows) bzw. `dlopen()` (Linux, AIX und macOS) zu entnehmen.

Pfade müssen auf Windows in der für Datei-Funktionen benutzten ANSI-Codepage, auf Linux, AIX und Linux Power in der für das Dateisystem benutzten Locale und auf macOS in der "decomposed form" von UTF-8 übergeben werden. Bitte weitere Betriebssystemspezifika bzgl. nicht erlaubter Zeichen in Pfaden und Pfadtrennzeichen beachten.

Für Details zu Pfaden im ERiC siehe [ERiC-Entwicklerhandbuch.pdf](#), Kapitel "Übergabe von Pfaden an ERiC API-Funktionen".

Parameter:

in	<i>zertifikatsPasswort</i>	Das Passwort oder die PIN des Sicherheitstokens. Bei Tokens, bei denen das Passwort oder die PIN nicht von der Anwendung übergeben, sondern separat über einen Treiber (z. B. von einem Kartenlesegerät) abgefragt wird, ist hier NULL zu übergeben.
in	<i>herstellerId</i>	Hersteller-ID des Softwareproduktes
in	<i>abholzertifikat</i>	Base64-kodierter Teil eines X.509-v3-Zertifikats im PEM-Format. Die Angabe eines Abholzertifikats ist optional und nur erlaubt, wenn im Parameter <i>zertifikatsPfad</i> kein clientseitig erzeugtes Zertifikat (CEZ) angegeben wurde. Wird ein Abholzertifikat übergeben, so werden die Abholdaten vom Server auf den öffentlichen Schlüssel des Zertifikats umgeschlüsselt. Diese Daten werden vom Otto nicht entschlüsselt und OttoDatenAbholen() gibt lediglich die verschlüsselten Daten zurück. Wenn eine nicht bei ELSTER registrierte Signaturkarte zur Authentifizierung verwendet wird, muss dieser Parameter gesetzt werden, ansonsten kann hier NULL übergeben werden.
out	<i>abholDaten</i>	Rückgabepuffer mit den abgeholten Daten. Der Inhalt des Rückgabepuffers darf nicht als null-terminierte Zeichenkette interpretiert werden, da die abgeholten Daten weitere Null-Bytes enthalten können.

Rückgabe:

- [OTTO_OK](#)
- [OTTO_TRANSFER_UNAUTHORIZED](#)

- [OTTO TRANSFER NOT FOUND](#)
- weitere, siehe [otto_statuscode.h](#)

[OttoStatusCode](#) **OttoEinstellungLesen** ([OttoInstanzHandle](#) instanz, const char *
einstellungName, [OttoRueckgabepufferHandle](#) einstellungWert)

Liest den aktuellen Wert einer Otto-Einstellung in der angegebenen Instanz aus.

Parameter:

in	<i>instanz</i>	Die Otto-Instanz, deren Einstellung ausgelesen werden soll.
in	<i>einstellungName</i>	Name der Einstellung, deren Wert ausgelesen werden soll.
out	<i>einstellungWert</i>	Der ausgelesene Wert der Einstellung

Rückgabe:

- [OTTO_OK](#)
- [OTTO_EINSTELLUNG_UNBEKANNT](#)
- weitere, siehe [otto_statuscode.h](#)

Siehe auch:

- [OttoEinstellungSetzen\(\)](#)
- [ERiC-Entwicklerhandbuch.pdf](#), Kap. "Bedeutung der Otto-Einstellungen"

**OttoStatusCode OttoEinstellungSetzen (OttoInstanzHandle instanz, const char *
einstellungName, const char * einstellungWert)**

Setzt den Wert einer Otto-Einstellung für die angegebene Instanz.

Die Einstellungen gelten immer nur für die übergebene Otto-Instanz.

Die Änderungen von Werten ist nicht immer unmittelbar wirksam.

Parameter:

in	<i>instanz</i>	Die Otto-Instanz, für die eine Einstellung gesetzt werden soll.
in	<i>einstellungName</i>	Name der Einstellung, deren Wert gesetzt werden soll.
in	<i>einstellungWert</i>	Wert, auf den die Einstellung gesetzt werden soll.

Rückgabe:

- [OTTO_OK](#)
- [OTTO_EINSTELLUNG_UNBEKANNT](#)
- [OTTO_EINSTELLUNG_WERT_UNGUELTIG](#)
- weitere, siehe [otto_statuscode.h](#)

Siehe auch:

- [OttoEinstellungLesen\(\)](#)
- [ERiC-Entwicklerhandbuch.pdf](#), Kap. "Bedeutung der Otto-Einstellungen"

OttoStatusCode OttoEmpfangBeenden (OttoEmpfangHandle empfang)

Gibt das Empfangsobjekt wieder frei.

Das Empfangsobjekt darf nach diesem Aufruf nicht mehr verwendet werden. Wird diese Funktion aufgerufen, bevor [OttoEmpfangFortsetzen\(\)](#) einen leeren Rückgabepuffer zurückgegeben hat, können die bis dahin empfangenen Daten unvollständig sein.

Parameter:

in	<i>empfang</i>	Ein mit OttoEmpfangBeginnen() erzeugtes Handle
----	----------------	--

Rückgabe:

- [OTTO_OK](#)
- [OTTO_EMPFANG_VORZEITIG_BEENDET](#) falls noch nicht alle Daten mit [OttoEmpfangFortsetzen\(\)](#) empfangen wurden
- weitere, siehe [otto_statuscode.h](#)

[OttoStatusCode](#) **OttoEmpfangBeginnen** ([OttoInstanzHandle](#) instanz, const [byteChar](#) * objektId, [OttoZertifikatHandle](#) zertifikat, const [byteChar](#) * herstellerId, [OttoEmpfangHandle](#) * empfang)

Initialisiert eine Datenabholung vom OTTER-Server.

Das zurückgegebene Handle des Empfangsobjekts wird der Funktion [OttoEmpfangFortsetzen\(\)](#) übergeben, um Daten blockweise abzuholen. Sind alle Daten abgeholt, wird [OttoEmpfangBeenden\(\)](#) aufgerufen, womit das Empfangsobjekt wieder freigegeben wird.

Zu beachten:

Wurde eine Otto-Instanz vor dem Aufruf dieser Funktion mit [OttoProxyKonfigurationSetzen\(\)](#) für einen Proxy konfiguriert, wird der Empfang über den Proxy durchgeführt. Die Proxy-Konfiguration wird intern an dem Empfangsobjekt gespeichert und spätere Aufrufe von [OttoProxyKonfigurationSetzen\(\)](#) haben keinen Einfluss auf den bereits begonnenen Empfang.

Parameter:

in	<i>instanz</i>	Handle der Otto-Instanz, auf der diese Funktion ausgeführt werden soll.
in	<i>objektId</i>	ID des Objekts, das vom OTTER-Server abgeholt werden soll.
in	<i>zertifikat</i>	Handle auf ein Zertifikatsobjekt
in	<i>herstellerId</i>	Hersteller-ID des Softwareproduktes
out	<i>empfang</i>	Handle auf das Empfangsobjekt. Im Fehlerfall wird kein Empfangsobjekt erzeugt.

Rückgabe:

- siehe [otto_statuscode.h](#)

Siehe auch:

- [OttoEmpfangFortsetzen\(\)](#)
- [OttoEmpfangBeenden\(\)](#)
- [OttoProxyKonfigurationSetzen\(\)](#)

[OttoStatusCode](#) [OttoEmpfangBeginnenAbholzertifikat](#) ([OttoInstanzHandle](#) instanz, const [byteChar](#) * objektId, [OttoZertifikatHandle](#) zertifikat, const [byteChar](#) * herstellerId, const [byteChar](#) * abholzertifikat, [OttoEmpfangHandle](#) * empfang)

Initialisiert eine Datenabholung vom OTTER-Server mit Angabe eines Abholzertifikats.

Die Angabe eines Abholzertifikats ist erforderlich, wenn eine nicht bei ELSTER registrierte Signaturkarte zur Authentifizierung verwendet wird.

Die Funktion darf nicht verwendet werden, wenn zur Authentifizierung ein clientseitig erzeugtes Zertifikat (CEZ) verwendet wird. (Parameter *zertifikat*)

Das zurückgegebene Handle des Empfangsobjekts wird der Funktion [OttoEmpfangFortsetzen\(\)](#) übergeben, um Daten blockweise abzuholen. Sind alle Daten abgeholt, wird [OttoEmpfangBeenden\(\)](#) aufgerufen, womit das Empfangsobjekt wieder freigegeben wird.

Ein wichtiger Unterschied zu [OttoEmpfangBeginnen\(\)](#) besteht darin, dass der OTTER-Server die Daten auf den in *abholzertifikat* enthaltenen öffentlichen Schlüssel umschlüsselt. Die Daten werden vom Otto nicht entschlüsselt und [OttoEmpfangFortsetzen\(\)](#) gibt lediglich die verschlüsselten Daten zurück.

Zu beachten:

Wurde eine Otto-Instanz vor dem Aufruf dieser Funktion mit [OttoProxyKonfigurationSetzen\(\)](#) für einen Proxy konfiguriert, wird der Empfang über den Proxy durchgeführt. Die Proxy-Konfiguration wird intern an dem Empfangsobjekt gespeichert und spätere Aufrufe von [OttoProxyKonfigurationSetzen\(\)](#) haben keinen Einfluss auf den bereits begonnenen Empfang.

Parameter:

in	<i>instanz</i>	Handle der Otto-Instanz, auf der diese Funktion ausgeführt werden soll.
in	<i>objektId</i>	ID des Objekts, das vom OTTER-Server abgeholt werden soll.
in	<i>zertifikat</i>	Handle auf ein Zertifikatsobjekt Es darf hier kein clientseitig erzeugtes Zertifikat (CEZ) angegeben werden.
in	<i>herstellerId</i>	Hersteller-ID des Softwareproduktes
in	<i>abholzertifikat</i>	Base64-kodierter Teil eines X.509-v3-Zertifikats im PEM-Format
out	<i>empfang</i>	Handle auf das Empfangsobjekt. Im Fehlerfall wird kein Empfangsobjekt erzeugt.

Rückgabe:

- siehe [otto_statuscode.h](#)

Siehe auch:

- [OttoEmpfangFortsetzen\(\)](#)
- [OttoEmpfangBeenden\(\)](#)
- [OttoProxyKonfigurationSetzen\(\)](#)

OttoStatusCode OttoEmpfangFortsetzen (OttoEmpfangHandle empfang, OttoRueckgabepufferHandle datenBlock)

Empfängt einen Datenblock vom OTTER-Server.

Otto empfängt Daten vom OTTER-Server und gibt sie blockweise an den Aufrufer zurück. Wird `OTTO_OK` zurückgegeben, kann diese Funktion erneut aufgerufen werden und weitere Datenblöcke empfangen werden. Werden leere Daten zurückgegeben, ist der Empfang beendet und alle Daten wurden empfangen. Dann muss [OttoEmpfangBeenden\(\)](#) aufgerufen werden.

Parameter:

in	<i>empfang</i>	Ein mit OttoEmpfangBeginnen() erzeugtes Handle.
out	<i>datenBlock</i>	Rückgabepuffer mit allen oder einem Teil der empfangenen Daten. Falls leer, ist der Empfang beendet. Der Inhalt des Rückgabepuffers darf nicht als null-terminierte Zeichenkette interpretiert werden, da die empfangenen Daten weitere Null-Bytes enthalten können.

Rückgabe:

- [OTTO_OK](#)
- [OTTO_TRANSFER_UNAUTHORIZED](#)
- [OTTO_TRANSFER_NOT_FOUND](#)
- weitere, siehe [otto_statuscode.h](#)

Siehe auch:

- [OttoEmpfangBeenden\(\)](#)

const char * OttoHoleFehlertext ([OttoStatusCode](#) statuscode)

Die Funktion liefert die Klartextfehlermeldung zu einem Otto-Statuscode -
definiert in [otto_statuscode.h](#).

Parameter:

in	<i>statuscode</i>	Statuscode
----	-------------------	------------

Rückgabe:

- Zeiger auf einen statischen Puffer mit der Klartextmeldung zu einem Statuscode als null-terminierte, UTF-8-kodierte Zeichenkette.
- `NULL` , falls kein Text ermittelt werden konnte.

OttoStatusCode OttoInstanzErzeugen (const byteChar * logPfad, OttoLogCallback logCallback, void * logCallbackBenutzerdaten, OttoInstanzHandle * instanz)

Erstellt und initialisiert eine neue Otto-Instanz.

Otto-Instanzen sind nicht an ihre Ersteller-Threads gebunden. Sie dürfen zwar *nicht gleichzeitig* in mehreren Threads verwendet werden, aber sie dürfen wechselnd von verschiedenen Threads verwendet werden. Das heißt insbesondere, dass sie von neuen Threads wiederverwendet werden können.

Otto-Instanzen sind in dem Sinne threadsicher, dass verschiedene Otto-Instanzen zeitgleich in verschiedenen Threads verwendet werden können. Jedoch darf ein- und dieselbe Otto-Instanz nicht zeitgleich in mehreren Threads verwendet werden.

Parameter:

in	<i>logPfad</i>	Optionaler Pfad zur Log-Datei otto.log. Ist der Wert gleich <code>NULL</code> , wird das betriebssystemspezifische Verzeichnis für temporäre Dateien verwendet.
in	<i>logCallback</i>	Callback-Funktion, die gegebenenfalls von Otto bei der Protokollierung von Meldungen aufgerufen wird. Siehe OttoLogCallback . Der Parameter darf <code>NULL</code> sein.
in	<i>logCallbackBenutzerdaten</i>	Beliebiger Zeiger auf Daten, den Otto beim Aufruf eines <code>logCallback</code> an den Callback weiterreicht. Über diesen Weg kann sich eine Anwendung eigene Daten an ihre Log-Callback-Funktion übergeben lassen. Der Parameter darf <code>NULL</code> sein.
out	<i>instanz</i>	Handle der erzeugten Otto-Instanz

Zu beachten:

Kann kein otto.log angelegt werden, wird eine entsprechende Fehlermeldung auf die Konsole (stderr) geschrieben und an den Windows-Ereignisdienst bzw. den syslogd-Dienst (Linux, AIX, macOS) geschickt.

Für Linux, AIX und macOS ist zu beachten, dass der syslogd-Dienst gegebenenfalls erst noch zu aktivieren und für die Protokollierung von Meldungen der Facility "User" zu konfigurieren ist. Suchkriterien für Otto-Meldungen in der Windows-Ereignisansicht sind "ERiC (Elster Rich Client)" als Quelle und "Anwendung" als Protokoll.

Suchkriterien für ERiC-Meldungen in den Systemlogdateien unter Linux, AIX und macOS sind die Facility "User" und der Ident "ERiC (Elster Rich Client)".

Rückgabe:

- siehe [otto_statuscode.h](#)

Siehe auch:

- [OttoInstanzFreigeben\(\)](#)
- [OttoLogCallback](#)

OttoStatusCode OttoInstanzFreigeben (OttoInstanzHandle instanz)

Gibt eine Otto-Instanz frei.

Die freigegebene Otto-Instanz sowie alle eventuell noch daran gebundenen Objekte dürfen nach der Freigabe nicht mehr verwendet werden.

Parameter:

in	<i>instanz</i>	Handle der Otto-Instanz, die freigegeben werden soll.
----	----------------	---

Rückgabe:

- siehe [otto_statuscode.h](#)

Siehe auch:

- [OttoInstanzErzeugen\(\)](#)

[OttoStatusCode](#) **OttoProxyKonfigurationSetzen** ([OttoInstanzHandle](#) instanz, const [OttoProxyKonfiguration](#) * proxyKonfiguration)

Konfiguriert eine Otto-Instanz für einen Proxy.

Damit eine Otto-Instanz ihre Internetverbindungen über einen Proxy aufbaut, muss ihr die Proxy-Konfiguration über diese Methode mitgeteilt werden. Die Konfiguration gilt dann für alle Verbindungen der Instanz nach außen, d.h. für die Verbindungen zu den OTTER-Servern ebenso wie für Verbindungen zum ELSTER-eID-Server bei der Verwendung eines elektronischen Personalausweises oder Aufenthaltstitels.

Parameter:

in	<i>instanz</i>	Die Otto-Instanz, für die die Konfiguration gelten soll.
in	<i>proxyKonfiguration</i>	Die Proxy-Konfiguration, die von der Otto-Instanz verwendet werden soll. Wenn hier <code>NULL</code> übergeben wird, verwendet die Otto-Instanz keinen Proxy.

Rückgabe:

- siehe [otto_statuscode.h](#)

Siehe auch:

- [OttoProxyKonfiguration](#)

[OttoStatusCode](#) [OttoPruefsummeAktualisieren](#) ([OttoPruefsummeHandle](#) pruefsumme, const [byteChar](#) * datenBlock, uint64_t datenBlockGroesse)

Aktualisiert die Prüfsumme über Daten. Eine Prüfsumme, die bereits signiert wurde, kann nicht mehr aktualisiert werden.

Parameter:

in,out	<i>pruefsumme</i>	Handle der Prüfsumme, die aktualisiert werden soll.
in	<i>datenBlock</i>	Zeiger auf die Daten, über die die Prüfsumme aktualisiert werden soll.
in	<i>datenBlockGroesse</i>	Größe der Daten, über die die Prüfsumme aktualisiert werden soll, in Bytes.

Rückgabe:

- [OTTO_OK](#) wenn die Prüfsumme erfolgreich aktualisiert werden konnte
- [OTTO_PRUEFSUMME_FINALISIERT](#) wenn die Prüfsumme bereits signiert wurde
- weitere, siehe [otto_statuscode.h](#)

Siehe auch:

- [OttoPruefsummeErzeugen\(\)](#)
- [OttoPruefsummeSignieren\(\)](#)
- [OttoPruefsummeFreigeben\(\)](#)

OttoStatusCode OttoPruefsummeErzeugen (OttoInstanzHandle instanz, OttoPruefsummeHandle * pruefsumme)

Erzeugt ein Objekt zur Berechnung einer Datenprüfsumme, die Otto zu Beginn einer Übermittlung an den OTTER-Server senden muss.

Das Prüfsummenobjekt ist an die Otto-Instanz gebunden, für die es erzeugt wurde und darf nicht zusammen mit einer anderen Otto-Instanz oder mit Objekten anderen Otto-Instanzen verwendet werden.

Parameter:

in	<i>instanz</i>	Handle der Otto-Instanz, für die das Prüfsummenobjekt erzeugt werden soll.
out	<i>pruefsumme</i>	Handle des erzeugten Prüfsummenobjekts

Rückgabe:

- siehe [otto_statuscode.h](#)

Siehe auch:

- [OttoPruefsummeAktualisieren\(\)](#)
- [OttoPruefsummeSignieren\(\)](#)
- [OttoPruefsummeFreigeben\(\)](#)

OttoStatusCode OttoPruefsummeFreigeben (OttoPruefsummeHandle pruefsumme)

Gibt ein Prüfsummenobjekt frei.

Das Prüfsummenobjekt darf danach nicht wieder verwendet werden.

Parameter:

in	<i>pruefsumme</i>	Handle des Prüfsummenobjekts, das freigegeben werden soll.
----	-------------------	--

Rückgabe:

- siehe [otto_statuscode.h](#)

Siehe auch:

- [OttoPruefsummeErzeugen\(\)](#)
- [OttoPruefsummeAktualisieren\(\)](#)
- [OttoPruefsummeSignieren\(\)](#)

[OttoStatusCode](#) OttoPruefsummeSignieren ([OttoPruefsummeHandle](#) pruefsumme, [OttoZertifikatHandle](#) zertifikat, [OttoRueckgabepufferHandle](#) rueckgabepuffer)

Erstellt eine Signatur über eine Prüfsumme.

Die Signierung der Prüfsumme ist nur dann möglich, wenn diese über die Mindestdatenmenge für eine Übermittlung an den OTTER-Server berechnet wurde. (20 MiB) Eine Prüfsumme kann nur einmalig signiert werden. Danach muß das Prüfsummenobjekt freigegeben werden.

Parameter:

in	<i>pruefsumme</i>	Handle der Prüfsumme, die signiert werden soll.
in	<i>zertifikat</i>	Handle des Sicherheitstoken, mit dem die Prüfsumme signiert werden soll.
out	<i>rueckgabepuffer</i>	Handle des Rückgabepuffers, in den die signierte Prüfsumme geschrieben werden soll. Die signierte Prüfsumme wird als base64-codierte Zeichenfolge übergeben.

Rückgabe:

- [OTTO_OK](#) wenn die Prüfsumme signiert werden konnte
- [OTTO_PRUEFSUMME_FINALISIERT](#) wenn die Prüfsumme bereits signiert wurde
- [OTTO_VERSAND_GERINGE_DATENMENGE](#) wenn die Prüfsumme über weniger Daten gebildet wurde als für den Versand an den OTTER-Server erforderlich sind
- [OTTO_ESIGNER_NICHT_GELADEN](#) wenn die Signaturkomponente eSigner nicht geladen werden konnte
- [OTTO_ESIGNER_VERALTET](#) wenn die vorliegende Version der Signaturkomponente eSigner zu alt ist
- [OTTO_ESIGNER_INKOMPATIBEL](#) wenn die vorliegende Signaturkomponente eSigner zur Otto-Bibliothek nicht kompatibel ist
- Fehler der Signaturkomponente eSigner aus dem Statuscodebereich ab [OTTO_ESIGNER_BUSY](#) = 610405801
- weitere, siehe [otto_statuscode.h](#)

Siehe auch:

- [OttoPruefsummeErzeugen\(\)](#)
- [OttoPruefsummeAktualisieren\(\)](#)
- [OttoPruefsummeFreigegeben\(\)](#)

[OttoStatusCode](#) [OttoRueckgabepufferErzeugen](#) ([OttoInstanzHandle](#) instanz, [OttoRueckgabepufferHandle](#) * rueckgabepuffer)

Erzeugt einen Rückgabepuffer und gibt ein Handle darauf zurück.

Die von dieser Funktion erzeugten Rückgabepuffer werden verwendet, um die Rückgabedaten von Otto-Funktionen (z. B. [OttoEmpfangFortsetzen\(\)](#) oder [OttoVersandBeenden\(\)](#)) aufzunehmen. Dazu wird das Rückgabepuffer-Handle für den Schreibvorgang an die ausgebende Funktion übergeben.

Zum Auslesen des von den API-Funktionen beschriebenen Puffers wird das Rückgabepuffer-Handle an [OttoRueckgabepufferInhalt\(\)](#) übergeben. Ein einmal erzeugtes Rückgabepuffer-Handle kann für weitere nachfolgende Aufrufe von Otto API-Funktionen wiederverwendet werden. Bei einer Wiederverwendung eines Handles werden frühere Inhalte überschrieben. Nach letztmaliger Verwendung muss jeder Rückgabepuffer mit [OttoRueckgabepufferFreigeben\(\)](#) freigegeben werden.

Der Rückgabepuffer ist an die Otto-Instanz gebunden, für die er erzeugt wurde und kann nicht zusammen mit einer anderen Otto-Instanz oder mit Objekten anderen Otto-Instanzen verwendet werden.

Parameter:

in	<i>instanz</i>	Handle der Otto-Instanz, auf der diese Funktion ausgeführt werden soll.
out	<i>rueckgabepuffer</i>	Zeiger auf das Handle des erzeugten Rückgabepuffers

Rückgabe:

- siehe [otto_statuscode.h](#)

Siehe auch:

- [OttoRueckgabepufferGroesse\(\)](#)
- [OttoRueckgabepufferInhalt\(\)](#)
- [OttoRueckgabepufferFreigeben\(\)](#)

[OttoStatusCode](#) OttoRueckgabepufferFreigeben ([OttoRueckgabepufferHandle](#) rueckgabepuffer)

Gibt einen Rückgabepuffer frei.

Das Handle des Rückgabepuffers darf danach nicht weiter verwendet werden. Es wird daher empfohlen, Handle-Variablen nach der Freigabe explizit auf `NULL` zu setzen.

Parameter:

in	<i>rueckgabepuffer</i>	Handle auf den Rückgabepuffer, der freigegeben werden soll. Dieser Rückgabepuffer darf nicht bereits freigegeben worden sein.
----	------------------------	---

Rückgabe:

- siehe [otto_statuscode.h](#)

Siehe auch:

- [OttoRueckgabepufferGroesse\(\)](#)
- [OttoRueckgabepufferInhalt\(\)](#)
- [OttoRueckgabepufferErzeugen\(\)](#)

uint64_t OttoRueckgabepufferGroesse ([OttoRueckgabepufferHandle](#) rueckgabepuffer)

Gibt die Anzahl der im Rückgabepuffer enthaltenen Bytes zurück. Das abschließende Null-Byte wird nicht mitgezählt.

Parameter:

in	<i>rueckgabepuffer</i>	Das Handle des Rückgabepuffers
----	------------------------	--------------------------------

Rückgabe:

- Anzahl der im Rückgabepuffer enthaltenen Bytes, wenn ein gültiges Handle übergeben wird.
- 0 sonst

Siehe auch:

- [OttoRueckgabepufferInhalt\(\)](#)
- [OttoRueckgabepufferErzeugen\(\)](#)
- [OttoRueckgabepufferFreigeben\(\)](#)

const [byteChar](#) * OttoRueckgabepufferInhalt ([OttoRueckgabepufferHandle](#) rueckgabepuffer)

Gibt den Inhalt eines Rückgabepuffers zurück.

Der zurückgegebene Zeiger verweist auf ein Byte-Array, das alle in den Rückgabepuffer geschriebenen Bytes enthält. Dieses Array existiert so lange im Speicher, bis der Rückgabepuffer entweder (bei einer Wiederverwendung des Handles) erneut beschrieben oder der Puffer explizit freigegeben wird. Der Array wird immer von einem Null-Byte abgeschlossen. Wenn der Rückgabepuffer keine weiteren Null-Bytes enthält, kann folglich der Rückgabepufferinhalt bequem als null-terminierte Zeichenkette interpretiert werden.

Parameter:

in	<i>rueckgabepuffer</i>	Das Handle des Rückgabepuffers, dessen Inhalt zurückgegeben werden soll.
----	------------------------	--

Rückgabe:

- Zeiger auf den Rückgabepufferinhalt, wenn ein gültiges Handle übergeben wird.
- NULL sonst

Siehe auch:

- [OttoRueckgabepufferGroesse\(\)](#)
- [OttoRueckgabepufferErzeugen\(\)](#)
- [OttoRueckgabepufferFreigegeben\(\)](#)

[OttoStatusCode](#) **OttoVersandAbschliessen** ([OttoVersandHandle](#) versand, [OttoRueckgabepufferHandle](#) objektId)

Schließt einen Versand ab und gibt die Objekt-ID zurück.

Mit dieser Funktion wird das Ende der Daten gekennzeichnet und der Datenversand abgeschlossen.

Im Erfolgsfall wird die vom OTTER-Server vergebene Objekt-ID zurückgegeben, über die die versendeten Daten bei OTTER referenziert werden.

Parameter:

in	<i>versand</i>	Ein mit OttoVersandBeginnen() erzeugtes Handle
out	<i>objektId</i>	Handle des Rückgabepuffers, in den die Objekt-ID geschrieben werden soll.

Rückgabe:

- [OTTO_OK](#) im Erfolgsfall
- [OTTO_TRANSFER_UNAUTHORIZED](#)
- [OTTO_TRANSFER_CONNECTSERVER](#)
- [OTTO_VERSAND_GERINGE_DATENMENGE](#)
- [OTTO_VERSAND_ABGESCHLOSSEN](#) falls [OttoVersandAbschliessen\(\)](#) bereits aufgerufen wurde
- weitere, siehe [otto_statuscode.h](#)

OttoStatusCode OttoVersandBeenden (OttoVersandHandle versand)

Gibt ein Versandobjekt frei.

Das Versandobjekt darf danach nicht wieder verwendet werden.

Parameter:

in	<i>versand</i>	Handle des Versandobjekts, das freigegeben werden soll.
----	----------------	---

Rückgabe:

- siehe [otto_statuscode.h](#)

Siehe auch:

- [OttoVersandBeginnen\(\)](#)
- [OttoVersandFortsetzen\(\)](#)
- [OttoVersandAbschliessen\(\)](#)

[OttoStatusCode](#) [OttoVersandBeginnen](#) ([OttoInstanzHandle](#) instanz, const [byteChar](#) * signiertePruefsumme, const [byteChar](#) * herstellerId, [OttoVersandHandle](#) * versand)

Initialisiert einen Datenversand an den OTTER-Server.

Das zurückgegebene Handle des Versandobjekts wird der Funktion [OttoVersandFortsetzen\(\)](#) übergeben, um Daten blockweise hochzuladen. Sind alle Daten versendet, ist [OttoVersandAbschliessen\(\)](#) aufzurufen, womit der Versand abgeschlossen wird. Zum Freigeben des Versandobjekts ist [OttoVersandBeenden\(\)](#) aufzurufen.

Bevor der Versand begonnen werden kann, muss eine Prüfsumme über alle zu versendenden Daten gebildet (siehe [OttoPruefsummeErzeugen\(\)](#)) und mit [OttoPruefsummeSignieren\(\)](#) signiert werden.

Zu beachten:

Wurde Otto vor dem Aufruf dieser Funktion für einen Proxy mit [OttoProxyKonfigurationSetzen\(\)](#) konfiguriert, wird der Versand über den Proxy durchgeführt. Die Proxy-Konfiguration wird intern an dem Versandobjekt gespeichert und spätere Aufrufe von [OttoProxyKonfigurationSetzen\(\)](#) haben keinen Einfluss auf den bereits begonnenen Versand.

Parameter:

in	<i>instanz</i>	Handle der Otto-Instanz, auf der diese Funktion ausgeführt werden soll.
in	<i>signiertePruefsumme</i>	Signierte Prüfsumme über die Gesamtheit der Daten, die in diesem Versand versendet werden sollen. Die signierte Prüfsumme wird als base64-codierte, nullterminierte Zeichenfolge erwartet, wie sie von OttoPruefsummeSignieren() zurückgeliefert wird.
in	<i>herstellerId</i>	Hersteller-ID des Softwareproduktes
out	<i>versand</i>	Handle auf das Versandobjekt. Im Fehlerfall wird kein Versandobjekt erzeugt.

Rückgabe:

- siehe [otto_statuscode.h](#)

Siehe auch:

- [OttoVersandFortsetzen\(\)](#)
- [OttoVersandAbschliessen\(\)](#)
- [OttoVersandBeenden\(\)](#)
- [OttoPruefsummeSignieren\(\)](#)
- [OttoProxyKonfigurationSetzen\(\)](#)

[OttoStatusCode](#) [OttoVersandFortsetzen](#) ([OttoVersandHandle](#) versand, const [byteChar](#) * datenBlock, uint64_t datenBlockGroe

ssse)
Versendet einen Datenblock an den OTTER-Server.

Otto liest den übergebenen Datenblock ein und versendet ihn an den OTTER-Server. Wenn `OTTO_OK` zurückgegeben wird, kann diese Funktion erneut mit einem weiteren Datenblock aufgerufen werden. Dies ist zu wiederholen, bis Otto alle zu diesem Versand gehörigen Daten erhalten hat. Falls nicht `OTTO_OK` zurückgegeben wird, ist der Versand fehlgeschlagen.

Ist das Ende der Daten erreicht, muss [OttoVersandAbschliessen\(\)](#) aufgerufen werden.

Parameter:

in	<i>versand</i>	Ein mit OttoVersandBeginnen() erzeugtes Handle
in	<i>datenBlock</i>	Zeiger auf die zu versendenden Daten. Falls <code>NULL</code> wird der Aufruf ignoriert.
in	<i>datenBlockGroe</i> <i>sse</i>	Größe des Arrays <i>datenBlock</i> in Bytes. Falls 0 wird der Aufruf ignoriert.

Rückgabe:

- [OTTO_OK](#) im Erfolgsfall
- [OTTO_TRANSFER_UNAUTHORIZED](#)
- [OTTO_TRANSFER_CONNECTSERVER](#)
- [OTTO_VERSAND_ABGESCHLOSSEN](#) falls [OttoVersandAbschliessen\(\)](#) bereits aufgerufen wurde
- weitere, siehe [otto_statuscode.h](#)

Siehe auch:

- [OttoVersandAbschliessen\(\)](#)

OttoStatusCode OttoVersion (OttoRueckgabepufferHandle rueckgabepuffer)

Gibt die Version der Otto-Bibliothek zurück.

Zu beachten:

Die Version der Otto-Bibliothek ist nicht zwingend gleich der Version des ERiC-Auslieferungspaketes, sondern kann davon abweichen.

Rückgabe:

- siehe [otto_statuscode.h](#)

[OttoStatusCode](#) [OttoZertifikatOeffnen](#) ([OttoInstanzHandle](#) instanz, const [byteChar](#) * zertifikatsPfad, const [byteChar](#) * zertifikatsPasswort, [OttoZertifikatHandle](#) * zertifikat)

Erstellt ein Otto-Zertifikatsobjekt für ein Sicherheitstoken.

Das Zertifikatsobjekt ist an die Otto-Instanz gebunden, für die es erzeugt wurde, und darf nicht zusammen mit einer anderen Otto-Instanz oder mit Objekten anderer Otto-Instanzen verwendet werden. Soll ein Sicherheitstoken von mehreren Otto-Instanzen verwendet werden, so sind hierfür mehrere Zertifikatsobjekte zu erstellen: für jede Instanz eines.

Parameter:

in	<i>instanz</i>	Handle der Otto-Instanz, die das Zertifikatsobjekt verwenden soll.
----	----------------	--

in	<i>zertifikatsPfad</i>	<p>Pfad zum Sicherheitstoken, folgende Angaben sind möglich:</p> <ol style="list-style-type: none"> 1. Clientseitig erzeugtes Zertifikat: Pfad Verzeichnis, in dem sich die Zertifikats-Datei (.cer) und die Datei mit dem privaten Schlüssel (.p12) befinden. Diese Sicherheitstokens wurden mit EricMtCreateKey() bzw. EricCreateKey() erzeugt. Der Pfad zum Verzeichnis ist bei clientseitig erzeugten Zertifikaten relativ zum aktuellen Arbeitsverzeichnis oder absolut anzugeben. 2. Software-Portalzertifikat: Pfad zur Software-Zertifikatsdatei (i.d.R. mit der Endung .pfx). Der Pfad zur Datei ist bei Software-Zertifikaten relativ zum aktuellen Arbeitsverzeichnis oder absolut anzugeben. 3. Sicherheitsstick: Pfad zur Treiberdatei, siehe (*). Bitte beachten, dass der Treiber betriebssystemabhängig sein kann. Weitere Informationen in der Anleitung zum Sicherheitsstick oder unter https://www.sicherheitsstick.de. 4. Signaturkarte: (**) Pfad zur Treiberdatei, welcher einen Zugriff auf die Signaturkarte ermöglicht, siehe (*). Weitere Informationen in der Anleitung zur Signaturkarte. 5. Elektronischer Personalausweis (nPA) oder Aufenthaltstitel (eAT): Die URL des eID-Clients wie zum Beispiel der AusweisApp 2. In den meisten Fällen lautet diese URL: http://127.0.0.1:24727/eID-Client Optional kann auf die folgende Weise noch ein Testmerker angehängt werden: http://127.0.0.1:24727/eID-Client?testmerker=520000000. Zu den verfügbaren Testmerkern siehe ERIC-Entwicklerhandbuch.pdf, Kap. "Testunterstützung bei der ERiC-Anbindung". <p>Wichtig: Das Ad-hoc-Zertifikat, das in diesem Fall für den elektronischen Personalausweis erzeugt wird, ist nur 24 Stunden gültig.</p>
----	------------------------	--

(*) Wird der Dateipfad eines Treibers angegeben, ist der Suchmechanismus zu beachten, mit dem das jeweilige Betriebssystem dynamische Bibliotheken lädt. Weitere Informationen sind der Systemdokumentation zu den Betriebssystemfunktionen `LoadLibrary()` (Windows) bzw. `dlopen()` (Linux, AIX und macOS) zu entnehmen.

(**) Bei Signaturkarten erfolgt eine PIN-Abfrage nicht beim Aufruf von [OttoZertifikatOeffnen\(\)](#), sondern beim Aufruf von [OttoPruefsummeSignieren\(\)](#), [OttoEmpfangBeginnen\(\)](#) und [OttoEmpfangBeginnenAbholzertifikat\(\)](#).

Pfade müssen auf Windows in der für Datei-Funktionen benutzten ANSI-Codepage, auf Linux, AIX und Linux Power in der für das Dateisystem benutzten Locale und auf macOS in der "decomposed form" von UTF-8 übergeben werden. Bitte weitere Betriebssystemspezifika bzgl. nicht erlaubter Zeichen in Pfaden und Pfadtrennzeichen beachten.

Für Details zu Pfaden im ERiC siehe [ERiC-Entwicklerhandbuch.pdf](#), Kapitel "Übergabe von Pfaden an ERiC API-Funktionen".

Parameter:

in	<i>zertifikatsPasswort</i>	Das Passwort oder die PIN des Sicherheitstokens. Bei Tokens, bei denen das Passwort oder die PIN nicht von der Anwendung übergeben, sondern separat über einen Treiber (z. B. von einem Kartenlesegerät) abgefragt wird, ist hier NULL zu übergeben.
out	<i>zertifikat</i>	Handle auf das erstellte Zertifikatsobjekt

Rückgabe:

- siehe [otto_statuscode.h](#)

Siehe auch:

- [OttoZertifikatSchliessen\(\)](#)
- [OttoZertifikatOeffnenAusBytes\(\)](#)

[OttoStatusCode](#) [OttoZertifikatOeffnenAusBytes](#) ([OttoInstanzHandle](#) instanz, const [byteChar](#) * pkcs12Container, [uint32_t](#) containerGroesse, const [byteChar](#) * zertifikatsPasswort, [OttoZertifikatHandle](#) * zertifikat)

Erstellt ein Otto-Zertifikatsobjekt für ein PKCS#12-Sicherheitstoken, das im Hauptspeicher übergeben wird.

Das Zertifikatsobjekt ist an die Otto-Instanz gebunden, für die es erzeugt wurde, und darf nicht zusammen mit einer anderen Otto-Instanz oder mit Objekten anderer Otto-Instanzen verwendet werden. Soll ein PKCS#12-Container von mehreren Otto-Instanzen verwendet werden, so sind hierfür mehrere Zertifikatsobjekte zu erstellen: für jede Instanz eines.

Parameter:

in	<i>instanz</i>	Handle der Otto-Instanz, die das Zertifikatsobjekt verwenden soll.
in	<i>pkcs12Container</i>	Adresse des PKCS#12-Containers im Hauptspeicher. Es werden nur passwortgeschützte PKCS#12-Container akzeptiert.
in	<i>containerGroesse</i>	Größe des PKCS#12-Containers in Bytes
in	<i>zertifikatsPasswort</i>	Das Passwort oder die PIN des PKCS#12-Containers. Dieser Parameter darf nicht NULL sein.
out	<i>zertifikat</i>	Handle auf das erstellte Zertifikatsobjekt

Rückgabe:

- siehe [otto_statuscode.h](#)

Siehe auch:

- [OttoZertifikatSchliessen\(\)](#)
- [OttoZertifikatOeffnen\(\)](#)

[OttoStatusCode](#) **OttoZertifikatSchliessen** ([OttoZertifikatHandle](#) *zertifikat*)

Schließt das Otto-Zertifikatsobjekt zu einem Sicherheitstoken. Anschließend darf das Zertifikatsobjekt nicht mehr verwendet werden.

Parameter:

in	<i>zertifikat</i>	Handle auf das Zertifikatsobjekt, das geschlossen werden soll.
----	-------------------	--

Rückgabe:

- siehe [otto_statuscode.h](#)

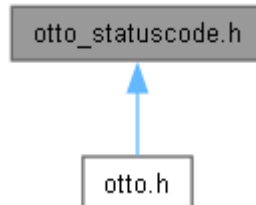
Siehe auch:

- [OttoZertifikatOeffnen\(\)](#)

otto_statuscode.h-Dateireferenz

Auflistung der Otto-Statuscodes.

Dieser Graph zeigt, welche Datei direkt oder indirekt diese Datei enthält:



Aufzählungen

- enum [OttoStatusCode](#) { [OTTO_OK](#) = 0, [OTTO_INTERNER FEHLER](#) = 610401001, [OTTO UNBEKANNTER FEHLER](#) = 610401002, [OTTO NPA ZERTIFIKATFEHLER](#) = 610401003, [OTTO TRANSFER FEHLER](#) = 610403001, [OTTO TRANSFER INIT](#) = 610403002, [OTTO TRANSFER CONNECTSERVER](#) = 610403003, [OTTO TRANSFER CONNECTPROXY](#) = 610403004, [OTTO TRANSFER TIMEOUT](#) = 610403005, [OTTO TRANSFER PROXYAUTH](#) = 610403006, [OTTO TRANSFER UNAUTHORIZED](#) = 610403007, [OTTO TRANSFER NOT FOUND](#) = 610403008, [OTTO TRANSFER SERVER FEHLER](#) = 610403009, [OTTO TRANSFER DECODING](#) = 610403010, [OTTO TRANSFER EID ZERTIFIKATFEHLER](#) = 610403011, [OTTO TRANSFER EID KEINCLIENT](#) = 610403012, [OTTO TRANSFER EID KEINKONTO](#) = 610403013, [OTTO TRANSFER EID CLIENTFEHLER](#) = 610403014, [OTTO TRANSFER EID NPABLOCKIERT](#) = 610403015, [OTTO UNGUELTIGER PARAMETER](#) = 610405001, [OTTO UNGUELTIGES HANDLE](#) = 610405002, [OTTO MEHRFACHAUFRUFE NICHT UNTERSTUETZT](#) = 610405003, [OTTO INSTANZEN INKONSISTENT](#) = 610405004, [OTTO INSTANZ UNTEROBJEKTE NICHT FREIGEgeben](#) = 610405005, [OTTO LOG FEHLER](#) = 610405006, [OTTO FUNKTION NICHT UNTERSTUETZT](#) = 610405007, [OTTO ZERTIFIKAT PIN FALSCH](#) = 610405008, [OTTO ZERTIFIKAT PFAD FALSCH](#) = 610405009, [OTTO ZERTIFIKAT NICHT ERKANNT](#) = 610405010, [OTTO PRUEFSUMME FINALISIERT](#) = 610405011,

[OTTO UNGUELTIGE HERSTELLERID](#) = 610405012,
[OTTO EMPFANG VORZEITIG BEENDET](#) = 610405013,
[OTTO VERSAND GERINGE DATENMENGE](#) = 610405014,
[OTTO ESIGNER NICHT GELADEN](#) = 610405015,
[OTTO ESIGNER VERALTET](#) = 610405016, [OTTO ESIGNER INKOMPATIBEL](#)
= 610405017, [OTTO PROXY URL](#) = 610405018, [OTTO PROXY PORT](#) =
610405019, [OTTO PROXY AUTHSCHEMA](#) = 610405020,
[OTTO VERSAND ABGESCHLOSSEN](#) = 610405021,
[OTTO VERSAND ZU GROSSE DATENMENGE](#) = 610405022,
[OTTO EINSTELLUNG UNBEKANNT](#) = 610405023,
[OTTO EINSTELLUNG WERT UNGUELTIG](#) = 610405024,
[OTTO ESIGNER BUSY](#) = 610405801, [OTTO ESIGNER DECRYPT](#) =
610405802, [OTTO ESIGNER ENCRYPT](#) = 610405803,
[OTTO ESIGNER ENCODE ERROR](#) = 610405804,
[OTTO ESIGNER ENCODE UNKNOWN](#) = 610405805,
[OTTO ESIGNER ESICL EXCEPTION](#) = 610405806,
[OTTO ESIGNER INVALID HANDLE](#) = 610405807,
[OTTO ESIGNER LOAD DLL](#) = 610405808, [OTTO ESIGNER MAX SESSION](#)
= 610405809, [OTTO ESIGNER NO SERVICE](#) = 610405810,
[OTTO ESIGNER NO SIG ENC KEY](#) = 610405811,
[OTTO ESIGNER OUT OF MEM](#) = 610405812,
[OTTO ESIGNER P11 ENC KEY](#) = 610405813,
[OTTO ESIGNER P11 ENGINE LOADED](#) = 610405814,
[OTTO ESIGNER P11 INIT FAILED](#) = 610405815,
[OTTO ESIGNER P11 NO ENC CERT](#) = 610405816,
[OTTO ESIGNER P11 NO SIG CERT](#) = 610405817,
[OTTO ESIGNER P11 SIG KEY](#) = 610405818,
[OTTO ESIGNER P11 SLOT EMPTY](#) = 610405819,
[OTTO ESIGNER P12 CREATE](#) = 610405820,
[OTTO ESIGNER P12 DECODE](#) = 610405821,
[OTTO ESIGNER P12 ENC KEY](#) = 610405822,
[OTTO ESIGNER P12 SIG KEY](#) = 610405823,
[OTTO ESIGNER P12 NO ENC CERT](#) = 610405824,
[OTTO ESIGNER P12 NO SIG CERT](#) = 610405825,
[OTTO ESIGNER P12 READ](#) = 610405826, [OTTO ESIGNER P7 DECODE](#) =
610405827, [OTTO ESIGNER P7 READ](#) = 610405828,
[OTTO ESIGNER P7 RECIPIENT](#) = 610405829,
[OTTO ESIGNER PIN LOCKED](#) = 610405830, [OTTO ESIGNER PIN WRONG](#)
= 610405831, [OTTO ESIGNER PSE PATH](#) = 610405832,
[OTTO ESIGNER SC ENC KEY](#) = 610405833,
[OTTO ESIGNER SC INIT FAILED](#) = 610405834,
[OTTO ESIGNER SC NO APPLLET](#) = 610405835,
[OTTO ESIGNER SC NO ENC CERT](#) = 610405836,
[OTTO ESIGNER SC NO SIG CERT](#) = 610405837,

[OTTO_ESIGNER_SC_SESSION](#) = 610405838,
[OTTO_ESIGNER_SC_SIG_KEY](#) = 610405839,
[OTTO_ESIGNER_SC_SLOT_EMPTY](#) = 610405840,
[OTTO_ESIGNER_TOKEN_TYPE_MISMATCH](#) = 610405841,
[OTTO_ESIGNER_USER_CANCEL](#) = 610405842,
[OTTO_ESIGNER_VERIFY_CERT_CHAIN](#) = 610405843,
[OTTO_ESIGNER_DATA_NOT_INITIALIZED](#) = 610405844,
[OTTO_ESIGNER_ASN1_READ_BUFFER_TOO_SMALL](#) = 610405845,
[OTTO_ESIGNER_ASN1_READ_DATA_INCOMPLETE](#) = 610405846,
[OTTO_ESIGNER_ASN1_NO_ENVELOPED_DATA](#) = 610405847,
[OTTO_ESIGNER_ASN1_NO_CONTENT_DATA](#) = 610405848,
[OTTO_INIDATEI_LESEFEHLER](#) = 610407001,
[OTTO_ZERTIFIKAT_LESEFEHLER](#) = 610407002,
[OTTO_ZERTIFIKAT_DEFEKT](#) = 610407003,
[OTTO_ZERTIFIKAT_FINGERABDRUCK_FEHLER](#) = 610407004,
[OTTO_SIGNIEREN_FEHLGESCHLAGEN](#) = 610407005,
[OTTO_ENTSCHLUESSELN_FEHLGESCHLAGEN](#) = 610407006,
[OTTO_DEKOMPRESSION_FEHLGESCHLAGEN](#) = 610407007,
[OTTO_NICHT_GENUEGEND_ARBEITSSPEICHER](#) = 610407008 }

Ausführliche Beschreibung

Auflistung der Otto-Statuscodes.

Dokumentation der Aufzählungstypen

enum [OttoStatusCode](#)

Aufzählungswerte:

OTTO_OK	Die Verarbeitung ist ordnungsgemäß abgeschlossen worden.
OTTO_INTERNER_FEHLER	Es trat ein interner Fehler auf, Details stehen im otto.log.
OTTO_UNBEKANNTER_FEHLER	Es trat ein unerwarteter Fehler auf, Details stehen im otto.log.

OTTO_NPA_ZERTIFIKATFEHLER	Es trat ein Fehler bei der Erzeugung eines Ad-hoc-Zertifikats für den nPA auf, Details stehen ggf. im otto.log.
OTTO_TRANSFER_FEHLER	Es trat ein Fehler beim Transfer auf, Details stehen ggf. im otto.log.
OTTO_TRANSFER_INIT	Es trat ein Fehler bei der Initialisierung des Transfers auf.
OTTO_TRANSFER_CONNECTIONSERVER	Es konnte keine Verbindung zum OTTER-Server aufgebaut werden.
OTTO_TRANSFER_CONNECTIONPROXY	Es konnte keine Verbindung zum Proxy aufgebaut werden.
OTTO_TRANSFER_TIMEOUT	Bei der Kommunikation mit dem Server kam es zu einer Zeitüberschreitung.
OTTO_TRANSFER_PROXYAUTH	Der Proxy erwartet Anmeldedaten oder der Proxy hat die Verbindung abgelehnt.
OTTO_TRANSFER_UNAUTHORIZED	Der Client darf die Schnittstelle nicht verwenden.
OTTO_TRANSFER_NOT_FOUND	Der OTTER-Server hat das Objekt nicht gefunden.
OTTO_TRANSFER_SERVER_FEHLER	Der OTTER-Server hat einen unerwarteten Fehler gemeldet. Möglicherweise ist ein Retry sinnvoll. Details stehen im otto.log.
OTTO_TRANSFER_DECODING	Die empfangenen Daten konnten nicht dekodiert werden.
OTTO_TRANSFER_EID_ZERTIFIKATFEHLER	Es konnte kein Ad-hoc-Zertifikat für den Personalausweis oder den Aufenthaltstitel erzeugt bzw. gefunden werden, Details stehen ggf. im otto.log.

OTTO_TRANS FER_EID_KEI NCLIENT	Der eID-Client ist nicht erreichbar. Wahrscheinlich wurde er nicht gestartet oder die übergebene lokale URL ist nicht korrekt.
OTTO_TRANS FER_EID_KEI NKONTO	Für die Identifikationsnummer des Benutzers existiert kein Konto bei ELSTER.
OTTO_TRANS FER_EID_CLIE NTFEHLER	Der eID-Client hat einen Fehler gemeldet. Details zu dem Fehler finden Sie im Log des eID-Clients oder ggf. im otto.log.
OTTO_TRANS FER_EID_NPA BLOCKIERT	Der Personalausweis wird von einem anderen Vorgang blockiert. Beenden Sie den anderen Vorgang und versuchen Sie es dann erneut.
OTTO_UNGUE LTIGER_PARA METER	Einer der übergebenen Parameter ist ungültig.
OTTO_UNGUE LTIGES_HAND LE	Das übergebene Handle ist ungültig.
OTTO_MEHRF ACHAUFRUFE _NICHT_UNTE RSTUETZT	Die übergebene Otto-Instanz wird gerade (zum Beispiel in einem anderen Thread) verwendet.
OTTO_INSTAN ZEN_INKONSI STENT	Eines der übergebenen Otto-Objekte wurde mit einer anderen Otto-Instanz erstellt.
OTTO_INSTAN Z_UNTEROBJ EKTE_NICHT_ FREIGEGERBE N	Mit dieser Instanz wurden Unterobjekte erzeugt, die noch nicht freigegeben worden sind.
OTTO_LOG_F EHLER	Die Protokolldatei konnte nicht erzeugt oder geöffnet werden.
OTTO_FUNKTI ON_NICHT_U NTERSTUETZ T	Die verwendete Funktion oder Funktionalität wird nicht, noch nicht oder nicht mehr unterstützt.

OTTO_ZERTIFIKAT_PIN_FALSCH	Für das Zertifikat wurde ein falsches Passwort bzw. eine falsche PIN angegeben.
OTTO_ZERTIFIKAT_PFAD_FALSCH	Unter dem angegebenen Pfad wurde kein Zertifikat gefunden.
OTTO_ZERTIFIKAT_NICHT_ERKANNT	Das Zertifikat wurde nicht erkannt, Details stehen ggf. im otto.log.
OTTO_PRUEFSUMME_FINALISIERT	Die Prüfsumme wurde bereits finalisiert
OTTO_UNGÜLTIGE_HERSTELLERID	Es wurde keine oder eine ungültige Hersteller-ID angegeben.
OTTO_EMPFANG_VORZEITIG_BEENDET	Der Empfang wurde durch einen API-Aufruf vorzeitig beendet.
OTTO_VERSAND_GERINGE_DATENMENGE	Die Versanddaten dürfen die Mindestgröße nicht unterschreiten.
OTTO_ESIGNER_NICHT_GELADEN	Die eSigner-Bibliothek konnte nicht geladen werden, Details stehen ggf. im otto.log
OTTO_ESIGNER_VERALTET	Die eSigner-Bibliothek ist veraltet, Details stehen ggf. im otto.log.
OTTO_ESIGNER_INKOMPATIBEL	Die eSigner-Bibliothek ist mit der Otto-Bibliothek inkompatibel.
OTTO_PROXY_URL	Es wurde keine URL oder IP für den Proxy angegeben.
OTTO_PROXY_PORT	Es wurde kein oder ein ungültiger Port für den Proxy angegeben.
OTTO_PROXY_AUTHSCHEMA	Es wurde kein gültiges Proxy-Authentifizierungschema angegeben.

OTTO_VERSAND_ABGESCHLOSSEN	Der Versand wurde bereits abgeschlossen.
OTTO_VERSAND_ZUGROSSE_DATENMENGE	Die Versanddaten dürfen die Maximalgröße nicht überschreiten.
OTTO_EINSTELLUNG_UNBEKANNT	Die Einstellung ist unbekannt
OTTO_EINSTELLUNG_WERT_UNGUELTIG	Der Wert ist für diese Einstellung nicht zulässig
OTTO_ESIGNER_BUSY	eSigner: Überlastung
OTTO_ESIGNER_DECRYPT	eSigner: Fehler beim Entschlüsseln
OTTO_ESIGNER_ENCRYPT	eSigner: Fehler beim Verschlüsseln
OTTO_ESIGNER_ENCODE_ERROR	eSigner: Fehler beim Encoding
OTTO_ESIGNER_ENCODE_UNKNOWN	eSigner: Parameter Fehler: unbekanntes Encoding
OTTO_ESIGNER_ESICL_EXCEPTION	eSigner: Eine Laufzeitausnahme ist aufgetreten und abgefangen worden.
OTTO_ESIGNER_INVALID_HANDLE	eSigner: Ungültiges Token-Handle
OTTO_ESIGNER_LOAD_DLL	eSigner: PKCS11- bzw. PC/SC-Bibliothek fehlt oder ist nicht ausführbar.
OTTO_ESIGNER_MAX_SESSION	eSigner: Zu viele Sessions geöffnet.

OTTO_ESIGN ER_NO_SERVICE	eSigner: Der PC/SC-Dienst ist nicht gestartet.
OTTO_ESIGN ER_NO_SIGNATURE_KEY	eSigner: Kein Signatur/Verschlüsselungszertifikat bzw. -schlüssel vorhanden.
OTTO_ESIGN ER_OUT_OF_MEM	eSigner: Speicherallokation fehlgeschlagen.
OTTO_ESIGN ER_P11_ENCKEY	eSigner: Fehler beim Zugriff auf Hard-Token-Entschlüsselungsschlüssel
OTTO_ESIGN ER_P11_ENGINE_LOADED	eSigner: Die PKCS#11-Engine wird von einer anderen Bibliothek belegt.
OTTO_ESIGN ER_P11_INIT_FAILED	eSigner: P11 Initialer Token-Zugriff fehlgeschlagen.
OTTO_ESIGN ER_P11_NO_ENCKERT	eSigner: P11 Verschlüsselungszertifikat fehlt.
OTTO_ESIGN ER_P11_NO_SIGNATURE_CERT	eSigner: P11 Signaturzertifikat fehlt.
OTTO_ESIGN ER_P11_SIGNATURE_KEY	eSigner: Fehler beim Zugriff auf den Hard-Token-Signaturschlüssel
OTTO_ESIGN ER_P11_SLOT_EMPTY	eSigner: Leere Slot-Liste, d.h. keine Karte eingesteckt.
OTTO_ESIGN ER_P12_CREATE	eSigner: Temporäres PKCS#12-Token konnte nicht erzeugt werden.
OTTO_ESIGN ER_P12_DECODE	eSigner: Fehler beim Dekodieren des PKCS#12-Objekts

OTTO_ESIGN ER_P12_ENC_ KEY	eSigner: Fehler beim Zugriff auf den Soft-PSE-Entschlüsselungsschlüssel
OTTO_ESIGN ER_P12_SIG_ KEY	eSigner: Fehler beim Zugriff auf den Soft-PSE-Signaturschlüssel
OTTO_ESIGN ER_P12_NO_E NC_CERT	eSigner: P12 Verschlüsselungszertifikat fehlt.
OTTO_ESIGN ER_P12_NO_S IG_CERT	eSigner: P12 Signaturzertifikat fehlt.
OTTO_ESIGN ER_P12_REA D	eSigner: Fehler beim Lesen des PKCS#12-Objekts
OTTO_ESIGN ER_P7_DECO DE	eSigner: Fehler beim Dekodieren des PKCS#7-Objekts
OTTO_ESIGN ER_P7_READ	eSigner: Fehler beim Lesen des PKCS#7-Objekts
OTTO_ESIGN ER_P7_RECIP IENT	eSigner: Das Entschlüsselungszertifikat ist nicht in der Empfängerliste enthalten.
OTTO_ESIGN ER_PIN_LOCK ED	eSigner: Die PIN bzw. das Passwort ist gesperrt.
OTTO_ESIGN ER_PIN_WRO NG	eSigner: Die PIN bzw. das Passwort ist falsch.
OTTO_ESIGN ER_PSE_PAT H	eSigner: Der Pfad zum Soft-PSE ist falsch oder ungültig.
OTTO_ESIGN ER_SC_ENC_ KEY	eSigner: Fehler beim Zugriff auf den Stick-Entschlüsselungsschlüssel.
OTTO_ESIGN ER_SC_INIT_F AILED	eSigner: Initialer Token-Zugriff auf die PC/SC-Schnittstelle fehlgeschlagen.

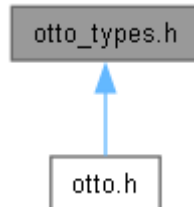
OTTO_ESIGN ER_SC_NO_A PPLET	eSigner: Kein unterstütztes Applet gefunden.
OTTO_ESIGN ER_SC_NO_E NC_CERT	eSigner: PC/SC Verschlüsselungszertifikat fehlt.
OTTO_ESIGN ER_SC_NO_SI G_CERT	eSigner: PC/SC Signaturzertifikat fehlt.
OTTO_ESIGN ER_SC_SESSI ON	eSigner: Fehler in der Karten-Session.
OTTO_ESIGN ER_SC_SIG_K EY	eSigner: Fehler beim Zugriff auf den Stick-Signaturschlüssel
OTTO_ESIGN ER_SC_SLOT _EMPTY	eSigner: Es ist keine Karte bzw. kein Stick eingesteckt.
OTTO_ESIGN ER_TOKEN_T YPE_MISMAT CH	eSigner: Der Token-Typ der CA (Certification Authority, Zertifizierungsstelle) stimmt nicht mit dem internen Token-Typ überein.
OTTO_ESIGN ER_USER_CA NCEL	eSigner: Die Aktion wurde vom Benutzer abgebrochen.
OTTO_ESIGN ER_VERIFY_C ERT_CHAIN	eSigner: Die Zertifikatskette konnte nicht verifiziert werden.
OTTO_ESIGN ER_DATA_NO T_INITIALIZED	eSigner: Die Datenstruktur ist nicht initialisiert.
OTTO_ESIGN ER_ASN1_RE AD_BUFFER_ TOO_SMALL	eSigner: Der Lesebuffer zum Dekodieren der ASN.1-Struktur ist zu klein.
OTTO_ESIGN ER_ASN1_RE AD_DATA_INC OMplete	eSigner: Die Daten der ASN.1-Struktur sind unvollständig.

OTTO_ESIGNER_ASN1_NO_ENVELOPED_DATA	eSigner: Die ASN.1-Struktur enthält kein Enveloped Data.
OTTO_ESIGNER_ASN1_NO_CONTENT_DATA	eSigner: Die ASN.1-Struktur enthält keine Daten.
OTTO_INIDATEI_LESEFEHLER	Fehler beim Einlesen der otto.ini.
OTTO_ZERTIFIKAT_LESEFEHLER	Das Zertifikat konnte nicht geladen werden.
OTTO_ZERTIFIKAT_DEFEKT	Die Zertifikatsdatei oder das Soft-PSE ist defekt.
OTTO_ZERTIFIKAT_FINGERABDRUCK_FEHLER	Für das Zertifikat konnte kein Fingerabdruck erstellt werden.
OTTO_SIGNIEREN_FEHLGESCHLAGEN	Die Daten konnten mit dem übergebenen Zertifikat nicht signiert werden.
OTTO_ENTSCHLUESSELN_FEHLGESCHLAGEN	Die Daten konnten mit dem übergebenen Zertifikat nicht entschlüsselt werden.
OTTO_DEKOMPRESSION_FEHLGESCHLAGEN	Die Daten konnten nicht dekomprimiert werden.
OTTO_NICHTGENUEGENDARBEITSSPEICHER	Es ist nicht genügend Arbeitsspeicher vorhanden.

otto_types.h-Dateireferenz

Definition von Datenstrukturen und Datentypen.

Dieser Graph zeigt, welche Datei direkt oder indirekt diese Datei enthält:



Datenstrukturen

struct [OttoProxyKonfiguration](#)

Diese Struktur enthält alle Informationen, die Otto benötigt, um die Verbindung zum OTTER-Server oder dem ELSTER-eID-Server über einen Proxy aufzubauen.

Typdefinitionen

- typedef struct OttoInstanz * [OttoInstanzHandle](#)
Handle auf eine Otto-Instanz.
- typedef struct OttoZertifikat * [OttoZertifikatHandle](#)
Handle auf ein Sicherheitstoken zur Authentifizierung des Daten-Übermittlers oder -Abholers.
- typedef struct OttoRueckgabepuffer * [OttoRueckgabepufferHandle](#)
Handle auf einen Otto-Rückgabepuffer.
- typedef struct OttoPruefsumme * [OttoPruefsummeHandle](#)
Handle auf eine Otto-Prüfsumme.
- typedef struct OttoVersand * [OttoVersandHandle](#)
Handle auf ein Otto-Versandobjekt.

- typedef struct OttoEmpfang * [OttoEmpfangHandle](#)
Handle auf ein Otto-Empfangsobjekt.
- typedef int(* [OttoLogCallback](#)) (const char *instanzId, const char *logZeitpunkt, [OttoLogEbene](#) logEbene, const char *logNachricht, void *benutzerdaten)
Funktionstyp für einen Log-Callback, den eine Anwendung beim Erzeugen einer Otto-Instanz angeben kann.

Aufzählungen

- enum [OttoLogEbene](#) { [OTTOLOG_FEHLERMELDUNGEN](#) = 4, [OTTOLOG_WARNUNGEN](#) = 3, [OTTOLOG_INFORMATIONEN](#) = 2, [OTTOLOG_DEBUGMELDUNGEN](#) = 1 }
- Aufzählung der Log-Ebenen von Otto.*

Ausführliche Beschreibung

Definition von Datenstrukturen und Datentypen.

Dokumentation der benutzerdefinierten Typen

typedef struct OttoEmpfang* [OttoEmpfangHandle](#)

Handle auf ein Otto-Empfangsobjekt.

Über ein Empfangsobjekt können Daten blockweise vom OTTER-Server heruntergeladen werden.

typedef struct OttoInstanz* [OttoInstanzHandle](#)

Handle auf eine Otto-Instanz.

Jede Funktion der Otto-API ist direkt oder indirekt an eine Otto-Instanz gebunden. Die Otto-Instanz enthält sämtliche veränderlichen Zustände von Otto.

Es können mehrere Instanzen gleichzeitig existieren. Jede der Instanzen ist unabhängig von allen anderen. Verfügen mehrere Threads über jeweils ihre eigene Otto-Instanz, können sie diese Instanzen parallel verwenden.

Eine Otto-Instanz kann zwischen Threads ausgetauscht werden. Sie darf aber nicht in zwei Threads gleichzeitig verwendet werden.

Siehe auch:

- [OttoInstanzErzeugen\(\)](#)
- [OttoInstanzFreigeben\(\)](#)

typedef int(* OttoLogCallback) (const char *instanzld, const char *logZeitpunkt, [OttoLogEbene](#) logEbene, const char *logNachricht, void *benutzerdaten)

Funktionstyp für einen Log-Callback, den eine Anwendung beim Erzeugen einer Otto-Instanz angeben kann.

Rückgabe:

- 0 wenn die Log-Meldung erfolgreich entgegengenommen werden konnte, ungleich 0 im Fehlerfall. Momentan wird der Rückgabewert vom Otto jedoch ignoriert.

typedef struct OttoPruefsumme* [OttoPruefsummeHandle](#)

Handle auf eine Otto-Prüfsumme.

Um die Datenintegrität sicherzustellen, ist beim Versand von Daten an den OTTER-Server eine Prüfsumme über die Daten zu übermitteln.

Diese Prüfsumme ist schon beim Verbindungsaufbau an die Server zu übermitteln und kann daher nicht während der Datenübertragung implizit gebildet werden, sondern muss vorab explizit berechnet werden.

typedef struct OttoRueckgabepuffer* [OttoRueckgabepufferHandle](#)

Handle auf einen Otto-Rückgabepuffer.

Ein Otto-Rückgabepuffer dient zur Übergabe von Daten vom Otto an die Anwendung. Die Anwendung erstellt dazu einen Otto-Rückgabepuffer und übergibt dessen Handle als Parameter einer API-Funktion an den Otto. Der Otto befüllt den Otto-Rückgabepuffer dann mit Daten.

Zu beachten:

Eventuell bereits im Otto-Rückgabepuffer befindliche Daten werden beim Aufruf von API-Funktionen gelöscht. Ausgenommen davon sind natürlich die API-Funktionen mit dem Präfix "OttoRueckgabepuffer", die einen Otto-Rückgabepuffer selbst behandeln.

Siehe auch:

- [OttoRueckgabepufferErzeugen\(\)](#)
- [OttoRueckgabepufferFreigeben\(\)](#)
- [OttoRueckgabepufferInhalt\(\)](#)
- [OttoRueckgabepufferGroesse\(\)](#)

typedef struct OttoVersand* [OttoVersandHandle](#)

Handle auf ein Otto-Versandobjekt.

Über ein Versandobjekt können Daten blockweise an die OTTER-Server übermittelt werden.

typedef struct OttoZertifikat* [OttoZertifikatHandle](#)

Handle auf ein Sicherheitstoken zur Authentifizierung des Daten-Übermittlers oder -Abholers.

Dokumentation der Aufzählungstypen**enum [OttoLogEbene](#)**

Aufzählung der Log-Ebenen von Otto.

Die Log-Meldungen von Otto sind in Ebenen angeordnet: von der höchsten Ebene mit den wichtigsten Fehlermeldungen bis hin zu niedrigsten Ebene mit einfachen Meldungen, die nur bei der Suche nach Fehlerursachen interessant sind.

Standardmäßig werden nur Meldungen der beiden höchsten Ebenen (Fehler und Warnungen) protokolliert.

Sollen auch Meldungen der niedrigeren Ebenen protokolliert werden, so kann über die Funktion [OttoInstanzErzeugen\(\)](#) ein Callback vom Typ [OttoLogCallback](#) registriert werden, in dem die Meldungen nach der Ebene gefiltert werden können.

Siehe auch:

- [OttoInstanzErzeugen\(\)](#)

Aufzählungswerte:

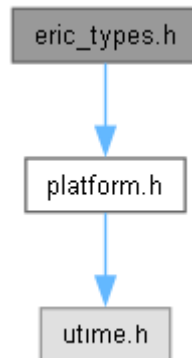
OTTOLOG_FEHLMER MELDUNGEN	Fehler, die zum Abbruch der gewünschten Aktion führen.
OTTOLOG_WARNUNGEN	Hinweise auf Zustände, die zu Fehlern führen können.
OTTOLOG_INFORMATIONEN	Grobe Informationen über den Programmablauf und Werte.
OTTOLOG_DEBUGMELDUNGEN	Feingranulare Informationen über den Programmablauf und Werte.

eric_types.h-Dateireferenz

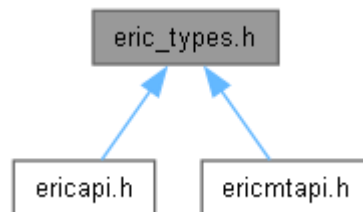
Definition von Datenstrukturen und Datentypen.

```
#include "platform.h"
```

Include-Abhängigkeitsdiagramm für eric_types.h:



Dieser Graph zeigt, welche Datei direkt oder indirekt diese Datei enthält:



Datenstrukturen

struct [eric_druck_parameter_t](#)

Diese Struktur enthält alle für den Druck notwendigen Informationen.

struct [eric_verschluesselungs_parameter_t](#)

Für die Signatur oder Authentifizierung benötigte Informationen.

struct [eric_zertifikat_parameter_t](#)

Struktur mit Informationen zur Erzeugung von Zertifikaten mit [EricCreateKey\(\)](#).

Typdefinitionen

- typedef struct EricInstanz * [EricInstanzHandle](#)

Handle auf eine ERiC-Instanz.

- typedef char [byteChar](#)
*Der Datentyp [byteChar](#) wird immer dann verwendet, wenn an diesem Parameter keine UTF-8 codierte Daten erwartet werden. Diese Daten werden ungeprüft verwendet. **Zum Beispiel:** Pfade.*
- typedef [uint32_t](#) [EricZertifikatHandle](#)
Integer-Typ für Zertifikat-Handles.
- typedef [uint32_t](#) [EricTransferHandle](#)
Das [EricTransferHandle](#) wird beim Anwendungsfall "Datenabholung" der API-Funktion [EricBearbeiteVorgang\(\)](#) übergeben.
- typedef struct EricReturnBufferApi * [EricRueckgabepufferHandle](#)
Handle zur Verwaltung und Verwendung von Rückgabepuffern.
- typedef int(* [EricPdfCallback](#)) (const char *pdfBezeichner, const BYTE *pdfDaten, [uint32_t](#) pdfGroesse, void *benutzerDaten)
Typ der Callback-Funktion zur Übergabe eines PDFs an die Anwendung.
- typedef void(* [EricLogCallback](#)) (const char *kategorie, [eric_log_level_t](#) loglevel, const char *nachricht, void *benutzerdaten)
Typ der Callback-Funktion zum Logging.
- typedef void(* [EricFortschrittCallback](#)) ([uint32_t](#) id, [uint32_t](#) pos, [uint32_t](#) max, void *benutzerdaten)
Typ der Callback-Funktionen, die am ERiC für Fortschrittanzeigen registriert werden können.

Aufzählungen

- enum [eric_bearbeitung_flag_t](#) { [ERIC_VALIDIERE](#) = 1L << 1, [ERIC_SENDE](#) = 1L << 2, [ERIC_DRUCKE](#) = 1L << 5, [ERIC_PRUEFE_HINWEISE](#) = 1L << 7, [ERIC_VALIDIERE_OHNE_FREIGABEDATUM](#) = 1L << 8 }
 - enum [eric_log_level_t](#) { [ERIC_LOG_ERROR](#) = 4, [ERIC_LOG_WARN](#) = 3, [ERIC_LOG_INFO](#) = 2, [ERIC_LOG_DEBUG](#) = 1, [ERIC_LOG_TRACE](#) = 0 }
- Bearbeitungsflags für die Anwendungsfälle von [EricBearbeiteVorgang\(\)](#).*

- enum { [ERIC_FORTSCHRITTCALLBACK_ID_EINLESEN](#) = 10,
[ERIC_FORTSCHRITTCALLBACK_ID_VORBEREITEN](#) = 20,
[ERIC_FORTSCHRITTCALLBACK_ID_VALIDIEREN](#) = 30,
[ERIC_FORTSCHRITTCALLBACK_ID_SENDEN](#) = 40,
[ERIC_FORTSCHRITTCALLBACK_ID_DRUCKEN](#) = 50 }

Ausführliche Beschreibung

Definition von Datenstrukturen und Datentypen.

Dokumentation der benutzerdefinierten Typen

typedef char [byteChar](#)

Der Datentyp [byteChar](#) wird immer dann verwendet, wenn an diesem Parameter keine UTF-8 codierte Daten erwartet werden. Diese Daten werden ungeprüft verwendet. **Zum Beispiel:** Pfade.

Der Datentyp [byteChar](#) wird immer dann verwendet, wenn an diesem Parameter keine UTF-8 codierte Daten erwartet werden. Diese Daten werden ungeprüft verwendet.

typedef void(* EricFortschrittCallback) ([uint32_t](#) id, [uint32_t](#) pos, [uint32_t](#) max, void *benutzerdaten)

Typ der Callback-Funktionen, die am ERiC für Fortschrittanzeigen registriert werden können.

Parameter:

<i>id</i>	Aktueller Verarbeitungsschritt
<i>pos</i>	Aktueller Fortschritt bezogen auf <i>max</i>
<i>max</i>	Maximalwert des aktuellen Fortschritts <i>pos</i>
<i>benutzerdaten</i>	Der Zeiger, der bei der Registrierung mit EricRegistriereGlobalenFortschrittCallback() oder EricRegistriereFortschrittCallback() übergeben worden ist, wird in diesem Parameter vom ERiC unverändert übergeben.

Es gilt stets, dass:

- `pos` größer oder gleich 0 und kleiner oder gleich `max` ist
- `max` ist immer größer als 0

typedef struct EricInstanz* [EricInstanzHandle](#)

Handle auf eine ERiC-Instanz.

ERiC-Instanzen werden von der Multithreading-API angelegt, verwendet und wieder freigegeben, siehe [ericmtapi.h](#).

Alle API-Funktionen der Multithreading-API nehmen einen Zeiger auf eine ERiC-Instanz entgegen und verrichten ihre Aufgaben auf dieser ERiC-Instanz. Die EricInstanz enthält sämtliche veränderlichen Zustände des ERiC. Dies sind ERiC-Einstellungen, Plugin- und Log-Verzeichnis, Proxyeinstellungen, Zertifikatshandle, Rückgabepuffer, etc.

Es können mehrere ERiC-Instanzen parallel angelegt werden. Jede dieser ERiC-Instanzen ist unabhängig von allen anderen ERiC-Instanzen. Verfügen mehrere Threads jeweils über ihre eigene ERiC-Instanz, können sie diese parallel verwenden. Dazu müssen die Threads den API-Funktionen der Multithreading-API ihre jeweils eigene ERiC-Instanz übergeben.

ERiC-Instanzen sollten nicht für jede Verarbeitung eines Steuerfalls neu erstellt und konfiguriert werden. Siehe hierzu [ERiC-Entwicklerhandbuch.pdf](#), Kap. "Hinweise zum optimierten Einsatz von ERiC-Instanzen und Plugins"

ERiC-Instanzen können zwischen Threads ausgetauscht werden. Eine ERiC-Instanz darf aber nicht in zwei Threads gleichzeitig verwendet werden.

Siehe auch:

- [ERiC-Entwicklerhandbuch.pdf](#), Kap. "Hinweise zum optimierten Einsatz von ERiC-Instanzen und Plugins"
- [EricMtlInstanzErzeugen\(\)](#)
- [EricMtlInstanzFreigegeben\(\)](#)

typedef void(* EricLogCallback) (const char *kategorie, [eric_log_level_t](#) loglevel, const char *nachricht, void *benutzerdaten)

Typ der Callback-Funktion zum Logging.

Wenn registriert, wird diese Callback-Funktion für jeden Log-Eintrag mit folgenden Parametern aufgerufen.

Parameter:

<i>kategorie</i>	Kategorie des Logeintrags. Beinhaltet das Modul, welches den Log-Eintrag ausgibt. Zum Beispiel "eric.ctrl2". Kann zum Filtern verwendet werden. Alle Log-Nachrichten besitzen eine Kategorie. Der Zeiger ist nur innerhalb dieser Funktion gültig.
<i>loglevel</i>	Log-Level des Logeintrags. Kann zum Filtern verwendet werden.
<i>nachricht</i>	Enthält die Log-Nachricht als Zeichenkette. Der Zeiger ist nur innerhalb dieser Funktion gültig.
<i>benutzerdaten</i>	Der Zeiger, der bei der Registrierung mit EricRegistriereLogCallback() übergeben worden ist, wird in diesem Parameter vom ERiC unverändert übergeben.

Siehe auch:

- [EricRegistriereLogCallback\(\)](#)

typedef int(* EricPdfCallback) (const char *pdfBezeichner, const BYTE *pdfDaten, [uint32 t](#) pdfGroesse, void *benutzerDaten)

Typ der Callback-Funktion zur Übergabe eines PDFs an die Anwendung.

Wenn diese Callback-Funktion im [eric druck parameter t](#) angegeben wird werden PDFs vom ERiC nicht in eine Datei geschrieben, sondern an diese Callback-Funktion übergeben.

Parameter:

in	<i>pdfBezeichner</i>	Bezeichner für das PDF. Für ein PDF, das Inhalte aus einem Nutzdatenblock enthält, wird hier das Nutzdatenticket aus dem Nutzdatenblock übergeben, für sonstige PDFs das Wort "Uebertragungsprotokoll". Bei der Erstellung mehrerer PDFs ermöglicht das Nutzdatenticket die Zuordnung eines PDFs zu einem bestimmten Nutzdatenblock.
in	<i>pdfDaten</i>	Der Inhalt des PDFs. Zu beachten: es handelt sich um binäre Daten, die Nullbytes enthalten können.
in	<i>pdfGroesse</i>	Die Größe der <i>pdfDaten</i> in Bytes.
in	<i>benutzerDaten</i>	Der Datenzeiger, der dem ERiC von der Anwendung im eric druck parameter t übergeben wurde.

Rückgabe:

- 0, wenn kein Fehler aufgetreten ist. Ein beliebiger Wert ungleich 0, wenn ein Fehler aufgetreten ist. Der zurückgegebene Wert wird im Fehlerfall in die Datei eric.log protokolliert.

Siehe auch:

- [eric_druck_parameter_t](#)
- [EricBearbeiteVorgang\(\)](#)

typedef struct EricReturnBufferApi* [EricRueckgabepufferHandle](#)

Handle zur Verwaltung und Verwendung von Rückgabepuffern.

Viele ERiC API-Funktionen geben Informationen an ihren Aufrufer zurück, indem sie Daten in sogenannte Rückgabepuffer schreiben. Solche Rückgabepuffer müssen mit [EricRueckgabepufferErzeugen\(\)](#) angelegt werden. Das bei dieser Erzeugung generierte Puffer-Handle wird vom Aufrufer an die API-Funktion übergeben, die den Puffer leert bevor sie dann in den Puffer schreibt. Ein einmal generiertes Puffer-Handle kann damit auch für mehrere aufeinanderfolgende Aufrufe von ERiC API-Funktionen wiederverwendet werden. Mittels [EricRueckgabepufferLaenge\(\)](#) kann danach die Anzahl der in den Puffer geschriebenen Bytes ermittelt werden. Mit [EricRueckgabepufferInhalt\(\)](#) kann der Pufferinhalt abgefragt werden. Jeder Rückgabepuffer muss nach seiner Verwendung mit [EricRueckgabepufferFreigeben\(\)](#) wieder freigegeben werden.

Die Struktur EricReturnBufferApi kapselt die Rückgabepuffer-Implementierung. Anwender der ERiC API verwenden ausschließlich Zeiger auf Instanzen dieser Struktur und müssen daher deren Felder nicht kennen.

Rückgabepuffer sind der Singlethreading-API bzw. einer ERiC-Instanz der Multithreading-API fest zugeordnet. Die Funktionen der ERiC API, die einen Rückgabepuffer entgegennehmen, geben den Fehlercode [ERIC_GLOBAL_PUFFER_UNGLEICHER_INSTANZ](#) zurück, wenn der übergebene Rückgabepuffer:

- mit der Singlethreading-API erzeugt worden ist und dann mit der Multithreading-API verwendet wird
- mit der Multithreading-API erzeugt worden ist und dann mit der Singlethreading-API verwendet wird
- mit einer ERiC-Instanz erzeugt worden ist und dann mit einer anderen Instanz verwendet wird.

Siehe auch:

- [ERiC-Entwicklerhandbuch.pdf](#), Kap. "Rückgabepuffer der ERiC Programmierschnittstelle"
- [EricRueckgabepufferErzeugen\(\)](#)
- [EricRueckgabepufferLaenge\(\)](#)
- [EricRueckgabepufferInhalt\(\)](#)
- [EricRueckgabepufferFreigeben\(\)](#)

typedef [uint32_t](#) [EricTransferHandle](#)

Das [EricTransferHandle](#) wird beim Anwendungsfall "Datenabholung" der API-Funktion [EricBearbeiteVorgang\(\)](#) übergeben.

Es ist vom Aufrufer zu initialisieren und wird [EricBearbeiteVorgang\(\)](#) als Zeiger übergeben. Es wird verwendet, um bei der Datenabholung mehrere Versandvorgänge zu bündeln. Dabei ist das Handle für den ersten Vorgang "Anfrage" mit dem Wert 0 zu initialisieren, bevor [EricBearbeiteVorgang\(\)](#) aufgerufen wird. Das von [EricBearbeiteVorgang\(\)](#) zurückgegebene Handle ist dann bei allen Folgevorgängen derselben Datenabholung unverändert wieder zu übergeben.

Wird bei einer Datenabholung NULL oder ein ungültiger Zeiger als Handle übergeben, gibt [EricBearbeiteVorgang\(\)](#) den Fehlercode [ERIC_GLOBAL_TRANSFERHANDLE](#) zurück.

Bei allen Verfahren außer der Datenabholung sollte das Transferhandle beim Aufruf der [EricBearbeiteVorgang\(\)](#) NULL sein. Wird bei solchen Verfahren ein Handle übergeben, so wird dieses ignoriert.

typedef [uint32_t](#) [EricZertifikatHandle](#)

Integer-Typ für Zertifikat-Handles.

Dokumentation der Aufzählungstypen

anonymous enum

Aufzählungswerte:

ERIC_FORTS CHRITTCALLB ACK_ID_EINL ESEN	<code>id</code> , die beim Einlesen des XMLs von Fortschrittcallbacks ausgegeben wird.
ERIC_FORTS CHRITTCALLB ACK_ID_VORB EREITEN	<code>id</code> , die gemeldet wird, wenn die Daten zum Versand noch vorbereitet werden müssen.
ERIC_FORTS CHRITTCALLB ACK_ID_VALI DIEREN	<code>id</code> , die beim Validieren der Eingangsdaten von Fortschrittcallbacks ausgegeben wird.

ERIC_FORTS CHRIITCALLB ACK_ID_SEN D	<code>id</code> , die beim Versand der Ausgangsdaten von Fortschrittcallbacks ausgegeben wird.
ERIC_FORTS CHRIITCALLB ACK_ID_DRU CKEN	<code>id</code> , die beim Druck der Eingangsdaten von Fortschrittcallbacks ausgegeben wird.

enum [eric_bearbeitung_flag_t](#)

Bearbeitungsflags für die Anwendungsfälle von [EricBearbeiteVorgang\(\)](#).

Welche Anwendungsfälle von der jeweiligen Datenart unterstützt werden, ist dem [ERiC-Entwicklerhandbuch.pdf](#) zu entnehmen.

Aufzählungswerte:

ERIC_VALIDI ERE	Der Datensatz soll validiert werden.
ERIC_SENDE	Der Datensatz soll an den ELSTER Annahmeserver versendet werden.
ERIC_DRUCK E	Der Datensatz soll gedruckt werden.
ERIC_PRUEFE _HINWEISE	Der Datensatz soll auf Hinweise hin geprüft werden.
ERIC_VALIDI ERE_OHNE_FR EIGABEDATU M	<p>Der Datensatz soll validiert werden, ohne dabei die Prüfbedingung "ERiC_DV_Freigabedatum_pruefen" auszuführen. Dies ist nur möglich, wenn kein Versand stattfindet.</p> <p>Das Flag kann nicht zusammen mit ERIC_SENDE oder ERIC_VALIDIERE angegeben werden.</p> <p>Zu beachten:</p> <p>Zur Prüfbedingung "ERiC_DV_Freigabedatum_pruefen" siehe das Dokument "Zusatzinformationen_zur_Plausibilitaetspruefung.pdf" im ERiC-Dokumentationspaket.</p>

enum [eric_log_level_t](#)

[eric_log_level_t](#) ist ein Parameter für Funktionen vom Typ [EricLogCallback](#). Der Loglevel kann zum Filtern für das ERiC Protokoll verwendet werden, siehe [ERiC-Entwicklerhandbuch.pdf](#), Kap. "Das ERiC Protokoll eric.log".

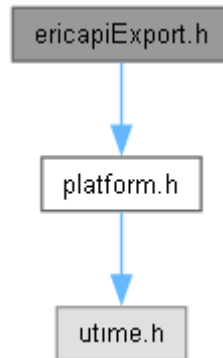
Aufzählungswerte:

ERIC_LOG_ERROR	Fehler, der zum Programmabbruch führt.
ERIC_LOG_WARN	Hinweise auf Zustände, die zu Fehlern führen können.
ERIC_LOG_INFO	Grobe Informationen über den Programmablauf und Werte.
ERIC_LOG_DEBUG	Feingranulare Informationen über den Programmablauf und Werte.
ERIC_LOG_TRACE	Sehr feingranulare Informationen über den Programmablauf und Werte.

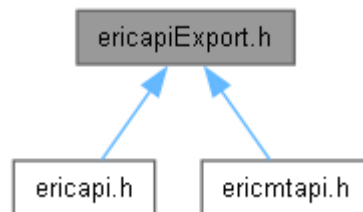
ericapiExport.h-Dateireferenz

```
#include "platform.h"
```

Include-Abhängigkeitsdiagramm für ericapiExport.h:



Dieser Graph zeigt, welche Datei direkt oder indirekt diese Datei enthält:



Makrodefinitionen

- #define [ERICAPI_IMPORT](#)

Ausführliche Beschreibung

Attribute für dynamische Bibliotheken

Diese Deklarationen sind für Windows-Plattformen relevant.

Makro-Dokumentation

#define ERICAPI_IMPORT

Dieses Makro ist nur für die Kompilierung unter Windows von Belang. Auf allen anderen Plattformen ist es leer definiert.

Unter Windows gibt es bei Einbindung der [ericapiExport.h](#) in fremden Code an, dass die deklarierten Funktionen zu importieren sind (dllimport).

Siehe auch:

- [ETKAPI_DECL](#)

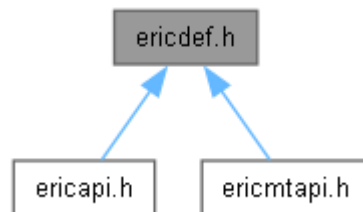
ericdef.h-Dateireferenz

```
#include "platform.h"
```

Include-Abhängigkeitsdiagramm für ericdef.h:



Dieser Graph zeigt, welche Datei direkt oder indirekt diese Datei enthält:



Makrodefinitionen

- #define [ERIC_MAX_LAENGE_FUSSTEXT](#) (30)
- #define [ERIC_TESTMERKER_CLEARINGSTELLE](#) "700000004"
- #define [ERIC_TESTMERKER_ECC](#) "700000001"

Ausführliche Beschreibung

Konstanten und Definitionen für Übergabeparameter

Makro-Dokumentation

#define ERIC_MAX_LAENGE_FUSSTEXT (30)

Definition der maximalen Länge des Fusstextes in [eric_druck_parameter t](#) + Nullterminierer

#define ERIC_TESTMERKER_CLEARINGSTELLE "700000004"

Definition des Standard Testmerkers. Bei der Verwendung dieses Testmerkers werden die Fälle in der Clearingstelle aussortiert und verworfen. Es findet keine Verarbeitung im Finanzamt statt.

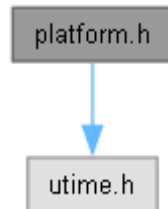
#define ERIC_TESTMERKER_ECC "700000001"

Definition des Testmerkers für das ECC. Bei der Verwendung dieses Testmerkers werden die Fälle in der Landeskopfstelle bzw. dem ECC aussortiert und verworfen. Es findet keine Verarbeitung im Finanzamt statt.

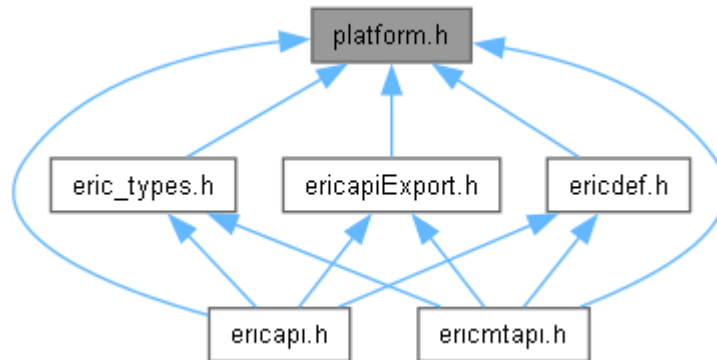
platform.h-Dateireferenz

```
#include <utime.h>
```

Include-Abhängigkeitsdiagramm für platform.h:



Dieser Graph zeigt, welche Datei direkt oder indirekt diese Datei enthält:



Typdefinitionen

- typedef __plattformabhaengige_Implementierung__ [uint32_t](#)

Ausführliche Beschreibung

Konstanten für verschiedene Betriebssysteme

Dokumentation der benutzerdefinierten Typen

typedef __plattformabhaengige_Implementierung__ [uint32_t](#)

Definition eines vorzeichenlosen, 32 Bit breiten Integer-Typs.

Siehe Quellcode von [platform.h](#) für Implementierung.

Index

abteilung	
eric_zertifikat_parameter_t	15
adresse	
eric_zertifikat_parameter_t	16
authentifizierungsmethode	
OttoProxyKonfiguration	19
benutzerName	
OttoProxyKonfiguration	19
benutzerPasswort	
OttoProxyKonfiguration	19
beschreibung	
eric_zertifikat_parameter_t	16
byteChar	
eric_types.h	277
duplexDruck	
eric_druck_parameter_t	9
email	
eric_zertifikat_parameter_t	16
eric_bearbeitung_flag_t	
eric_types.h	282
ERIC_CRYPT_CORRUPTED	
eric_fehlercodes.h	42
ERIC_CRYPT_E_ASN1_NO_CONTENT_DATA	
eric_fehlercodes.h	44
ERIC_CRYPT_E_ASN1_NO_ENVELOPED_DATA	
eric_fehlercodes.h	44
ERIC_CRYPT_E_ASN1_READ_BUFFER_TOO_SMALL	
eric_fehlercodes.h	43
ERIC_CRYPT_E_ASN1_READ_DATA_INCOMPLETE	
eric_fehlercodes.h	43
ERIC_CRYPT_E_BUSY	
eric_fehlercodes.h	38
ERIC_CRYPT_E_DATA_NOT_INITIALIZED	
eric_fehlercodes.h	43
ERIC_CRYPT_E_DECRYPT	
eric_fehlercodes.h	40
ERIC_CRYPT_E_ENCODE_ERROR	
eric_fehlercodes.h	40
ERIC_CRYPT_E_ENCODE_UNKNOWN	
eric_fehlercodes.h	40
ERIC_CRYPT_E_ENCRYPT	
eric_fehlercodes.h	40
ERIC_CRYPT_E_ESIGNER_EXCEPTION	
eric_fehlercodes.h	40
ERIC_CRYPT_E_ESIGNER_NOT_LOADED	
eric_fehlercodes.h	40
ERIC_CRYPT_E_INCOMPATIBLE_SIGNER_VERSION	
eric_fehlercodes.h	41
ERIC_CRYPT_E_INTERNAL	
eric_fehlercodes.h	42
ERIC_CRYPT_E_INVALID_HANDLE	
eric_fehlercodes.h	38
ERIC_CRYPT_E_LOAD_DLL	
eric_fehlercodes.h	40
ERIC_CRYPT_E_MAX_SESSION	
eric_fehlercodes.h	38
ERIC_CRYPT_E_NO_SERVICE	
eric_fehlercodes.h	40
ERIC_CRYPT_E_NO_SIG_ENC_KEY	
eric_fehlercodes.h	40
ERIC_CRYPT_E_OUT_OF_MEM	

eric_fehlercodes.h 38	ERIC_CRYPT_E_P7_RECIPIENT
ERIC_CRYPT_E_P11_ENC_KEY	eric_fehlercodes.h 39
eric_fehlercodes.h 39	ERIC_CRYPT_E_PIN_LOCKED
ERIC_CRYPT_E_P11_ENGINE_LOADED	eric_fehlercodes.h 39
eric_fehlercodes.h 41	ERIC_CRYPT_E_PIN_WRONG
ERIC_CRYPT_E_P11_INIT_FAILED	eric_fehlercodes.h 38
eric_fehlercodes.h 43	ERIC_CRYPT_E_PSE_PATH
ERIC_CRYPT_E_P11_NO_ENC_CERT	eric_fehlercodes.h 38
eric_fehlercodes.h 43	ERIC_CRYPT_E_SC_ENC_KEY
ERIC_CRYPT_E_P11_NO_SIG_CERT	eric_fehlercodes.h 43
eric_fehlercodes.h 42	ERIC_CRYPT_E_SC_INIT_FAILED
ERIC_CRYPT_E_P11_SIG_KEY	eric_fehlercodes.h 43
eric_fehlercodes.h 39	ERIC_CRYPT_E_SC_NO_APPLET
ERIC_CRYPT_E_P11_SLOT_EMPTY	eric_fehlercodes.h 42
eric_fehlercodes.h 40	ERIC_CRYPT_E_SC_NO_ENC_CERT
ERIC_CRYPT_E_P12_CREATE	eric_fehlercodes.h 43
eric_fehlercodes.h 41	ERIC_CRYPT_E_SC_NO_SIG_CERT
ERIC_CRYPT_E_P12_DECODE	eric_fehlercodes.h 43
eric_fehlercodes.h 39	ERIC_CRYPT_E_SC_SESSION
ERIC_CRYPT_E_P12_ENC_KEY	eric_fehlercodes.h 42
eric_fehlercodes.h 39	ERIC_CRYPT_E_SC_SIG_KEY
ERIC_CRYPT_E_P12_NO_ENC_CERT	eric_fehlercodes.h 43
eric_fehlercodes.h 43	ERIC_CRYPT_E_SC_SLOT_EMPTY
ERIC_CRYPT_E_P12_NO_SIG_CERT	eric_fehlercodes.h 42
eric_fehlercodes.h 43	ERIC_CRYPT_E_TOKEN_TYPE_MISMATCH
ERIC_CRYPT_E_P12_READ	eric_fehlercodes.h 41
eric_fehlercodes.h 39	ERIC_CRYPT_E_USER_CANCEL
ERIC_CRYPT_E_P12_SIG_KEY	eric_fehlercodes.h 41
eric_fehlercodes.h 39	ERIC_CRYPT_E_VERALTETE_ESIGNER_VERSION
ERIC_CRYPT_E_P7_DECODE	eric_fehlercodes.h 41
eric_fehlercodes.h 39	ERIC_CRYPT_E_VERIFY_CERT_CHAIN
ERIC_CRYPT_E_P7_READ	eric_fehlercodes.h 41
eric_fehlercodes.h 39	ERIC_CRYPT_E_XML_INIT
	eric_fehlercodes.h 40

ERIC_CRYPT_E_XML_PARSE	pdfCallback 10
eric_fehlercodes.h 39	pdfCallbackBenutzerdaten 10
ERIC_CRYPT_E_XML_SIG_ADD	pdfName 10
eric_fehlercodes.h 39	version 11
ERIC_CRYPT_E_XML_SIG_SIGN	vorschau 11
eric_fehlercodes.h 40	ERIC_DRUCKE
ERIC_CRYPT_E_XML_SIG_TAG	eric_types.h 282
eric_fehlercodes.h 39	eric_fehlercode
ERIC_CRYPT_EIDKARTE_NICHT_UNTERSTUETZT	eric_fehlercodes.h 26
eric_fehlercodes.h 42	eric_fehlercode_t
ERIC_CRYPT_ERROR_CREATE_KEY	eric_fehlercodes.h 26
eric_fehlercodes.h 38	eric_fehlercodes.h 21
ERIC_CRYPT_NICHT_UNTERSTUETZTES_PSE_FORMAT	ERIC_CRYPT_CORRUPTED 42
eric_fehlercodes.h 41	ERIC_CRYPT_E_ASN1_NO_CONTENT_DATA 44
ERIC_CRYPT_PIN_BENOETIGT	ERIC_CRYPT_E_ASN1_NO_ENVELOPED_DATA 44
eric_fehlercodes.h 41	ERIC_CRYPT_E_ASN1_READ_BUFFER_TOO_SMALL 43
ERIC_CRYPT_PIN_ENTHAELT_UNGUELTIGE_ZEICHEN	ERIC_CRYPT_E_ASN1_READ_DATA_INCOMPLETE 43
eric_fehlercodes.h 42	ERIC_CRYPT_E_BUSY 38
ERIC_CRYPT_PIN_STAERKE_NICHT_AUSREICHEND	ERIC_CRYPT_E_DATA_NOT_INITIALIZED 43
eric_fehlercodes.h 42	ERIC_CRYPT_E_DECRYPT 40
ERIC_CRYPT_SIGNATUR	ERIC_CRYPT_E_ENCODE_ERROR 40
eric_fehlercodes.h 41	ERIC_CRYPT_E_ENCODE_UNKNOWN 40
ERIC_CRYPT_ZERTIFIKAT	ERIC_CRYPT_E_ENCRYPT 40
eric_fehlercodes.h 41	ERIC_CRYPT_E_ESICL_EXCEPTION 40
ERIC_CRYPT_ZERTIFIKATSDATEI_EXISTIERT_BEREITS	ERIC_CRYPT_E_ESIGNER_NICHT_GELADEN 40
eric_fehlercodes.h 42	ERIC_CRYPT_E_INKOMPATIBLE_ESIGNER_VERSION 41
ERIC_CRYPT_ZERTIFIKATSPFAD_K EIN_VERZEICHNIS	ERIC_CRYPT_E_INTERN 42
eric_fehlercodes.h 42	ERIC_CRYPT_E_INVALID_HANDLE 38
eric_druck_parameter_t 8	
duplexDruck 9	
fussText 10	

ERIC_CRYPT_E_LOAD_DLL 40	ERIC_CRYPT_E_PIN_LOCKED 39
ERIC_CRYPT_E_MAX_SESSION 38	ERIC_CRYPT_E_PIN_WRONG 38
ERIC_CRYPT_E_NO_SERVICE 40	ERIC_CRYPT_E_PSE_PATH 38
ERIC_CRYPT_E_NO_SIG_ENC_KEY 40	ERIC_CRYPT_E_SC_ENC_KEY 43
ERIC_CRYPT_E_OUT_OF_MEM 38	ERIC_CRYPT_E_SC_INIT_FAILED 43
ERIC_CRYPT_E_P11_ENC_KEY 39	ERIC_CRYPT_E_SC_NO_APPLET 42
ERIC_CRYPT_E_P11_ENGINE_LOADED 41	ERIC_CRYPT_E_SC_NO_ENC_CERT 43
ERIC_CRYPT_E_P11_INIT_FAILED 43	ERIC_CRYPT_E_SC_NO_SIG_CERT 43
ERIC_CRYPT_E_P11_NO_ENC_CERT 43	ERIC_CRYPT_E_SC_SESSION 42
ERIC_CRYPT_E_P11_NO_SIG_CERT 42	ERIC_CRYPT_E_SC_SIG_KEY 43
ERIC_CRYPT_E_P11_SIG_KEY 39	ERIC_CRYPT_E_SC_SLOT_EMPTY 42
ERIC_CRYPT_E_P11_SLOT_EMPTY 40	ERIC_CRYPT_E_TOKEN_TYPE_MISMATCH 41
ERIC_CRYPT_E_P12_CREATE 41	ERIC_CRYPT_E_USER_CANCEL 41
ERIC_CRYPT_E_P12_DECODE 39	ERIC_CRYPT_E_VERALTETE_ESIGNER_VERSION 41
ERIC_CRYPT_E_P12_ENC_KEY 39	ERIC_CRYPT_E_VERIFY_CERT_CHAIN 41
ERIC_CRYPT_E_P12_NO_ENC_CERT 43	ERIC_CRYPT_E_XML_INIT 40
ERIC_CRYPT_E_P12_NO_SIG_CERT 43	ERIC_CRYPT_E_XML_PARSE 39
ERIC_CRYPT_E_P12_READ 39	ERIC_CRYPT_E_XML_SIG_ADD 39
ERIC_CRYPT_E_P12_SIG_KEY 39	ERIC_CRYPT_E_XML_SIG_SIGN 40
ERIC_CRYPT_E_P7_DECODE 39	ERIC_CRYPT_E_XML_SIG_TAG 39
ERIC_CRYPT_E_P7_READ 39	ERIC_CRYPT_EIDKARTE_NICHT_UNTERSTUETZT 42
ERIC_CRYPT_E_P7_RECIPIENT 39	

ERIC_CRYPT_ERROR_CREATE_KEY 38	ERIC_GLOBAL_DATENSATZ_ZU_GROSS 28
ERIC_CRYPT_NICHT_UNTERSTUETZTES_PSE_FORMAT 41	ERIC_GLOBAL_DRUCK_FUER_VERFAHREN_NICHT_ERLAUBT 31
ERIC_CRYPT_PIN_BENOETIGT 41	ERIC_GLOBAL_ECHTFALL_NICHT_ERLAUBT 28
ERIC_CRYPT_PIN_ENTHAELT_UNGUELTIGE_ZEICHEN 42	ERIC_GLOBAL_EINSTELLUNG_NAMME_UNGUELTIG 34
ERIC_CRYPT_PIN_STAERKE_NICHT_AUSREICHEND 42	ERIC_GLOBAL_EINSTELLUNG_WERT_UNGUELTIG 34
ERIC_CRYPT_SIGNATUR 41	ERIC_GLOBAL_ERR_DEKODIEREN 34
ERIC_CRYPT_ZERTIFIKAT 41	ERIC_GLOBAL_ERROR_XML_CREATE 28
ERIC_CRYPT_ZERTIFIKATSDATEI_EXISTIERT_BEREITS 42	ERIC_GLOBAL_EWAZ_LANDESKUERZEL_UNBEKANNT 33
ERIC_CRYPT_ZERTIFIKATSPFAD_KEIN_VERZEICHNIS 42	ERIC_GLOBAL_EWAZ_UNGUELTIG 33
eric_fehlercode 26	ERIC_GLOBAL_FEHLER_INITIALISIERUNG 31
eric_fehlercode_t 26	ERIC_GLOBAL_FEHLERMELDUNG_NICHT_VORHANDEN 27
ERIC_GLOBAL_ARITHMETIKFEHLER 29	ERIC_GLOBAL_FUNKTION_NICHT_ERLAUBT 28
ERIC_GLOBAL_BIC_FORMALER_FEHLER 33	ERIC_GLOBAL_FUNKTION_NICHT_UNTERSTUETZT 34
ERIC_GLOBAL_BIC_LAENDERCODE_FEHLER 33	ERIC_GLOBAL_HERSTELLER_ID_NICHT_ERLAUBT 27
ERIC_GLOBAL_BUFANR_UNBEKANNT 29	ERIC_GLOBAL_HINWEISE 27
ERIC_GLOBAL_BUNDESLAENDER_UNEINHEITLICH 34	ERIC_GLOBAL_IBAN_FORMALER_FEHLER 32
ERIC_GLOBAL_CHECK_CORRUPTED_NDS 31	ERIC_GLOBAL_IBAN_LAENDERCODE_FEHLER 33
ERIC_GLOBAL_COMMONDATA_NICHT_VERFUEGBAR 30	ERIC_GLOBAL_IBAN_LANDESFORMAT_FEHLER 33
ERIC_GLOBAL_DATEI_NICHT_GEFUNDEN 27	ERIC_GLOBAL_IBAN_PRUEFZIFFER_FEHLER 33
ERIC_GLOBAL_DATEIZUGRIFF_VERWEIGERT 30	ERIC_GLOBAL_IDNUMMER_UNGUELTIG 33
ERIC_GLOBAL_DATENARTVERSION_UNBEKANNT 29	
ERIC_GLOBAL_DATENARTVERSION_XML_INKONSISTENT 30	

ERIC_GLOBAL_ILLEGAL_STATE 27	ERIC_GLOBAL_PLUGININITIALISIERUNG 32
ERIC_GLOBAL_INKOMPATIBLE_VERSIONEN 32	ERIC_GLOBAL_PRUEF_FEHLER 27
ERIC_GLOBAL_INTERNER_FEHLER 28	ERIC_GLOBAL_PUFFER_UEBERLAUF 29
ERIC_GLOBAL_KEINE_DATEN_VORHANDEN 27	ERIC_GLOBAL_PUFFER_UNGLEICHER_INSTANZ 30
ERIC_GLOBAL_LANDESNUMMER_BUFANR 29	ERIC_GLOBAL_PUFFER_ZUGRIFFSKONFLIKT 29
ERIC_GLOBAL_LANDESNUMMER_UNBEKANNT 29	ERIC_GLOBAL_SEND_FLAG_MEHRALSEINES 31
ERIC_GLOBAL_LOG_EXCEPTION 30	ERIC_GLOBAL_STEUERNUMMER_FALSCHE_LAENGE 29
ERIC_GLOBAL_MEHRFACHAUFRUFE_NICHT_UNTERSTUETZT 32	ERIC_GLOBAL_STEUERNUMMER_NICHT_NUMERISCH 29
ERIC_GLOBAL_MEHRFACHE_INITIALISIERUNG 31	ERIC_GLOBAL_STEUERNUMMER_UNGUELTIG 29
ERIC_GLOBAL_NICHT_GENUEGEND_ARBEITSSPEICHER 27	ERIC_GLOBAL_TESTMERKER_UNGUELTIG 28
ERIC_GLOBAL_NICHT_INITIALISIERT 31	ERIC_GLOBAL_TEXTPUFFERGRUESSE_FIX 28
ERIC_GLOBAL_NO_VERSAND_IN_BETA_VERSION 28	ERIC_GLOBAL_TRANSFERHANDLE 32
ERIC_GLOBAL_NULL_PARAMETER 33	ERIC_GLOBAL_TRANSPORTSCHLUESSEL_NICHT_ERLAUBT 30
ERIC_GLOBAL_NUR_PORTALZERTIFIKAT_ERLAUBT 28	ERIC_GLOBAL_TRANSPORTSCHLUESSEL_TYP_FALSCH 30
ERIC_GLOBAL_NUTZDATENHEADER_EMPFAENGER_NICHT_KORREKT 35	ERIC_GLOBAL_UNGUELTIGE_FLAG_KOMBINATION 31
ERIC_GLOBAL_NUTZDATENHEADER_VERSIONEN_UNEINHEITLICH 34	ERIC_GLOBAL_UNGUELTIGE_INSTANZ 30
ERIC_GLOBAL_NUTZDATENTICKETS_NICHT_EINDEUTIG 34	ERIC_GLOBAL_UNGUELTIGE_PARAMETER_VERSION 32
ERIC_GLOBAL_OEFFENTLICHER_SCHLUESSEL_UNGUELTIG 30	ERIC_GLOBAL_UNGUELTIGER_PARAMETER 31
	ERIC_GLOBAL_UNKNOWN 27
	ERIC_GLOBAL_UNKNOWN_PARAMETER_ERROR 31

ERIC_GLOBAL_UPDATE_NECES SARY 33	ERIC_IO_READER_FALSCHES_E NCODING 45
ERIC_GLOBAL_UTI_COUNTRY_N OT_SUPPORTED 32	ERIC_IO_READER_FORMALE_FE HLER 45
ERIC_GLOBAL_VERSAND_ART_N ICHT_UNTERSTUETZT 32	ERIC_IO_READER_KEINE_RABEI D 46
ERIC_GLOBAL_VERSCHLUESSEL UNGS_PARAMETER_NICHT_AN GEGEBEN 31	ERIC_IO_READER_MEHRFACHE_ NUTZDATEN_ELEMENTE 45
ERIC_GLOBAL_VERSCHLUESSEL UNGS_PARAMETER_NICHT_ER LAUBT 28	ERIC_IO_READER_MEHRFACHE_ NUTZDATENBLOCK_ELEMENT E 45
ERIC_GLOBAL_VERSCHLUESSEL UNGSVERFAHREN_NICHT_UN TERSTUETZT 32	ERIC_IO_READER_MEHRFACHE_ STEUERFAELLE 45
ERIC_GLOBAL_VORSATZ_UNGU ELTIG 30	ERIC_IO_READER_RABE_FEHLE R 46
ERIC_GLOBAL_ZEITRAEUME_UN EINHEITLICH 34	ERIC_IO_READER_RABE_REFER ENZID_NICHT_ERLAUBT 46
ERIC_GLOBAL_ZULASSUNGSNU MMER_ZU_LANG 33	ERIC_IO_READER_RABE_REFER ENZID_UNGUELTIG 46
ERIC_IO_DATEI_INKORREKT 44	ERIC_IO_READER_RABE_REFER ENZIDS_NICHT_EINDEUTIG 47
ERIC_IO_DATENTEILENDNOTFO UND 47	ERIC_IO_READER_RABE_VERIFI KATIONSID_UNGUELTIG 46
ERIC_IO_DATENTEILNOTFOUND 47	ERIC_IO_READER_RABEID_UNG UELTIG 46
ERIC_IO_FALSCHES_VERFAHRE N 44	ERIC_IO_READER_SCHEMA_VAL IDIERUNGSFEHLER 47
ERIC_IO_FEHLER 44	ERIC_IO_READER_STEUERZEIC HEN_IM_NUTZDATENHEADER 46
ERIC_IO_MASTERDATENSERVIC E_NICHT_VERFUEGBAR 44	ERIC_IO_READER_STEUERZEIC HEN_IM_TRANSFERHEADER 46
ERIC_IO_NDS_GENERIERUNG_F EHLGESCHLAGEN 44	ERIC_IO_READER_STEUERZEIC HEN_IN_DEN_NUTZDATEN 46
ERIC_IO_PARSE_FEHLER 44	ERIC_IO_READER_UNBEKANNT E_XML_ENTITY 47
ERIC_IO_READER_ANHAENGE_Z U_GROSS 47	ERIC_IO_READER_UNERWARTE TE_ELEMENTE 45
ERIC_IO_READER_ANHANG_ZU_ GROSS 47	
ERIC_IO_READER_ANHANG_ZU_ KLEIN 47	

ERIC_IO_READER_UNTERSACHB EREICH_UNGUELTIG 45	ERIC_PRINT_UNGUELTIGER_DA TEI_PFAD 48
ERIC_IO_READER_ZU_VIELE_AN HAENGE 47	ERIC_TRANSFER_COM_ERROR 35
ERIC_IO_READER_ZU_VIELE_NU TZDATENBLOCK_ELEMENTE 45	ERIC_TRANSFER_EID_CLIENTFE HLER 38
ERIC_IO_STEUERZEICHEN_IM_N DS 44	ERIC_TRANSFER_EID_FEHLEND EFELDER 38
ERIC_IO_TESTHERSTELLERID_G ESPERRT 47	ERIC_TRANSFER_EID_IDENTIFIK ATIONABGEBROCHEN 38
ERIC_IO_UEBERGABEPARAMET ER_FEHLERHAFT 48	ERIC_TRANSFER_EID_IDNRNICH TEINDEUTIG 37
ERIC_IO_UNBEKANNTE_DATENA RT 45	ERIC_TRANSFER_EID_KEINCLIE NT 37
ERIC_IO_UNGUELTIGE_UTF8_SE QUENZ 48	ERIC_TRANSFER_EID_KEINKONT O 37
ERIC_IO_UNGUELTIGE_ZEICHEN _IN_PARAMETER 48	ERIC_TRANSFER_EID_NPABLOC KIERT 38
ERIC_IO_VERSIONSINFORMATIO NEN_NICHT_GEFUNDEN 44	ERIC_TRANSFER_EID_SERVERF EHLER 37
ERIC_OK 26	ERIC_TRANSFER_EID_ZERTIFIKA TFEHLER 37
ERIC_PRINT_ABBRUCH_DRUCKV ORBEREITUNG 48	ERIC_TRANSFER_ERR_BEGINDA TENGROESSE 36
ERIC_PRINT_ABBRUCH_GENERI ERUNG 48	ERIC_TRANSFER_ERR_BEGINDA TENLIEFERANT 35
ERIC_PRINT_AUSGABEZIEL_UNB EKANNT 48	ERIC_TRANSFER_ERR_BEGINTR ANSPORTSCHLUESSEL 35
ERIC_PRINT_DRUCKVORLAGE_N ICHT_GEFUNDEN 48	ERIC_TRANSFER_ERR_CONNEC TSERVER 36
ERIC_PRINT_FUSSTEXT_ZU_LAN G 49	ERIC_TRANSFER_ERR_DATENTE ILENDNOTFOUND 35
ERIC_PRINT_INITIALIZIERUNG_F EHLERHAFT 48	ERIC_TRANSFER_ERR_DATENTE ILFEHLER 37
ERIC_PRINT_INTERNER_FEHLER 48	ERIC_TRANSFER_ERR_ENDDAT ENGROESSE 36
ERIC_PRINT_PDFCALLBACK 49	ERIC_TRANSFER_ERR_ENDDAT ENLIEFERANT 35
ERIC_PRINT_STEUERFALL_NICH T_UNTERSTUETZT 49	ERIC_TRANSFER_ERR_ENDSIGU SER 37

ERIC_TRANSFER_ERR_ENDTRANSPORTSCHLUESSEL	35	eric_types.h	282
ERIC_TRANSFER_ERR_NORESPONSE	36	eric_types.h	281
ERIC_TRANSFER_ERR_NOTENCRYPTED	36	eric_types.h	281
ERIC_TRANSFER_ERR_OTHER	36	eric_types.h	281
ERIC_TRANSFER_ERR_PARAMETER	35	eric_fehlercodes.h	29
ERIC_TRANSFER_ERR_PROXYAUTH	36	eric_fehlercodes.h	33
ERIC_TRANSFER_ERR_PROXYCONNECT	36	eric_fehlercodes.h	33
ERIC_TRANSFER_ERR_PROXYPORT_INVALID	36	eric_fehlercodes.h	33
ERIC_TRANSFER_ERR_SEND	36	eric_fehlercodes.h	29
ERIC_TRANSFER_ERR_SEND_INIT	36	eric_fehlercodes.h	29
ERIC_TRANSFER_ERR_TIMEOUT	36	eric_fehlercodes.h	34
ERIC_TRANSFER_ERR_XML_ENCODING	37	eric_fehlercodes.h	31
ERIC_TRANSFER_ERR_XML_HEADER	37	eric_fehlercodes.h	30
ERIC_TRANSFER_ERR_XML_HEADER	35	eric_fehlercodes.h	30
ERIC_TRANSFER_ERR_XMLTAG_NICHT_GEFUNDEN	37	eric_fehlercodes.h	27
ERIC_TRANSFER_VORGANG_NICHT_UNTERSTUETZT	35	eric_fehlercodes.h	30
ERIC_FORTSCHRITTCALLBACK_ID_DRUCKEN	eric_types.h	282	
ERIC_FORTSCHRITTCALLBACK_ID_EINLESEN	eric_types.h	281	
ERIC_FORTSCHRITTCALLBACK_ID_SENDEN			
ERIC_FORTSCHRITTCALLBACK_ID_VALIDIEREN			
ERIC_FORTSCHRITTCALLBACK_ID_VORBEREITEN			
ERIC_GLOBAL_ARITHMETIKFEHLER			
ERIC_GLOBAL_BIC_FORMALER_FEHLER			
ERIC_GLOBAL_BIC_LAENDERCODE_FEHLER			
ERIC_GLOBAL_BUFANR_UNBEKANNT			
ERIC_GLOBAL_BUNDESLAENDER_UNEINHEITLICH			
ERIC_GLOBAL_CHECK_CORRUPTED_NDS			
ERIC_GLOBAL_COMMONDATA_NICHT_VERFUEGBAR			
ERIC_GLOBAL_DATEI_NICHT_GEFUNDEN			
ERIC_GLOBAL_DATEIZUGRIFF_VERWEIGERT			
ERIC_GLOBAL_DATENARTVERSION_UNBEKANNT			
ERIC_GLOBAL_DATENARTVERSION_XML_INKONSISTENT			

eric_fehlercodes.h	30	ERIC_GLOBAL_HERSTELLER_ID_NI	
ERIC_GLOBAL_DATENSATZ_ZU_G		CHT_ERLAUBT	
ROSS		eric_fehlercodes.h	27
eric_fehlercodes.h	28	ERIC_GLOBAL_HINWEISE	
ERIC_GLOBAL_DRUCK_FUER_VER		eric_fehlercodes.h	27
FAHREN_NICHT_ERLAUBT		ERIC_GLOBAL_IBAN_FORMALER_F	
eric_fehlercodes.h	31	EHLER	
ERIC_GLOBAL_ECHTFALL_NICHT_		eric_fehlercodes.h	32
ERLAUBT		ERIC_GLOBAL_IBAN_LAENDERCO	
eric_fehlercodes.h	28	DE_FEHLER	
ERIC_GLOBAL_EINSTELLUNG_NA		eric_fehlercodes.h	33
ME_UNGUELTIG		ERIC_GLOBAL_IBAN_LANDESFOR	
eric_fehlercodes.h	34	MAT_FEHLER	
ERIC_GLOBAL_EINSTELLUNG_WE		eric_fehlercodes.h	33
RT_UNGUELTIG		ERIC_GLOBAL_IBAN_PRUEFZIFFE	
eric_fehlercodes.h	34	R_FEHLER	
ERIC_GLOBAL_ERR_DEKODIEREN		eric_fehlercodes.h	33
eric_fehlercodes.h	34	ERIC_GLOBAL_IDNUMMER_UNGUE	
ERIC_GLOBAL_ERROR_XML_CREA		LTIG	
TE		eric_fehlercodes.h	33
eric_fehlercodes.h	28	ERIC_GLOBAL_ILLEGAL_STATE	
ERIC_GLOBAL_EWAZ_LANDESKUE		eric_fehlercodes.h	27
RZEL_UNBEKANNT		ERIC_GLOBAL_INKOMPATIBLE_VE	
eric_fehlercodes.h	33	RSIONEN	
ERIC_GLOBAL_EWAZ_UNGUELTIG		eric_fehlercodes.h	32
eric_fehlercodes.h	33	ERIC_GLOBAL_INTERNER_FEHLER	
ERIC_GLOBAL_FEHLER_INITIALISI		eric_fehlercodes.h	28
ERUNG		ERIC_GLOBAL_KEINE_DATEN_VOR	
eric_fehlercodes.h	31	HANDEN	
ERIC_GLOBAL_FEHLERMELDUNG_		eric_fehlercodes.h	27
NICHT_VORHANDEN		ERIC_GLOBAL_LANDESNUMMER_	
eric_fehlercodes.h	27	BUFANR	
ERIC_GLOBAL_FUNKTION_NICHT_		eric_fehlercodes.h	29
ERLAUBT		ERIC_GLOBAL_LANDESNUMMER_	
eric_fehlercodes.h	28	UNBEKANNT	
ERIC_GLOBAL_FUNKTION_NICHT_		eric_fehlercodes.h	29
UNTERSTUETZT		ERIC_GLOBAL_LOG_EXCEPTION	
eric_fehlercodes.h	34	eric_fehlercodes.h	30

ERIC_GLOBAL_MEHRFACHAUFRUF E_NICHT_UNTERSTUETZT eric_fehlercodes.h 32	ERIC_GLOBAL_PUFFER_UEBERLAUF eric_fehlercodes.h 29
ERIC_GLOBAL_MEHRFACHE_INITIA LISIERUNG eric_fehlercodes.h 31	ERIC_GLOBAL_PUFFER_UNGLEICH ER_INSTANZ eric_fehlercodes.h 30
ERIC_GLOBAL_NICHT_GENUEGEN D_ARBEITSSPEICHER eric_fehlercodes.h 27	ERIC_GLOBAL_PUFFER_ZUGRIFFS KONFLIKT eric_fehlercodes.h 29
ERIC_GLOBAL_NICHT_INITIALISIER T eric_fehlercodes.h 31	ERIC_GLOBAL_SEND_FLAG_MEHR _ALS_EINES eric_fehlercodes.h 31
ERIC_GLOBAL_NO_VERSAND_IN_B ETA_VERSION eric_fehlercodes.h 28	ERIC_GLOBAL_STEUERNUMMER_F ALSCHE_LAENGE eric_fehlercodes.h 29
ERIC_GLOBAL_NULL_PARAMETER eric_fehlercodes.h 33	ERIC_GLOBAL_STEUERNUMMER_ NICHT_NUMERISCH eric_fehlercodes.h 29
ERIC_GLOBAL_NUR_PORTALZERTI FIKAT_ERLAUBT eric_fehlercodes.h 28	ERIC_GLOBAL_STEUERNUMMER_ UNGUELTIG eric_fehlercodes.h 29
ERIC_GLOBAL_NUTZDATENHEADE R_EMPFAENGER_NICHT_KORRE KT eric_fehlercodes.h 35	ERIC_GLOBAL_TESTMERKER_UNG UELTIG eric_fehlercodes.h 28
ERIC_GLOBAL_NUTZDATENHEADE RVERSIONEN_UNEINHEITLICH eric_fehlercodes.h 34	ERIC_GLOBAL_TEXTPUFFERGROE SSE_FIX eric_fehlercodes.h 28
ERIC_GLOBAL_NUTZDATENTICKET S_NICHT_EINDEUTIG eric_fehlercodes.h 34	ERIC_GLOBAL_TRANSFERHANDLE eric_fehlercodes.h 32
ERIC_GLOBAL_OEFFENTLICHER_S CHLUESSEL_UNGUELTIG eric_fehlercodes.h 30	ERIC_GLOBAL_TRANSPORTSCHLU ESSEL_NICHT_ERLAUBT eric_fehlercodes.h 30
ERIC_GLOBAL_PLUGININITIALISIE RUNG eric_fehlercodes.h 32	ERIC_GLOBAL_TRANSPORTSCHLU ESSEL_TYP_FALSCH eric_fehlercodes.h 30
ERIC_GLOBAL_PRUEF_FEHLER eric_fehlercodes.h 27	ERIC_GLOBAL_UNGUELTIGE_FLAG _KOMBINATION eric_fehlercodes.h 31

ERIC_GLOBAL_UNGUELTIGE_INSTANZ eric_fehlercodes.h 30	ERIC_GLOBAL_ZEITRAEUME_UNEINGHEITLICH eric_fehlercodes.h 34
ERIC_GLOBAL_UNGUELTIGE_PARAMETER_VERSION eric_fehlercodes.h 32	ERIC_GLOBAL_ZULASSUNGSNUMMER_ZU_LANG eric_fehlercodes.h 33
ERIC_GLOBAL_UNGUELTIGER_PARAMETER eric_fehlercodes.h 31	ERIC_IO_DATEI_INKORREKT eric_fehlercodes.h 44
ERIC_GLOBAL_UNKNOWN eric_fehlercodes.h 27	ERIC_IO_DATENTEILENDNOTFOUND eric_fehlercodes.h 47
ERIC_GLOBAL_UNKNOWN_PARAMETER_ERROR eric_fehlercodes.h 31	ERIC_IO_DATENTEILNOTFOUND eric_fehlercodes.h 47
ERIC_GLOBAL_UPDATE_NECESSARY eric_fehlercodes.h 33	ERIC_IO_FALSCHES_VERFAHREN eric_fehlercodes.h 44
ERIC_GLOBAL_UTI_COUNTRY_NOT_SUPPORTED eric_fehlercodes.h 32	ERIC_IO_FEHLER eric_fehlercodes.h 44
ERIC_GLOBAL_VERSAND_ART_NICHT_UNTERSTUETZT eric_fehlercodes.h 32	ERIC_IO_MASTERDATENSERVICE_NICHT_VERFUEGBAR eric_fehlercodes.h 44
ERIC_GLOBAL_VERSCHLUESSELUNGS_PARAMETER_NICHT_ANGEGEBEN eric_fehlercodes.h 31	ERIC_IO_NDS_GENERIERUNG_FEHLGESCHLAGEN eric_fehlercodes.h 44
ERIC_GLOBAL_VERSCHLUESSELUNGS_PARAMETER_NICHT_ERLAUBT eric_fehlercodes.h 28	ERIC_IO_PARSE_FEHLER eric_fehlercodes.h 44
ERIC_GLOBAL_VERSCHLUESSELUNGSVERFAHREN_NICHT_UNTERSTUETZT eric_fehlercodes.h 32	ERIC_IO_READER_ANHAENGE_ZU_GROSS eric_fehlercodes.h 47
ERIC_GLOBAL_VORSATZ_UNGUELTIG eric_fehlercodes.h 30	ERIC_IO_READER_ANHANG_ZU_GROSS eric_fehlercodes.h 47
	ERIC_IO_READER_ANHANG_ZU_KLEIN eric_fehlercodes.h 47
	ERIC_IO_READER_FALSCHES_ENCODING eric_fehlercodes.h 45
	ERIC_IO_READER_FORMALE_FEHLER

eric_fehlercodes.h 45	ERIC_IO_READER_STEUERZEICHEN_IN_DEN_NUTZDATEN
ERIC_IO_READER_KEINE_RABEID	eric_fehlercodes.h 46
eric_fehlercodes.h 46	ERIC_IO_READER_UNBEKANNTE_XML_ENTITY
ERIC_IO_READER_MEHRFACHE_NUTZDATEN_ELEMENTE	eric_fehlercodes.h 47
eric_fehlercodes.h 45	ERIC_IO_READER_UNERWARTETE_ELEMENTE
ERIC_IO_READER_MEHRFACHE_NUTZDATENBLOCK_ELEMENTE	eric_fehlercodes.h 45
eric_fehlercodes.h 45	ERIC_IO_READER_UNTERSACHBEREICH_UNGUELTIG
ERIC_IO_READER_MEHRFACHE_STEUERFAELLE	eric_fehlercodes.h 45
eric_fehlercodes.h 45	ERIC_IO_READER_ZU_VIELE_ANHANGENGE
ERIC_IO_READER_RABE_FEHLER	eric_fehlercodes.h 47
eric_fehlercodes.h 46	ERIC_IO_READER_ZU_VIELE_NUTZDATENBLOCK_ELEMENTE
ERIC_IO_READER_RABE_REFERENZID_NICHT_ERLAUBT	eric_fehlercodes.h 45
eric_fehlercodes.h 46	ERIC_IO_STEUERZEICHEN_IM_ND
ERIC_IO_READER_RABE_REFERENZID_UNGUELTIG	S
eric_fehlercodes.h 46	eric_fehlercodes.h 44
ERIC_IO_READER_RABE_REFERENZIDS_NICHT_EINDEUTIG	ERIC_IO_TESTHERSTELLERID_GESPERRT
eric_fehlercodes.h 47	eric_fehlercodes.h 47
ERIC_IO_READER_RABE_VERIFIKATIONSID_UNGUELTIG	ERIC_IO_UEBERGABEPARAMETER_FEHLERHAFT
eric_fehlercodes.h 46	eric_fehlercodes.h 48
ERIC_IO_READER_RABEID_UNGUELTIG	ERIC_IO_UNBEKANNTE_DATENART
eric_fehlercodes.h 46	eric_fehlercodes.h 45
ERIC_IO_READER_SCHEMA_VALIDIERUNGSFEHLER	ERIC_IO_UNGUELTIGE_UTF8_SEQUENZ
eric_fehlercodes.h 47	eric_fehlercodes.h 48
ERIC_IO_READER_STEUERZEICHEN_IM_NUTZDATENHEADER	ERIC_IO_UNGUELTIGE_ZEICHEN_IN_PARAMETER
eric_fehlercodes.h 46	eric_fehlercodes.h 48
ERIC_IO_READER_STEUERZEICHEN_IM_TRANSFERHEADER	ERIC_IO_VERSIONSINFORMATION_EN_NICHT_GEFUNDEN
eric_fehlercodes.h 46	eric_fehlercodes.h 44

ERIC_LOG_DEBUG	eric_fehlercodes.h 48
eric_types.h 283	ERIC_PRINT_INTERNER_FEHLER
ERIC_LOG_ERROR	eric_fehlercodes.h 48
eric_types.h 283	ERIC_PRINT_PDFCALLBACK
ERIC_LOG_INFO	eric_fehlercodes.h 49
eric_types.h 283	ERIC_PRINT_STEUERFALL_NICHT_
eric_log_level_t	UNTERSTUETZT
eric_types.h 283	eric_fehlercodes.h 49
ERIC_LOG_TRACE	ERIC_PRINT_UNGUELTIGER_DATEI
eric_types.h 283	_PFAD
ERIC_LOG_WARN	eric_fehlercodes.h 48
eric_types.h 283	ERIC_PRUEFE_HINWEISE
ERIC_MAJOR_VERSION	eric_types.h 282
ericversion.h 209	ERIC_SENDE
ERIC_MAX_LAENGE_FUSSTEXT	eric_types.h 282
ericdef.h 287	ERIC_TESTMERKER_CLEARINGST
ERIC_MINOR_VERSION	ELLE
ericversion.h 209	ericdef.h 287
ERIC_OK	ERIC_TESTMERKER_ECC
eric_fehlercodes.h 26	ericdef.h 287
ERIC_PATCH_VERSION	ERIC_TRANSFER_COM_ERROR
ericversion.h 209	eric_fehlercodes.h 35
ERIC_PRINT_ABBRUCH_DRUCKVO	ERIC_TRANSFER_EID_CLIENTFEHL
RBEREITUNG	ER
eric_fehlercodes.h 48	eric_fehlercodes.h 38
ERIC_PRINT_ABBRUCH_GENERIER	ERIC_TRANSFER_EID_FEHLENDEF
UNG	ELDER
eric_fehlercodes.h 48	eric_fehlercodes.h 38
ERIC_PRINT_AUSGABEZIEL_UNBE	ERIC_TRANSFER_EID_IDENTIFIKAT
KANNT	IONABGEBROCHEN
eric_fehlercodes.h 48	eric_fehlercodes.h 38
ERIC_PRINT_DRUCKVORLAGE_NIC	ERIC_TRANSFER_EID_IDNRNICHT
HT_GEFUNDEN	EINDEUTIG
eric_fehlercodes.h 48	eric_fehlercodes.h 37
ERIC_PRINT_FUSSTEXT_ZU_LANG	ERIC_TRANSFER_EID_KEINCLIENT
eric_fehlercodes.h 49	eric_fehlercodes.h 37
ERIC_PRINT_INITIALIZIERUNG_FE	ERIC_TRANSFER_EID_KEINKONTO
HLERHAFT	eric_fehlercodes.h 37

ERIC_TRANSFER_EID_NPABLOCKI ERT eric_fehlercodes.h 38	ERIC_TRANSFER_ERR_NOESPON NSE eric_fehlercodes.h 36
ERIC_TRANSFER_EID_SERVERFEH LER eric_fehlercodes.h 37	ERIC_TRANSFER_ERR_NOTENCRY PTED eric_fehlercodes.h 36
ERIC_TRANSFER_EID_ZERTIFIKAT FEHLER eric_fehlercodes.h 37	ERIC_TRANSFER_ERR_OTHER eric_fehlercodes.h 36
ERIC_TRANSFER_ERR_BEGINDAT ENGROESSE eric_fehlercodes.h 36	ERIC_TRANSFER_ERR_PARAM eric_fehlercodes.h 35
ERIC_TRANSFER_ERR_BEGINDAT ENLIEFERANT eric_fehlercodes.h 35	ERIC_TRANSFER_ERR_PROXYAUT H eric_fehlercodes.h 36
ERIC_TRANSFER_ERR_BEGINTRA NSPORTSCHLUESSEL eric_fehlercodes.h 35	ERIC_TRANSFER_ERR_PROXYCO NNECT eric_fehlercodes.h 36
ERIC_TRANSFER_ERR_CONNECTS ERVER eric_fehlercodes.h 36	ERIC_TRANSFER_ERR_PROXYPOR T_INVALID eric_fehlercodes.h 36
ERIC_TRANSFER_ERR_DATENTEIL ENDNOTFOUND eric_fehlercodes.h 35	ERIC_TRANSFER_ERR_SEND eric_fehlercodes.h 36
ERIC_TRANSFER_ERR_DATENTEIL FEHLER eric_fehlercodes.h 37	ERIC_TRANSFER_ERR_SEND_INIT eric_fehlercodes.h 36
ERIC_TRANSFER_ERR_ENDDATEN GROESSE eric_fehlercodes.h 36	ERIC_TRANSFER_ERR_TIMEOUT eric_fehlercodes.h 36
ERIC_TRANSFER_ERR_ENDDATEN LIEFERANT eric_fehlercodes.h 35	ERIC_TRANSFER_ERR_XML_ENCO DING eric_fehlercodes.h 37
ERIC_TRANSFER_ERR_ENDSIGUS ER eric_fehlercodes.h 37	ERIC_TRANSFER_ERR_XML_NHEA DER eric_fehlercodes.h 37
ERIC_TRANSFER_ERR_ENDTRANS PORTSCHLUESSEL eric_fehlercodes.h 35	ERIC_TRANSFER_ERR_XML_THEA DER eric_fehlercodes.h 35
	ERIC_TRANSFER_ERR_XMLTAG_NI CHT_GEFUNDEN eric_fehlercodes.h 37
	ERIC_TRANSFER_VORGANG_NICH T_UNTERSTUETZT

eric_fehlercodes.h	35	eric_verschluesselungs_parameter_t	12
eric_types.h	275	pin	13
byteChar	277	version	13
eric_bearbeitung_flag_t	282	zertifikatHandle	13
ERIC_DRUCKE	282	eric_zertifikat_parameter_t	14
ERIC_FORTSCHRITTCALLBACK_I		abteilung	15
D_DRUCKEN	282	adresse	16
ERIC_FORTSCHRITTCALLBACK_I		beschreibung	16
D_EINLESEN	281	email	16
ERIC_FORTSCHRITTCALLBACK_I		land	16
D_SENDEN	282	name	16
ERIC_FORTSCHRITTCALLBACK_I		organisation	16
D_VALIDIEREN	281	ort	16
ERIC_FORTSCHRITTCALLBACK_I		version	17
D_VORBEREITEN	281	ericapi.h	50
ERIC_LOG_DEBUG	283	EricBearbeiteVorgang	58
ERIC_LOG_ERROR	283	EricBeende	64
ERIC_LOG_INFO	283	EricChangePassword	65
eric_log_level_t	283	EricCheckXML	66
ERIC_LOG_TRACE	283	EricCloseHandleToCertificate	67
ERIC_LOG_WARN	283	EricCreateKey	69
ERIC_PRUEFE_HINWEISE	282	EricCreateTH	71
ERIC_SENDE	282	EricCreateUUID	74
ERIC_VALIDIERE	282	EricDekodiereDaten	75
ERIC_VALIDIERE_OHNE_FREIGA		EricEinstellungAlleZuruecksetzen	77
BEDATUM	282	EricEinstellungLesen	78
EricFortschrittCallback	277	EricEinstellungSetzen	79
EricInstanzHandle	278	EricEinstellungZuruecksetzen	80
EricLogCallback	278	EricEntladePlugins	81
EricPdfCallback	279	EricFormatEWaz	82
EricRueckgabepufferHandle	280	EricFormatStNr	83
EricTransferHandle	281	EricGetAuswahlListen	84
EricZertifikatHandle	281	EricGetErrorMessageFromXMLAns	86
ERIC_VALIDIERE		wer	86
eric_types.h	282	EricGetHandleToCertificate	88
ERIC_VALIDIERE_OHNE_FREIGAB			
EDATUM			
eric_types.h	282		

EricGetPinStatus	93	ericapiExport.h	285
EricGetPublicKey	95	ericapiExport.h	284
EricHoleFehlerText	96	ERICAPI_IMPORT	285
EricHoleFinanzaemter	97	EricBearbeiteVorgang	
EricHoleFinanzamtLandNummern	98	ericapi.h	58
EricHoleFinanzamtsdaten	99	EricBeende	
EricHoleTestfinanzaemter	100	ericapi.h	64
EricHoleZertifikatEigenschaften	101	EricChangePassword	
EricHoleZertifikatFingerabdruck	103	ericapi.h	65
EricInitialisiere	104	EricCheckXML	
EricMakeElsterEWAZ	105	ericapi.h	66
EricMakeElsterStnr	106	EricCloseHandleToCertificate	
EricPruefeBIC	107	ericapi.h	67
EricPruefeBuFaNummer	108	EricCreateKey	
EricPruefeEWAZ	109	ericapi.h	69
EricPruefeIBAN	110	EricCreateTH	
EricPruefeIdentifikationsMerkmal	111	ericapi.h	71
EricPruefeSteuernummer	112	EricCreateUUID	
EricPruefeWIdNr	113	ericapi.h	74
EricPruefeZertifikatPin	114	ericdef.h	286
EricRegistriereFortschrittCallback	116	ERIC_MAX_LAENGE_FUSSTEXT	287
EricRegistriereGlobalenFortschrittCallback	117	ERIC_TESTMERKER_CLEARINGS	287
EricRegistriereLogCallback	118	ERIC_TESTMERKER_ECC	287
EricRueckgabepufferErzeugen	119	EricDekodiereDaten	
EricRueckgabepufferFreigeben	120	ericapi.h	75
EricRueckgabepufferInhalt	121	EricEinstellungAlleZuruecksetzen	
EricRueckgabepufferLaenge	122	ericapi.h	77
EricSystemCheck	123	EricEinstellungLesen	
EricVersion	124	ericapi.h	78
EricZertifikatOeffnenAusBytes	125	EricEinstellungSetzen	
ERICAPI_IMPORT		ericapi.h	79
		EricEinstellungZuruecksetzen	
		ericapi.h	80
		EricEntladePlugins	
		ericapi.h	81

EricFormatEWaz	EricMakeElsterEWaz
ericapi.h 82	ericapi.h 105
EricFormatStNr	EricMakeElsterStnr
ericapi.h 83	ericapi.h 106
EricFortschrittCallback	ericmtapi.h 127
eric_types.h 277	EricMtBearbeiteVorgang 135
EricGetAuswahlListen	EricMtChangePassword 141
ericapi.h 84	EricMtCheckXML 143
EricGetErrormessagesFromXMLAnswer	EricMtCloseHandleToCertificate 144
ericapi.h 86	EricMtCreateKey 146
EricGetHandleToCertificate	EricMtCreateTH 148
ericapi.h 88	EricMtCreateUUID 151
EricGetPinStatus	EricMtDekodiereDaten 152
ericapi.h 93	EricMtEinstellungAlleZuruecksetzen 154
EricGetPublicKey	EricMtEinstellungLesen 155
ericapi.h 95	EricMtEinstellungSetzen 156
EricHoleFehlerText	EricMtEinstellungZuruecksetzen 157
ericapi.h 96	EricMtEntladePlugins 158
EricHoleFinanzaemter	EricMtFormatEWaz 159
ericapi.h 97	EricMtFormatStNr 160
EricHoleFinanzamtLandNummern	EricMtGetAuswahlListen 161
ericapi.h 98	EricMtGetErrormessagesFromXML Answer 163
EricHoleFinanzamtsdaten	EricMtGetHandleToCertificate 165
ericapi.h 99	EricMtGetPinStatus 170
EricHoleTestfinanzaemter	EricMtGetPublicKey 172
ericapi.h 100	EricMtHoleFehlerText 173
EricHoleZertifikatEigenschaften	EricMtHoleFinanzaemter 174
ericapi.h 101	EricMtHoleFinanzamtLandNummern 176
EricHoleZertifikatFingerabdruck	EricMtHoleFinanzamtsdaten 177
ericapi.h 103	EricMtHoleTestfinanzaemter 178
EricInitialisiere	EricMtHoleZertifikatEigenschaften 179
ericapi.h 104	
EricInstanzHandle	
eric_types.h 278	
EricLogCallback	
eric_types.h 278	

EricMtHoleZertifikatFingerabdruck 181	ericmtapi.h 144
EricMtInstanzErzeugen 183	EricMtCreateKey ericmtapi.h 146
EricMtInstanzFreigeben 184	EricMtCreateTH ericmtapi.h 148
EricMtMakeElsterEWAZ 185	EricMtCreateUUID ericmtapi.h 151
EricMtMakeElsterStnr 186	EricMtDekodiereDaten ericmtapi.h 152
EricMtPruefeBIC 187	EricMtEinstellungAlleZuruecksetzen ericmtapi.h 154
EricMtPruefeBuFaNummer 188	EricMtEinstellungLesen ericmtapi.h 155
EricMtPruefeEWAZ 189	EricMtEinstellungSetzen ericmtapi.h 156
EricMtPruefelBAN 190	EricMtEinstellungZuruecksetzen ericmtapi.h 157
EricMtPruefeldentifikationsMerkmal 191	EricMtEntladePlugins ericmtapi.h 158
EricMtPruefeSteuernummer 192	EricMtFormatEWAZ ericmtapi.h 159
EricMtPruefeWIdNr 193	EricMtFormatStNr ericmtapi.h 160
EricMtPruefeZertifikatPin 194	EricMtGetAuswahlListen ericmtapi.h 161
EricMtRegistriereFortschrittCallback 196	EricMtGetErrorMessageFromXMLAn swer ericmtapi.h 163
EricMtRegistriereGlobalenFortschritt Callback 197	EricMtGetHandleToCertificate ericmtapi.h 165
EricMtRegistriereLogCallback 198	EricMtGetPinStatus ericmtapi.h 170
EricMtRueckgabepufferErzeugen 200	EricMtGetPublicKey ericmtapi.h 172
EricMtRueckgabepufferFreigeben 201	EricMtHoleFehlerText ericmtapi.h 173
EricMtRueckgabepufferInhalt 202	EricMtHoleFinanzaemter
EricMtRueckgabepufferLaenge 203	
EricMtSystemCheck 204	
EricMtVersion 205	
EricMtZertifikatOeffnenAusBytes 207	
EricMtBearbeiteVorgang ericmtapi.h 135	
EricMtChangePassword ericmtapi.h 141	
EricMtCheckXML ericmtapi.h 143	
EricMtCloseHandleToCertificate	

ericmtapi.h 174	EricMtRegistriereGlobalenFortschrittC allback ericmtapi.h 197
EricMtHoleFinanzamtLandNummern ericmtapi.h 176	EricMtRegistriereLogCallback ericmtapi.h 198
EricMtHoleFinanzamtsdaten ericmtapi.h 177	EricMtRueckgabepufferErzeugen ericmtapi.h 200
EricMtHoleTestfinanzaemter ericmtapi.h 178	EricMtRueckgabepufferFreigeben ericmtapi.h 201
EricMtHoleZertifikatEigenschaften ericmtapi.h 179	EricMtRueckgabepufferInhalt ericmtapi.h 202
EricMtHoleZertifikatFingerabdruck ericmtapi.h 181	EricMtRueckgabepufferLaenge ericmtapi.h 203
EricMtInstanzErzeugen ericmtapi.h 183	EricMtSystemCheck ericmtapi.h 204
EricMtInstanzFreigeben ericmtapi.h 184	EricMtVersion ericmtapi.h 205
EricMtMakeElsterEWaz ericmtapi.h 185	EricMtZertifikatOeffnenAusBytes ericmtapi.h 207
EricMtMakeElsterStrn ericmtapi.h 186	EricPdfCallback eric_types.h 279
EricMtPruefeBIC ericmtapi.h 187	EricPruefeBIC ericapi.h 107
EricMtPruefeBuFaNummer ericmtapi.h 188	EricPruefeBuFaNummer ericapi.h 108
EricMtPruefeEWaz ericmtapi.h 189	EricPruefeEWaz ericapi.h 109
EricMtPruefelBAN ericmtapi.h 190	EricPruefelBAN ericapi.h 110
EricMtPruefeldentifikationsMerkmal ericmtapi.h 191	EricPruefeldentifikationsMerkmal ericapi.h 111
EricMtPruefeSteuernummer ericmtapi.h 192	EricPruefeSteuernummer ericapi.h 112
EricMtPruefeWldNr ericmtapi.h 193	EricPruefeWldNr ericapi.h 113
EricMtPruefeZertifikatPin ericmtapi.h 194	EricPruefeZertifikatPin ericapi.h 114
EricMtRegistriereFortschrittCallback ericmtapi.h 196	

EricRegistriereFortschrittCallback	ERIC_MINOR_VERSION 209
ericapi.h 116	ERIC_PATCH_VERSION 209
EricRegistriereGlobalenFortschrittCall back	EricZertifikatHandle
ericapi.h 117	eric_types.h 281
EricRegistriereLogCallback	EricZertifikatOeffnenAusBytes
ericapi.h 118	ericapi.h 125
EricRueckgabepufferErzeugen	ETKAPI_DECL
ericapi.h 119	erictoolkit.h 211
EricRueckgabepufferFreigeben	EtkHoleDateiVersion
ericapi.h 120	erictoolkit.h 212
EricRueckgabepufferHandle	EtkHoleProduktVersion
eric_types.h 280	erictoolkit.h 213
EricRueckgabepufferInhalt	EtkPruefeBIC
ericapi.h 121	erictoolkit.h 214
EricRueckgabepufferLaenge	EtkPruefeBuFaNummer
ericapi.h 122	erictoolkit.h 215
EricSystemCheck	EtkPruefeEWaz
ericapi.h 123	erictoolkit.h 216
erictoolkit.h 210	EtkPruefeIBAN
ETKAPI_DECL 211	erictoolkit.h 217
EtkHoleDateiVersion 212	EtkPruefeldentifikationsMerkmal
EtkHoleProduktVersion 213	erictoolkit.h 218
EtkPruefeBIC 214	EtkPruefeSteuernummer
EtkPruefeBuFaNummer 215	erictoolkit.h 219
EtkPruefeEWaz 216	EtkPruefeWIdNr
EtkPruefeIBAN 217	erictoolkit.h 220
EtkPruefeldentifikationsMerkmal 218	fussText
EtkPruefeSteuernummer 219	eric_druck_parameter_t 10
EtkPruefeWIdNr 220	land
EricTransferHandle	eric_zertifikat_parameter_t 16
eric_types.h 281	name
EricVersion	eric_zertifikat_parameter_t 16
ericapi.h 124	organisation
ericversion.h 209	eric_zertifikat_parameter_t 16
ERIC_MAJOR_VERSION 209	ort
	eric_zertifikat_parameter_t 16
	otto.h 221

OttoDatenAbholen	225	OTTO_EMPFANG_VORZEITIG_BEE NDET	
OttoEinstellungLesen	229	otto_statuscode.h	264
OttoEinstellungSetzen	230	OTTO_ENTSCHLUESSELN_FEHLGE SCHLAGEN	
OttoEmpfangBeenden	231	otto_statuscode.h	269
OttoEmpfangBeginnen	232	OTTO_ESIGNER_ASN1_NO_CONTE NT_DATA	
OttoEmpfangBeginnenAbholzertifika t	233	otto_statuscode.h	269
OttoEmpfangFortsetzen	235	OTTO_ESIGNER_ASN1_NO_ENVEL OPED_DATA	
OttoHoleFehlertext	236	otto_statuscode.h	269
OttoInstanzErzeugen	237	OTTO_ESIGNER_ASN1_READ_BUF FER_TOO_SMALL	
OttoInstanzFreigeben	239	otto_statuscode.h	268
OttoProxyKonfigurationSetzen	240	OTTO_ESIGNER_ASN1_READ_DAT A_INCOMPLETE	
OttoPruefsummeAktualisieren	241	otto_statuscode.h	268
OttoPruefsummeErzeugen	242	OTTO_ESIGNER_BUSY	
OttoPruefsummeFreigeben	243	otto_statuscode.h	265
OttoPruefsummeSignieren	244	OTTO_ESIGNER_DATA_NOT_INITIA LIZED	
OttoRueckgabepufferErzeugen	245	otto_statuscode.h	268
OttoRueckgabepufferFreigeben	246	OTTO_ESIGNER_DECRYPT	
OttoRueckgabepufferGroesse	247	otto_statuscode.h	265
OttoRueckgabepufferInhalt	248	OTTO_ESIGNER_ENCODE_ERROR	
OttoVersandAbschliessen	249	otto_statuscode.h	265
OttoVersandBeenden	250	OTTO_ESIGNER_ENCODE_UNKNO WN	
OttoVersandBeginnen	251	otto_statuscode.h	265
OttoVersandFortsetzen	252	OTTO_ESIGNER_ENCRYPT	
OttoVersion	253	otto_statuscode.h	265
OttoZertifikatOeffnen	254	OTTO_ESIGNER_ESICL_EXCEPTIO N	
OttoZertifikatOeffnenAusBytes	257	otto_statuscode.h	265
OttoZertifikatSchliessen	258	OTTO_ESIGNER_INKOMPATIBEL	
OTTO_DEKOMPRESSSION_FEHLGE SCHLAGEN		otto_statuscode.h	264
otto_statuscode.h	269	OTTO_ESIGNER_INVALID_HANDLE	
OTTO_EINSTELLUNG_UNBEKANNT			
otto_statuscode.h	265		
OTTO_EINSTELLUNG_WERT_UNG UELTIG			
otto_statuscode.h	265		

otto_statuscode.h 265	OTTO_ESIGNER_P12_NO_ENC_CER
OTTO_ESIGNER_LOAD_DLL	RT
otto_statuscode.h 265	otto_statuscode.h 267
OTTO_ESIGNER_MAX_SESSION	OTTO_ESIGNER_P12_NO_SIG_CER
otto_statuscode.h 265	T
OTTO_ESIGNER_NICHT_GELADEN	otto_statuscode.h 267
otto_statuscode.h 264	OTTO_ESIGNER_P12_READ
OTTO_ESIGNER_NO_SERVICE	otto_statuscode.h 267
otto_statuscode.h 266	OTTO_ESIGNER_P12_SIG_KEY
OTTO_ESIGNER_NO_SIG_ENC_KEY	otto_statuscode.h 267
Y	OTTO_ESIGNER_P7_DECODE
otto_statuscode.h 266	otto_statuscode.h 267
OTTO_ESIGNER_OUT_OF_MEM	OTTO_ESIGNER_P7_READ
otto_statuscode.h 266	otto_statuscode.h 267
OTTO_ESIGNER_P11_ENC_KEY	OTTO_ESIGNER_P7_RECIPIENT
otto_statuscode.h 266	otto_statuscode.h 267
OTTO_ESIGNER_P11_ENGINE_LOADED	OTTO_ESIGNER_PIN_LOCKED
otto_statuscode.h 266	otto_statuscode.h 267
OTTO_ESIGNER_P11_INIT_FAILED	OTTO_ESIGNER_PIN_WRONG
otto_statuscode.h 266	otto_statuscode.h 267
OTTO_ESIGNER_P11_NO_ENC_CER	OTTO_ESIGNER_PSE_PATH
RT	otto_statuscode.h 267
otto_statuscode.h 266	OTTO_ESIGNER_SC_ENC_KEY
OTTO_ESIGNER_P11_NO_SIG_CER	otto_statuscode.h 267
T	OTTO_ESIGNER_SC_INIT_FAILED
otto_statuscode.h 266	otto_statuscode.h 267
OTTO_ESIGNER_P11_SIG_KEY	OTTO_ESIGNER_SC_NO_APPLET
otto_statuscode.h 266	otto_statuscode.h 268
OTTO_ESIGNER_P11_SLOT_EMPTY	OTTO_ESIGNER_SC_NO_ENC_CER
Y	T
otto_statuscode.h 266	otto_statuscode.h 268
OTTO_ESIGNER_P12_CREATE	OTTO_ESIGNER_SC_NO_SIG_CER
otto_statuscode.h 266	T
OTTO_ESIGNER_P12_DECODE	otto_statuscode.h 268
otto_statuscode.h 266	OTTO_ESIGNER_SC_SESSION
OTTO_ESIGNER_P12_ENC_KEY	otto_statuscode.h 268
otto_statuscode.h 267	OTTO_ESIGNER_SC_SIG_KEY
	otto_statuscode.h 268

OTTO_ESIGNER_SC_SLOT_EMPTY	OTTO_PROXY_PORT
otto_statuscode.h 268	otto_statuscode.h 264
OTTO_ESIGNER_TOKEN_TYPE_MISMATCH	OTTO_PROXY_URL
otto_statuscode.h 268	otto_statuscode.h 264
OTTO_ESIGNER_USER_CANCEL	OTTO_PRUEFSUMME_FINALISIERT
otto_statuscode.h 268	otto_statuscode.h 264
OTTO_ESIGNER_VERALTET	OTTO_SIGNIEREN_FEHLGESCHLAGEN
otto_statuscode.h 264	otto_statuscode.h 269
OTTO_ESIGNER_VERIFY_CERT_CHAIN	otto_statuscode.h 259
otto_statuscode.h 268	OTTO_DEKOMPRESSION_FEHLGESCHLAGEN 269
OTTO_FUNKTION_NICHT_UNTESTET	OTTO_EINSTELLUNG_UNBEKANNT 265
otto_statuscode.h 263	OTTO_EINSTELLUNG_WERT_UNGUELTIG 265
OTTO_INIDATEI_LESEFEHLER	OTTO_EMPFANG_VORZEITIG_BEENDET 264
otto_statuscode.h 269	OTTO_ENTSCHLUESSELN_FEHLGESCHLAGEN 269
OTTO_INSTANZ_UNTEROBJEKTE_NICHT_FREIGEgeben	OTTO_ESIGNER_ASN1_NO_CONTENT_DATA 269
otto_statuscode.h 263	OTTO_ESIGNER_ASN1_NO_ENVELOPED_DATA 269
OTTO_INSTANZEN_INKONSISTENT	OTTO_ESIGNER_ASN1_READ_BUFFER_TOO_SMALL 268
otto_statuscode.h 263	OTTO_ESIGNER_ASN1_READ_DATA_INCOMPLETE 268
OTTO_INTERNER_FEHLER	OTTO_ESIGNER_BUSY 265
otto_statuscode.h 261	OTTO_ESIGNER_DATA_NOT_INITIALIZED 268
OTTO_LOG_FEHLER	OTTO_ESIGNER_DECRYPT 265
otto_statuscode.h 263	OTTO_ESIGNER_ENCODE_ERROR 265
OTTO_MEHRFACHAUFRUFE_NICHT_UNTESTET	OTTO_ESIGNER_ENCODE_UNKNOWN 265
otto_statuscode.h 263	OTTO_ESIGNER_ENCRYPT 265
OTTO_NICHT_GENUEGEND_ARBEITSSPEICHER	OTTO_ESIGNER_ESICL_EXCEPTION 265
otto_statuscode.h 269	
OTTO_NPA_ZERTIFIKATFEHLER	
otto_statuscode.h 262	
OTTO_OK	
otto_statuscode.h 261	
OTTO_PROXY_AUTHSCHEMA	
otto_statuscode.h 264	

OTTO_ESIGNER_INKOMPATIBEL 264	OTTO_ESIGNER_P12_SIG_KEY 267
OTTO_ESIGNER_INVALID_HANDL E 265	OTTO_ESIGNER_P7_DECODE 267
OTTO_ESIGNER_LOAD_DLL 265	OTTO_ESIGNER_P7_READ 267
OTTO_ESIGNER_MAX_SESSION 265	OTTO_ESIGNER_P7_RECIPIENT 267
OTTO_ESIGNER_NICHT_GELADE N 264	OTTO_ESIGNER_PIN_LOCKED 267
OTTO_ESIGNER_NO_SERVICE 266	OTTO_ESIGNER_PIN_WRONG 267
OTTO_ESIGNER_NO_SIG_ENC_K EY 266	OTTO_ESIGNER_PSE_PATH 267
OTTO_ESIGNER_OUT_OF_MEM 266	OTTO_ESIGNER_SC_ENC_KEY 267
OTTO_ESIGNER_P11_ENC_KEY 266	OTTO_ESIGNER_SC_INIT_FAILE D 267
OTTO_ESIGNER_P11_ENGINE_L OADED 266	OTTO_ESIGNER_SC_NO_APPLET 268
OTTO_ESIGNER_P11_INIT_FAILE D 266	OTTO_ESIGNER_SC_NO_ENC_C ERT 268
OTTO_ESIGNER_P11_NO_ENC_C ERT 266	OTTO_ESIGNER_SC_NO_SIG_CE RT 268
OTTO_ESIGNER_P11_NO_SIG_C ERT 266	OTTO_ESIGNER_SC_SESSION 268
OTTO_ESIGNER_P11_SIG_KEY 266	OTTO_ESIGNER_SC_SIG_KEY 268
OTTO_ESIGNER_P11_SLOT_EMP TY 266	OTTO_ESIGNER_SC_SLOT_EMP TY 268
OTTO_ESIGNER_P12_CREATE 266	OTTO_ESIGNER_TOKEN_TYPE_ MISMATCH 268
OTTO_ESIGNER_P12_DECODE 266	OTTO_ESIGNER_USER_CANCEL 268
OTTO_ESIGNER_P12_ENC_KEY 267	OTTO_ESIGNER_VERALTET 264
OTTO_ESIGNER_P12_NO_ENC_C ERT 267	OTTO_ESIGNER_VERIFY_CERT_ CHAIN 268
OTTO_ESIGNER_P12_NO_SIG_C ERT 267	OTTO_FUNKTION_NICHT_UNTER STUETZT 263
OTTO_ESIGNER_P12_READ 267	OTTO_INIDATEI_LESEFEHLER 269

OTTO_INSTANZ_UNTEROBJEKTE _NICHT_FREIGEgeben 263	OTTO_TRANSFER_NOT_FOUND 262
OTTO_INSTANZEN_INKONSISTE NT 263	OTTO_TRANSFER_PROXYAUTH 262
OTTO_INTERNER_FEHLER 261	OTTO_TRANSFER_SERVER_FEH LER 262
OTTO_LOG_FEHLER 263	OTTO_TRANSFER_TIMEOUT 262
OTTO_MEHRFACHAUFRUFE_NIC HT_UNTERSTUETZT 263	OTTO_TRANSFER_UNAUTHORIZ ED 262
OTTO_NICHT_GENUEGEND_ARB EITSSPEICHER 269	OTTO_UNBEKANNTER_FEHLER 261
OTTO_NPA_ZERTIFIKATFEHLER 262	OTTO_UNGUELTIGE_HERSTELLE RID 264
OTTO_OK 261	OTTO_UNGUELTIGER_PARAMET ER 263
OTTO_PROXY_AUTHSCHEMA 264	OTTO_UNGUELTIGES_HANDLE 263
OTTO_PROXY_PORT 264	OTTO_VERSAND_ABGESCHLOS SEN 265
OTTO_PROXY_URL 264	OTTO_VERSAND_GERINGE_DAT ENMENGE 264
OTTO_PRUEFSUMME_FINALISIE RT 264	OTTO_VERSAND_ZU_GROSSE_D ATENMENGE 265
OTTO_SIGNIEREN_FEHLGESCHL AGEN 269	OTTO_ZERTIFIKAT_DEFEKT 269
OTTO_TRANSFER_CONNECTPR OXY 262	OTTO_ZERTIFIKAT_FINGERABDR UCK_FEHLER 269
OTTO_TRANSFER_CONNECTSE RVER 262	OTTO_ZERTIFIKAT_LESEFEHLER 269
OTTO_TRANSFER_DECODING 262	OTTO_ZERTIFIKAT_NICHT_ERKA NNT 264
OTTO_TRANSFER_EID_CLIENTF EHLER 263	OTTO_ZERTIFIKAT_PFAD_FALSC H 264
OTTO_TRANSFER_EID_KEINCLIE NT 263	OTTO_ZERTIFIKAT_PIN_FALSCH 264
OTTO_TRANSFER_EID_KEINKON TO 263	OttoStatusCode 261
OTTO_TRANSFER_EID_NPABLO CKIERT 263	OTTO_TRANSFER_CONNECTPROX Y
OTTO_TRANSFER_EID_ZERTIFIK ATFEHLER 262	otto_statuscode.h 262
OTTO_TRANSFER_FEHLER 262	
OTTO_TRANSFER_INIT 262	

OTTO_TRANSFER_CONNECTSERVER	OTTOLOG_DEBUGMELDUNGEN 274
otto_statuscode.h 262	OTTOLOG_FEHLERMELDUNGEN 274
OTTO_TRANSFER_DECODING	OTTOLOG_INFORMATIONEN 274
otto_statuscode.h 262	OTTOLOG_WARNUNGEN 274
OTTO_TRANSFER_EID_CLIENTFEHLER	OttoLogCallback 272
otto_statuscode.h 263	OttoLogEbene 273
OTTO_TRANSFER_EID_KEINCLIENT	OttoPruefsummeHandle 272
otto_statuscode.h 263	OttoRueckgabepufferHandle 272
OTTO_TRANSFER_EID_KEINKONTO	OttoVersandHandle 273
otto_statuscode.h 263	OttoZertifikatHandle 273
OTTO_TRANSFER_EID_NPABLOCKIERT	OTTO_UNBEKANNTER_FEHLER otto_statuscode.h 261
otto_statuscode.h 263	OTTO_UNGUELTIGE_HERSTELLERID otto_statuscode.h 264
OTTO_TRANSFER_EID_ZERTIFIKATFEHLER	OTTO_UNGUELTIGER_PARAMETER otto_statuscode.h 263
otto_statuscode.h 262	OTTO_UNGUELTIGES_HANDLE otto_statuscode.h 263
OTTO_TRANSFER_FEHLER	OTTO_VERSAND_ABGESCHLOSSEN otto_statuscode.h 265
otto_statuscode.h 262	OTTO_VERSAND_GERINGE_DATENMENGE otto_statuscode.h 264
OTTO_TRANSFER_INIT	OTTO_VERSAND_ZU_GROSSE_DATENMENGE otto_statuscode.h 265
otto_statuscode.h 262	OTTO_ZERTIFIKAT_DEFECT otto_statuscode.h 269
OTTO_TRANSFER_NOT_FOUND	OTTO_ZERTIFIKAT_FINGERABDRUCK_FEHLER otto_statuscode.h 269
otto_statuscode.h 262	OTTO_ZERTIFIKAT_LESEFEHLER
OTTO_TRANSFER_PROXYAUTH	
otto_statuscode.h 262	
OTTO_TRANSFER_SERVER_FEHLER	
otto_statuscode.h 262	
OTTO_TRANSFER_TIMEOUT	
otto_statuscode.h 262	
OTTO_TRANSFER_UNAUTHORIZED	
otto_statuscode.h 262	
otto_types.h 270	
OttoEmpfangHandle 271	
OttoInstanzHandle 271	

otto_statuscode.h 269	otto_types.h 274
OTTO_ZERTIFIKAT_NICHT_ERKANNT	OTTOLOG_WARNUNGEN
otto_statuscode.h 264	otto_types.h 274
OTTO_ZERTIFIKAT_PFAD_FALSCH	OttoLogCallback
otto_statuscode.h 264	otto_types.h 272
OTTO_ZERTIFIKAT_PIN_FALSCH	OttoLogEbene
otto_statuscode.h 264	otto_types.h 273
OttoDatenAbholen	OttoProxyKonfiguration 18
otto.h 225	authentifizierungsmethode 19
OttoEinstellungLesen	benutzerName 19
otto.h 229	benutzerPasswort 19
OttoEinstellungSetzen	url 19
otto.h 230	version 20
OttoEmpfangBeenden	OttoProxyKonfigurationSetzen
otto.h 231	otto.h 240
OttoEmpfangBeginnen	OttoPruefsummeAktualisieren
otto.h 232	otto.h 241
OttoEmpfangBeginnenAbholzertifikat	OttoPruefsummeErzeugen
otto.h 233	otto.h 242
OttoEmpfangFortsetzen	OttoPruefsummeFreigeben
otto.h 235	otto.h 243
OttoEmpfangHandle	OttoPruefsummeHandle
otto_types.h 271	otto_types.h 272
OttoHoleFehlertext	OttoPruefsummeSignieren
otto.h 236	otto.h 244
OttoInstanzErzeugen	OttoRueckgabepufferErzeugen
otto.h 237	otto.h 245
OttoInstanzFreigeben	OttoRueckgabepufferFreigeben
otto.h 239	otto.h 246
OttoInstanzHandle	OttoRueckgabepufferGroesse
otto_types.h 271	otto.h 247
OTTOLOG_DEBUGMELDUNGEN	OttoRueckgabepufferHandle
otto_types.h 274	otto_types.h 272
OTTOLOG_FEHLERMELDUNGEN	OttoRueckgabepufferInhalt
otto_types.h 274	otto.h 248
OTTOLOG_INFORMATIONEN	OttoStatusCode
	otto_statuscode.h 261

OttoVersandAbschliessen	eric_druck_parameter_t 10
otto.h 249	pdfName
OttoVersandBeenden	eric_druck_parameter_t 10
otto.h 250	pin
OttoVersandBeginnen	eric_verschluesselungs_parameter_t 13
otto.h 251	platform.h 288
OttoVersandFortsetzen	uint32_t 288
otto.h 252	Start 4
OttoVersandHandle	uint32_t
otto_types.h 273	platform.h 288
OttoVersion	url
otto.h 253	OttoProxyKonfiguration 19
OttoZertifikatHandle	version
otto_types.h 273	eric_druck_parameter_t 11
OttoZertifikatOeffnen	eric_verschluesselungs_parameter_t 13
otto.h 254	eric_zertifikat_parameter_t 17
OttoZertifikatOeffnenAusBytes	OttoProxyKonfiguration 20
otto.h 257	vorschau
OttoZertifikatSchliessen	eric_druck_parameter_t 11
otto.h 258	zertifikatHandle
pdfCallback	eric_verschluesselungs_parameter_t 13
eric_druck_parameter_t 10	
pdfCallbackBenutzerdaten	