

ERiC API-Referenz

Version 41.6.2.0

Inhaltsverzeichnis

Start	2
Suchfunktion	2
Dokumentation	2
Encoding und Zeichensatz	2
Datenstruktur-Verzeichnis	3
Datei-Verzeichnis	4
Datenstruktur-Dokumentation	5
eric_druck_parameter_t	5
eric_verschlüsselungs_parameter_t	8
eric_zertifikat_parameter_t	10
OttoProxyKonfiguration	13
Datei-Dokumentation	15
eric_fehlercodes.h	15
ericapi.h	36
Inhalt des Rückgabepuffers und des Serverantwortpuffers	42
Erfolgsfall	42
Hinweise	42
Plausibilitätsfehler	43
Fehler in der Serverantwort	43
Sonstige Fehler	43
Fortschrittcallbacks	44
ericmtapi.h	75
Inhalt des Rückgabepuffers und des Serverantwortpuffers	81
Erfolgsfall	81
Hinweise	82
Plausibilitätsfehler	82
Fehler in der Serverantwort	83
Sonstige Fehler	83
Fortschrittcallbacks	83
ericversion.h	116
erictoolkit.h	117
otto.h	122
otto_statuscode.h	138
otto_types.h	146
eric_types.h	150
ericapiExport.h	157
ericdef.h	158
platform.h	160
Index	162

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 HTML-Seiten der API-Referenz enthalten ein Suchfeld. Voraussetzung ist ein Browser mit aktiviertem JavaScript. Es kann nur nach Symbolen gesucht werden. Eine Volltextsuche ist nicht möglich.

Dokumentation

Das Dokumentationspaket beinhaltet das *ERiC-Entwicklerhandbuch.pdf*, *ERiC-Tutorial.pdf*, *ERiC-Releasenotes.pdf*, *Datenartversionmatrix.xml*, diese API-Referenz sowie die Dokumentation aller Feldkennungen, Plausibilitätsprüfungen, Schemata und Schnittstellenbeschreibungen.

Im Entwicklerhandbuch finden Sie sowohl allgemeine Zusatzinformationen als auch spezielle Hinweise zum Gebrauch der Bibliotheken, Datensätze, Datensatzformate und Werte.

Das Tutorial illustriert detailliert die Softwareentwicklung mit ERiC an den mitgelieferten Beispielen *ericdemo* und *ottodemo*.

Die Release Notes enthalten die Änderungen der aktuell unterstützten ERiC Releases.

Die Datenartversionmatrix enthält eine Übersicht der *datenartVersion*en, die ERiC unterstützt. Einige API-Funktionen verwenden die *datenartVersion* als Parameter, weitere Informationen siehe *ERiC-Entwicklerhandbuch.pdf*, Kapitel "datenartVersion – Definition und Verwendung."

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*, Kapitel "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)	5
<u>eric_verschluesselungs_parameter_t</u> (Für die Signatur oder Authentifizierung benötigte Informationen)	8
<u>eric_zertifikat_parameter_t</u> (Struktur mit Informationen zur Erzeugung von Zertifikaten mit <u>EricCreateKey</u>)	10
<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)	13

Datei-Verzeichnis

Auflistung der Dateien

Hier folgt die Aufzählung aller Dateien mit einer Kurzbeschreibung:

<u>eric fehlercodes.h</u> (Auflistung der ERIC API-Fehlercodes)	15
<u>ericapi.h</u> (Deklaration der ERiC API-Funktionen für die Singlethreading-API)	36
<u>ericmtapi.h</u> (Deklaration der ERiC API-Funktionen für die Multithreading-API)	75
<u>ericversion.h</u> (Bereitstellung der ERiC API-Version über C-Präprozessor Makros. Die ERiC API-Version entspricht nicht unbedingt der Version des Setup-Pakets)	116
<u>erictoolkit.h</u> (Bereitstellung von Prüffunktionen ohne Abhängigkeit zu anderen ERiC Bibliotheken)	117
<u>otto.h</u> (Deklaration der Otto-Funktionen)	122
<u>otto_statuscode.h</u> (Auflistung der Otto-Statuscodes)	138
<u>otto_types.h</u> (Definition von Datenstrukturen und Datentypen)	146
<u>eric_types.h</u> (Definition von Datenstrukturen und Datentypen)	150
<u>ericapiExport.h</u> (Attribute für dynamische Bibliotheken)	157
<u>ericdef.h</u> (Konstanten und Definitionen für Übergabeparameter)	158
<u>platform.h</u> (Konstanten für verschiedene Betriebssysteme)	160

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 allokieren 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?

- `duplexDruck = 1`: Die geraden Seiten werden für einen Heftrand zum Lochen nach links eingerückt, 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 Entwicklerhandbuch Kapitel "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 Dateieindung 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?

- 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 char * 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 char* 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:

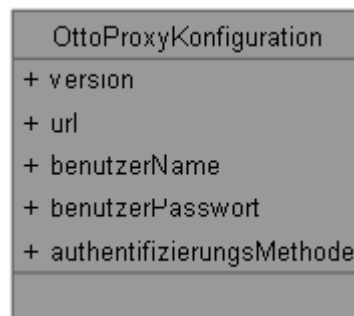
[eric_types.h](#)

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_FALSCHES_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_ANGEGEBEN](#) = 610001206, [ERIC_GLOBAL_SEND_FLAG_MEHREALS_EINES](#) = 610001209, [ERIC_GLOBAL_UNGUELTIGE_FLAG_KOMBINATION](#) = 610001218,

[ERIC GLOBAL ERSTE SEITE DRUCK NICHT UNTERSTUETZT](#) = 610001220,
[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 UNTERSTUETZT](#) =
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 APPLET](#) =
 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 ABBRUCH DRUCKVORBEREITUNG](#) = 610501009,
[ERIC PRINT ABBRUCH 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	[610001001] Verarbeitung fehlerhaft, keine genaueren Informationen

_UNKNOWN	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_NICHT_ERLAUBT	[610001018] Die aufgerufene Funktion ist nicht erlaubt.
ERIC_GLOBAL_ECHTFALL_NICHT_ERLAUBT	[610001019] Für dieses Verfahren/diese Datenart/diese Test-Hersteller-ID/diese ERiC-Einstellungen sind Echtfälle nicht erlaubt.
ERIC_GLOBAL_NO_VERSAND_IN_BETA_VERSION	[610001020] Der Versand von Echtfällen (= Fällen ohne gesetzten Testmarker) ist mit einer BETA-Version nicht möglich.
ERIC_GLOBAL_TESTMERKER_UNGUELTIG	[610001025] Der übergebene Testmarker ist für das angegebene Verfahren nicht zulässig.
ERIC_GLOBAL	[610001026] Der zu versendende Datensatz ist zu groß.

_DATENSATZ_ZU_GROSS	
ERIC_GLOBAL_VERSCHLUESSELUNGS_PARAMETER_NICHT_ERLAUBT	[610001027] Der Verschlüsselungsparameter darf nur bei authentifiziertem Versand angegeben werden.
ERIC_GLOBAL_NUR_PORTAL_ZERTIFIKAT_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_TEXTPUFFER_GROESSE_FIX	[610001031] Die Größe des Textpuffers kann nicht verändert werden.
ERIC_GLOBAL_INTERNER_FEHLER	[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_STEUERNUMMER_UNGUELTIG	[610001034] Ungültige Steuernummer.
ERIC_GLOBAL_STEUERNUMMER_FALSCH_E_LAENGE	[610001035] Ungültige Steuernummer: Es werden 13 Stellen erwartet.
ERIC_GLOBAL_STEUERNUMMER_NICHT_NUMERISCH	[610001036] Ungültige Steuernummer: Es werden nur Ziffern erwartet.
ERIC_GLOBAL_LANDESNUMMER_UNBEKANNT	[610001037] Ungültige Landesnummer.
ERIC_GLOBAL_BUFANR_UNBEKANNT	[610001038] Ungültige Bundesfinanzamtsnummer.
ERIC_GLOBAL_LANDESNUMMER_BUFANR	[610001039] Ungültige Bundesfinanzamtsnummer.
ERIC_GLOBAL_PUFFER_ZUGRIFFSKONFLIKT	[610001040] Ein Puffer-Handle wurde mehrfach übergeben.

ERIC_GLOBAL_PUFFER_UEBERLAUF	[610001041] Es wurde versucht, einen Puffer über die maximal mögliche Länge hinaus zu beschreiben.
ERIC_GLOBAL_DATENARTVERSION_UNBEKANNT	[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_DATENARTVERSION_XML_INKONSISTENT	[610001044] Die übergebene Datenartversion passt nicht zum Eingangs-XML. Details stehen ggf. im Logfile (eric.log).
ERIC_GLOBAL_COMMONDATA_NICHT_VERFUEGBAR	[610001045] Das Plugin 'commonData' konnte nicht geladen werden oder bietet einen benötigten Service nicht an. Details stehen im Logfile (eric.log).
ERIC_GLOBAL_LOG_EXCEPTION	[610001046] Beim Schreiben in die Protokolldatei ist eine Ausnahme aufgetreten.
ERIC_GLOBAL_TRANSPORTSCHLUESSEL_NICHT_ERLAUBT	[610001047] Für diese Datenart darf im TransferHeader kein TransportSchlüssel angegeben werden.
ERIC_GLOBAL_OEFFENTLICHER_SCHLUESSEL_UNGUELTIG	[610001048] Der übergebene öffentliche Schlüssel kann nicht eingelesen werden.
ERIC_GLOBAL_TRANSPORTSCHLUESSEL_TYP_FALSCH	[610001049] Der Typ des im TransferHeader angegebenen Transportschlüssels ist für diese Datenart nicht erlaubt.
ERIC_GLOBAL_PUFFER_UNGLEICHER_INSTANZ	[610001050] Das übergebene Puffer-Handle wurde nicht mit der vorliegenden Instanz erzeugt.
ERIC_GLOBAL_VORSATZ_UNGUELTIG	[610001051] Das Element "Vorsatz" enthält ungültige Werte, Details stehen im Logfile (eric.log).
ERIC_GLOBAL_DATEIZUGRIFF_VERWEIGERT	[610001053] Auf eine Datei konnte nicht in gewünschter Weise zugegriffen werden. Details stehen im Logfile (eric.log).
ERIC_GLOBAL_UNGUELTIGE_INSTANZ	[610001080] Die übergebene Instanz ist gleich NULL oder bereits freigegeben worden.
ERIC_GLOBAL_NICHT_INITIALISIERT	[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_P ARAMETER_E RROR	[610001102] Unbekannter Parameterfehler.
ERIC_GLOBAL _CHECK_COR RUPTED_NDS	[610001108] Defekter Nutzdatensatz.
ERIC_GLOBAL _VERSCHLUES SELUNGS_PAR AMETER_NIC HT_ANGEGB EN	[610001206] Verschlüsselter/authentifizierter Versand gewünscht, aber keine notwendigen Verschlüsselungsparameter angegeben.
ERIC_GLOBAL _SEND_FLAG_ MEHR_ALS_EI NES	[610001209] Es ist mehr als ein Versandflag angegeben.
ERIC_GLOBAL _UNGUELTI GE_FLAG_KOMBI NATION	[610001218] Die übergebene Kombination von Bearbeitungsflags ist nicht erlaubt.
ERIC_GLOBAL _ERSTE_SEITE _DRUCK_NICH T_UNTERSTUE TZT	[610001220] Der Erste-Seite-Druck wird nicht mehr unterstützt..
ERIC_GLOBAL _UNGUELTI GE_R_PARAMETE R	[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 _UNGUELTI GE_PARAMETER _VERSION	[610001226] Die Version eines der angegebenen Parameter ist ungültig.

ERIC_GLOBAL_TRANSFERHANDLE	[610001227] Für das Verfahren Datenabholung wurde ein illegales Transferhandle angegeben.
ERIC_GLOBAL_PLUGININITIALISIERUNG	[610001228] Die Initialisierung eines Plugins ist fehlgeschlagen.
ERIC_GLOBAL_INKOMPATIBLE_VERSIONEN	[610001229] Die Versionen der im Logfile genannten ERiC-Dateien sind nicht kompatibel. (Siehe eric.log.)
ERIC_GLOBAL_VERSCHLUESSELUNGSVERFAHREN_NICHT_UNTERSTUETZT	[610001230] Das im XML-Element "<Verschlüsselung>" angegebene Verschlüsselungsverfahren wird vom ERiC nicht unterstützt.
ERIC_GLOBAL_MEHRFACHAUFRUFE_NICHT_UNTERSTUETZT	[610001231] Der Aufruf eine API-Funktion des ERiCs darf erst dann erfolgen, wenn ein vorheriger Aufruf zurückgekehrt ist.
ERIC_GLOBAL_UTILITY_COUNTRY_NOT_SUPPORTED	[610001404] Das Bundesland/Finanzamt mit der angegebenen Nummer nimmt bei der angegebenen Steuerart am ELSTER-Verfahren nicht teil.
ERIC_GLOBAL_IBAN_FORMATLER_FEHLER	[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_FORMATLER_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_ZULASSUNGS	[610001519] Die angegebene Zulassungsnummer entspricht nicht den Längenvorgaben. Es sind maximal 6 Stellen erlaubt.

NUMMER_ZU_LANG	
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_NCESSARY	[610001851] Update des ERiC erforderlich. Starten Sie nun das Update.
ERIC_GLOBAL_EINSTELLUNG_NAME_UNGUELTIG	[610001860] Ungültiger Name für Einstellung.
ERIC_GLOBAL_EINSTELLUNG_WERT_UNGUELTIG	[610001861] Ungültiger Wert für Einstellung.
ERIC_GLOBAL_ERR_DEKODIEREN	[610001862] Fehler beim Dekodieren.
ERIC_GLOBAL_FUNKTION_NICHT_UNTERSTUETZT	[610001863] Die aufgerufene Funktion wird nicht unterstützt.
ERIC_GLOBAL_NUTZDATENTICKETS_NICHT_EINDEUTIG	[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_NUTZDATENHEADERVERSIONEN_UNEINHEITLICH	[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_BUNDESLÄNDER_UNEINHEITLICH	[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	[610001868] Fehler im übergebenen EDS-XML: Es wurden Fälle für unterschiedliche Jahre angegeben. Innerhalb einer Datenlieferung

_UNEINHEITLICH	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 "<Empfänger>" 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 "<TransportSchlüssel>" nicht gefunden werden.
ERIC_TRANSFER_ERR_ENDTRANSPORTSCHLUESSEL	[610101258] Im XML-String konnte der Text "</TransportSchlüssel>" nicht gefunden werden.
ERIC_TRANSFER_ERR_BEGINN_DATENGROESSE	[610101259] Im XML-String konnte der Text "<DatenGroesse>" nicht gefunden werden.
ERIC_TRANSFER_ERR_ENDDATENGROESSE	[610101260] Im XML-String konnte der Text "</DatenGroesse>" nicht gefunden werden.

ERIC_TRANSF ER_ERR_SEND	[610101271] Beim Datenaustausch ist ein Fehler aufgetreten.
ERIC_TRANSF ER_ERR_NOTE NCRYPTED	[610101274] Die Antwortdaten waren nicht PKCS#7-verschlüsselt.
ERIC_TRANSF ER_ERR_PROX YCONNECT	[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_NORE SPONSE	[610101279] Von der Clearingstelle konnte keine Antwort empfangen werden.
ERIC_TRANSF ER_ERR_PROX YAUTH	[610101280] Der Proxyserver erwartet Anmeldedaten.
ERIC_TRANSF ER_ERR_SEND _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_PROX YPORT_INVAL ID	[610101284] Es wurde kein gültiger Port für den Proxy angegeben.
ERIC_TRANSF ER_ERR_OTHE R	[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_ENDS IGUSER	[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_DATE NTEILFEHLER	[610101297] Das XML-Element "<DatenTeil>" konnte nicht gelesen werden.

ERIC_TRANSF ER_EID_ZERTI FIKATFEHLER	[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_KEINC LIENT	[610101520] Der eID-Client ist nicht erreichbar. Wahrscheinlich wurde er nicht gestartet oder die übergebene lokale URL ist nicht korrekt.
ERIC_TRANSF ER_EID_CLIEN TFEHLER	[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_IDENT IFIKATIONAB 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_ME	[610201104] eSigner: Speicherzuordnungsfehler.

M	
ERIC_CRYPT_ E_PSE_PATH	[610201105] eSigner: Ungültiger PSE Pfad.
ERIC_CRYPT_ E_PIN_WRONG	[610201106] eSigner: Es wurde ein falsches Passwort bzw. eine falsche PIN angegeben.
ERIC_CRYPT_ E_PIN_LOCKED	[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	[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_ER ROR	[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_E MPTY	[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_EXCE PTION	[610201130] eSigner: Unbekannte Ausnahme aufgetreten.
ERIC_CRYPT_ E_ESIGNER_NI CHT_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_VERALTETE _ESIGNER_VE RSION	[610201142] eSigner: Die eSigner-Bibliothek liegt in einer veralteten Version vor
ERIC_CRYPT_ E_TOKEN_TYP E_MISMATCH	[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_ENGINE _LOADED	[610201148] eSigner: PKCS#11 Engine mit anderer Bibliothek belegt.

ERIC_CRYPT_ E_USER_CANCEL	[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_UNTERSTUETZTES_PSE_FORMAT	[610201203] Das Format der PSE wird nicht unterstützt.
ERIC_CRYPT_ PIN_BENOTIGT	[610201205] Für die ausgewählte Operation muss ein Passwort bzw. eine PIN angegeben werden.
ERIC_CRYPT_ PIN_STAERKE_NICHT_AUSREICHEND	[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_ ZERTIFIKATSPfad_KEIN_VERZEICHNIS	[610201209] Der angegebene Zertifikatspfad ist kein Verzeichnis.
ERIC_CRYPT_ ZERTIFIKATS_DATEI_EXISTIERT_BEREITS	[610201210] Im angegebenen Verzeichnis existiert bereits ein Bestandteil eines ERiC-Zertifikats.
ERIC_CRYPT_ PIN_ENTHAELT_UNGUELTIGE_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_NICHT_UNTERSTUETZT	[610201214] Die aufgerufene Funktion unterstützt den neuen Personalausweis (nPA) und den elektronischen Aufenthaltstitel (eAT) nicht.
ERIC_CRYPT_ E_SC_SLOT_EMPTY	[610201215] Es ist keine Karte/kein Stick eingesteckt.
ERIC_CRYPT_ E_SC_NO_APPLET	[610201216] Kein unterstütztes Applet gefunden.
ERIC_CRYPT_ E_SC_SESSION	[610201217] Fehler in der Kartensession.

ERIC_CRYPT_ E_P11_NO_SIG _CERT	[610201218] P11 Signaturzertifikat fehlt.
ERIC_CRYPT_ E_P11_INIT_FA ILED	[610201219] P11 Der initiale Tokenzugriff ist fehlgeschlagen.
ERIC_CRYPT_ E_P11_NO_EN C_CERT	[610201220] P11 Verschlüsselungszertifikat fehlt.
ERIC_CRYPT_ E_P12_NO_SIG _CERT	[610201221] P12 Signaturzertifikat fehlt.
ERIC_CRYPT_ E_P12_NO_EN C_CERT	[610201222] P12 Verschlüsselungszertifikat fehlt.
ERIC_CRYPT_ E_SC_ENC_KE Y	[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_ENC _CERT	[610201225] PC/SC Verschlüsselungszertifikat fehlt.
ERIC_CRYPT_ E_SC_INIT_FAI LED	[610201226] PC/SC Der initiale Tokenzugriff ist fehlgeschlagen.
ERIC_CRYPT_ E_SC_SIG_KEY	[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_READ _BUFFER_TOO _SMALL	[610201229] Der Lesepuffer zum Dekodieren der ASN.1-Struktur ist zu klein
ERIC_CRYPT_ E_ASN1_READ _DATA_INCO Mplete	[610201230] Die Daten der ASN.1-Struktur sind unvollständig
ERIC_CRYPT_ E_ASN1_NO_E NVELOPED_D ATA	[610201231] Die ASN.1-Struktur enthält kein EnvelopedData
ERIC_CRYPT_ E_ASN1_NO_C ONTENT_DAT A	[610201232] Die ASN.1-Struktur enthält keine Daten

ERIC_IO_FEHLER	[610301001] Verarbeitung fehlerhaft, keine genaueren Informationen vorhanden.
ERIC_IO_DATEI_INKORREKT	[610301005] Der Dateiaufbau ist nicht korrekt.
ERIC_IO_PARSE_FEHLER	[610301006] Fehler beim Parsen der Eingabedaten. Details stehen im Logfile (eric.log).
ERIC_IO_NDS_GENERIERUNG_FEHLGESCHLAGEN	[610301007] Die Generierung des Nutzdatensatzes ist fehlgeschlagen.
ERIC_IO_MASTERDATENSERVICE_NICHT_VERFUEGBAR	[610301010] Interner Fehler, der Masterdatenservice ist nicht verfügbar.
ERIC_IO_STEUERZEICHEN_IM_NDS	[610301014] Es wurden ungültige Steuerzeichen im Nutzdatensatz gefunden.
ERIC_IO_VERSIONSINFORMATIONEN_NICHT_GEFUNDEN	[610301031] Die Versionsinformationen der ERiC-Bibliotheken konnten nicht ausgelesen werden.
ERIC_IO_FALSCHE_VERFAHREN	[610301104] Der Wert im Transferheader-Element "Verfahren" wird vom verwendeten Reader nicht unterstützt.
ERIC_IO_READER_MEHRFACHE_STEUERFÄLLE	[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_FALSCHE_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_E	[610301110] Es wurde mehr als ein Nutzdatenblock in der Eingabedatei gefunden.

LEMENTE	
ERIC_IO_UNB EKANNTENART	[610301111] Der im Transferheader-Element "Datenart" angegebene Wert ist unbekannt.
ERIC_IO_REA DER_UNTERS ACHBEREICH_ UNGUELTIG	[610301114] Ungültiger oder fehlender Wert für den Untersuchungsbereich.
ERIC_IO_REA DER_ZU_VIEL E_NUTZDATE NBLOCK_ELE MENTE	[610301115] Es wurden zu viele Nutzdatenblöcke in der Eingabedatei gefunden.
ERIC_IO_REA DER_STEUERZ EICHEN_IM_T RANSFERHEA DER	[610301150] Es wurden ungültige Steuerzeichen im TransferHeader-Element gefunden.
ERIC_IO_REA DER_STEUERZ EICHEN_IM_N UTZDATENHE ADER	[610301151] Es wurden ungültige Steuerzeichen im NutzdatenHeader-Element gefunden.
ERIC_IO_REA DER_STEUERZ EICHEN_IN_D EN_NUTZDAT EN	[610301152] Es wurden ungültige Steuerzeichen im Nutzdaten-Element gefunden.
ERIC_IO_REA DER_RABE_FE HLER	[610301170] Es wurden Fehler in den Angaben zur Referenzierung von Belegen (RABE) gefunden. Details stehen im Logfile (eric.log).
ERIC_IO_REA DER_KEINE_R ABEID	[610301171] Es wurde keine RabeId angegeben, Details stehen ggf. im Logfile (eric.log)
ERIC_IO_REA DER_RABEID_ UNGUELTIG	[610301172] Es wurde eine ungültige RabeId gefunden, Details stehen ggf. im Logfile (eric.log)
ERIC_IO_REA DER_RABE_VE RIFIKATIONS ID_UNGUELTIG	[610301173] Es wurde eine ungültige VerifikationsId gefunden, Details stehen ggf. im Logfile (eric.log)
ERIC_IO_REA DER_RABE_RE FERENZID_UN GUELTIG	[610301174] Es wurde eine ungültige ReferenzId gefunden, Details stehen ggf. im Logfile (eric.log)
ERIC_IO_REA DER_RABE_RE FERENZID_NI CHT_ERLAUB T	[610301175] Es wurde eine ReferenzId für ein Feld angegeben, das keine ReferenzId erlaubt. Details stehen ggf. im Logfile (eric.log)

ERIC_IO_READER_RABEFERENZIDS_NICHT_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_VALIDIERUNGSGEFUEHRER	[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_FEHLERHAFT	[610301300] Falsche Übergabeparameter für die Funktion.
ERIC_IO_UNGUELTIGE_UTF8_SEQUENZ	[610301400] Der Parameter enthält ungültige UTF-8 Multibytesequenzen.
ERIC_IO_UNGUELTIGE_ZEICHEN_IN_PARAMETER	[610301401] Der Parameter enthält mindestens ein unzulässiges Zeichen.
ERIC_PRINTINTERNER_Fehler	[610501001] Verarbeitung fehlerhaft, keine genaueren Informationen vorhanden.
ERIC_PRINT_D	[610501002] Keine Druckvorlage für die angegebene Kombination aus

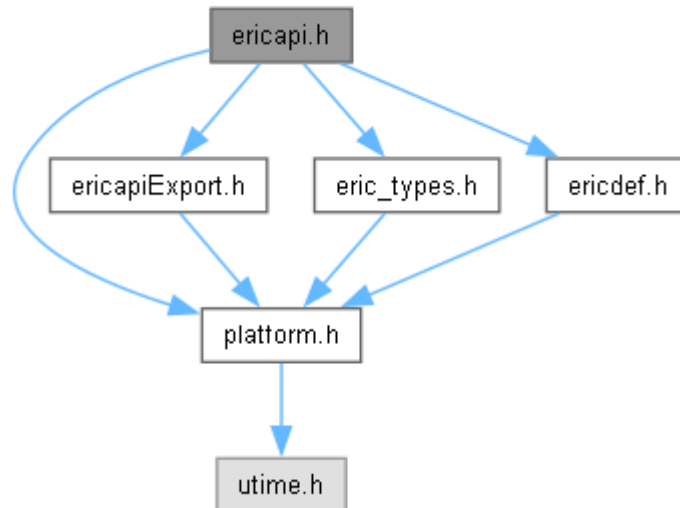
RUCKVORLAGE_NICHT_GEFUNDEN	Unterfallart und Veranlagungszeitraum gefunden. Bitte prüfen Sie die installierten Druckvorlagen.
ERIC_PRINT_UNGUELTIGER_DATEI_Pfad	[610501004] Es wurde ein falscher Dateipfad angegeben, es fehlen Zugriffsrechte oder die Datei wird aktuell von einer anderen Anwendung verwendet.
ERIC_PRINT_INITIALIZIERUNG_FEHLERHAFT	[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_STEUERFALL_NICHT_UNTERSTÜTZT	[610501011] Die Kombination aus Unterfallart und Veranlagungszeitraum wird nicht unterstützt.
ERIC_PRINT_FÜSSTEXT_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 zurück gesetzt.
- [ERICAPI_IMPORT](#) int [EricEinstellungLesen](#) (const char *name, [EricRueckgabepufferHandle](#) rueckgabePuffer)
Der Wert der API-Einstellung name wird im rueckgabePuffer zurück geliefert.
- [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 zurück gesetzt.
- [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 zurück geliefert.

- [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.
- [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 [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 in cryptoParameter zurückgeliefert.
- [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 Kapitel "Datenverarbeitung mit ERiC" im Entwicklerhandbuch. 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 Entwicklerhandbuch 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 <code>datenartVersion</code> ist der Datenartversionmatrix zu entnehmen, siehe Dokumentation\Datenartversionmatrix.xml und ERiC-Entwicklerhandbuch.pdf. Dieser Parameter darf nicht NULL sein und muss zu den XML-Eingangsdaten passen.
in	<i>bearbeitungsFlags</i>	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 . Bei welchen Anwendungsfällen welche Flags möglich oder notwendig sind, ist im Entwicklerhandbuch nachzulesen.
in	<i>druckParameter</i>	Parameter, der für den PDF-Druck benötigt wird, siehe eric_druck_parameter_t . Bei welchen Anwendungsfällen der Druckparameter möglich oder notwendig ist, ist im Entwicklerhandbuch nachzulesen. Soll kein PDF-Druck erfolgen, so ist NULL zu übergeben.
in	<i>cryptoParameter</i>	Enthält die für den authentifizierten Versand benötigten Informationen und darf nur dann übergeben werden, siehe eric_verschluesselungs_parameter_t . Erfolgt kein authentifizierter Versand, so ist NULL zu übergeben.
in,out	<i>transferHandle</i>	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"). Wenn bei der Datenabholung kein Versandflag gesetzt ist (nur Validierung), darf dem <code>transferHandle</code> auch ein Nullzeiger (NULL) übergeben werden. Bei allen anderen Anwendungsfällen ist immer NULL zu übergeben.
out	<i>rueckgabeXmlPuffer</i>	Handle auf einen Rückgabepuffer, in den beim Versand Telenummer und Ordnungsbegriff, Hinweise oder Fehler bei der Regelprüfung geschrieben werden, siehe Inhalt des Rückgabepuffers und des Serverantwortpuffers und EricRueckgabepufferHandle .
out	<i>serverantwortXmlPuffer</i>	Handle auf einen Rückgabepuffer, in den beim Versand die Antwort des Empfangsservers geschrieben wird, siehe Inhalt des Rückgabepuffers und des Serverantwortpuffers und EricRueckgabepufferHandle .

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 THEADER](#),
[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 Entwicklerhandbuch.

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 [EricGetErrorMessageFromXMLAnswer\(\)](#) 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, Kapitel "Anwendungsfälle von EricBearbeiteVorgang()"
- ERiC-Entwicklerhandbuch.pdf, Kapitel der jeweiligen Datenart
- ERiC-Entwicklerhandbuch.pdf, Kapitel "Datenabholung"
- ERiC-Entwicklerhandbuch.pdf, Kapitel "Größenbegrenzung der Eingangsdaten"
- ERiC-Entwicklerhandbuch.pdf, Kapitel "Funktionen für Fortschrittcallbacks"
- [EricHoleFehlerText\(\)](#)
- [EricGetErrorMessageFromXMLAnswer\(\)](#)
- [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_INITIALISIERT](#)
- [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 Entwicklerhandbuch Kapitel "Übergabe von Pfaden an ERiC API-Funktionen"

Parameter

in	<i>psePath</i>	In dem angegebenen Pfad liegt das Schlüsselpaar (eric_private.p12 und eric_public.cer).
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 unter Dokumentation\Schnittstellenbeschreibungen\ nachgeschlagen werden.

- Nicht unterstützte Datenartversionen: ElsterKMV
- alle Bilanz Datenartversionen

Parameter

in	<i>xml</i>	XML-Zeichenfolge
in	<i>datenartVersion</i>	Die datenartVersion ist der Datenartversionmatrix zu entnehmen, siehe Dokumentation\Datenartversionmatrix.xml und ERiC-Entwicklerhandbuch.pdf. Dieser Parameter darf nicht NULL sein und muss zu den XML-Eingangsdaten passen.
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_UNTERSTUEZT](#): 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_GENUEGEND_ARBEITSSPEICHER](#)
- [ERIC_GLOBAL_UNKNOWN](#)
-
- **Nur bei Verwendung des neuen Personalausweises:**
- [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\(\)](#)
- [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.p12: Enthält den privaten Schlüssel. Der Zugriff ist über die pin geschützt.
- eric.sfv: Enthält die Prüfsumme der Dateien eric_public.cer und eric_private.p12. 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 Entwicklerhandbuch 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>	XML-Datensatz, für den der TransferHeader erzeugt werden soll. Es kann entweder ein komplettes Elster-XML oder nur der
----	------------	---

		<p>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. 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 im Kapitel "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>	<p>Der Wert entspricht dem XML-Element "DatenLieferant", wie es im Schema des Transferheaders der ElsterBasis-XML-Schnittstelle definiert ist.</p> <p>ERiC konvertiert bei diesem Parameter Sonderzeichen in Entitätenreferenzen.</p>
in	<i>versionClient</i>	<p>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 Element VersionClient im Transferheader erzeugt.</p> <p>ERiC konvertiert bei diesem Parameter Sonderzeichen in Entitätenreferenzen.</p>
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>uuidRueckgabePuffer</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 abgeholten 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. "Datenabholung"

ERICAPI_IMPORT int EricEinstellungAlleZuruecksetzen (void)

Alle Einstellungen werden auf den jeweiligen Standardwert zurück gesetzt.

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` zurück geliefert.

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 Entwicklerhandbuch 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 zurück gesetzt.

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.

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 zurück geliefert.

- 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 Dokumentation\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 [EricGetErrorMessagesFromXMLAnswer](#) (const char * xml, [EricRueckgabepufferHandle](#) transferticketPuffer, [EricRueckgabepufferHandle](#) returncodeTHPuffer, [EricRueckgabepufferHandle](#) fehlertextTHPuffer, [EricRueckgabepufferHandle](#) returncodesUndFehlertexteNDHXmlPuffer)

Aus dem Antwort-XML des Finanzamtservers 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"?>
<EricGetErrorMessagesFromXMLAnswer
xmlns="http://www.elster.de/EricXML/1.0/EricGetErrorMessagesFromXMLAnswer">
  <Fehler>
    <Code>1</Code>
    <Meldung>Fehlermeldung 1</Meldung>
  </Fehler>
  <Fehler>
    <Code>2</Code>
    <Meldung>Fehlermeldung 2</Meldung>
  </Fehler>
  (...)
</EricGetErrorMessagesFromXMLAnswer>
```

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>	<ul style="list-style-type: none"> ◦ Handle zu einem der folgenden Zertifikate: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. In diesem Wert ist kodiert abgelegt, ob eine PIN-Eingabe erforderlich ist und welche PIN-Statusinformationen unterstützt werden. 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"> ◦ 0x00: Keine PIN-Angabe erforderlich, kein PIN-Status unterstützt. ◦ 0x01: PIN-Angabe für Signatur erforderlich. ◦ 0x02: PIN-Angabe für Entschlüsselung erforderlich. ◦ 0x04: PIN-Angabe für Verschlüsselung des Zertifikats erforderlich. ◦ 0x08: reserviert (wird derzeit nicht verwendet) ◦ 0x10: PIN-Status "Pin Ok" wird unterstützt. ◦ 0x20: PIN-Status "Der letzte Versuch der Pin-Eingabe schlug fehl" wird unterstützt. ◦ 0x40: PIN-Status "Beim nächsten fehlerhaften Versuch wird die Pin gesperrt" wird unterstützt. ◦ 0x80: PIN-Status "Pin ist gesperrt" wird unterstützt. ◦ Falls vom Aufrufer NULL übergeben wird, gibt die

		Funktion nichts zurück.
in	<i>pathToKeystore</i>	<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 Entwicklerhandbuch 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](#)
-
- **Nur bei Verwendung des neuen Personalausweises:**
- [ERIC TRANSFER EID CLIENTFEHLER](#)
- [ERIC TRANSFER EID FEHLENDEFELDER](#)
- [ERIC TRANSFER EID IDENTIFIKATIONABGEBROCHEN](#)
- [ERIC TRANSFER EID NPABLOCKIERT](#)
- [ERIC TRANSFER EID IDNRNICHT EINDEUTIG](#)
- [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>	<ul style="list-style-type: none"> ◦ Mögliche Rückgabewerte: 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 in `cryptoParameter` zurückgeliefert.

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 GENUEGEND 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_GENUEGEND_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>
    <GueltigBis>230817152116Z</GueltigBis>

<Signaturalgorithmus>sha1WithRSAEncryption(1.2.840.113549.1.1.5)</Signaturalgorith
mus>
  <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 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 [EricPruefeSteuernummer\(\)](#) 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).
- [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 EricPruefeIBAN (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\(\)](#)
- Prüfung_der_Steuer_und_Steueridentifikatsnummer.pdf, siehe [Entwicklerbereich](#) bei [ELSTER](#).

ERICAPI_IMPORT int EricPruefeWIdNr (const byteChar * wIdNr)

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

Parameter

in	<i>wIdNr</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 Entwicklerhandbuch 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_UNTERSTUEZT](#)
- [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 NULL
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 <code>NULL</code> .
in	<i>schreibeEricLogDatei</i>	<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 <code>NULL</code> sein.

		<ul style="list-style-type: none"> ◦ 0 Falls funktion != NULL werden keine Log-Nachrichten nach eric.log geschrieben, andernfalls werden die Log-Nachrichten nach eric.log 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 entgegen nehmen, 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\(\)](#)
- [EricRueckgabepufferFreigeben\(\)](#)

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

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

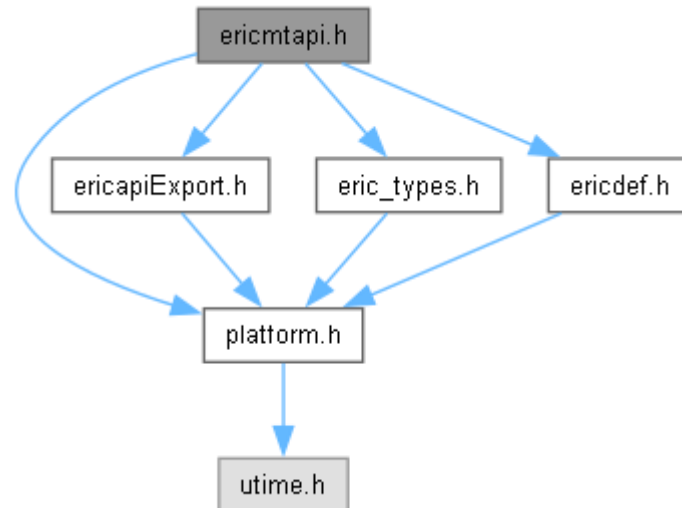
- [EricSystemCheck\(\)](#)

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_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 [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 abgeholt und verschlüsselten Daten entschlüsselt.
- [ERICAPI_IMPORT](#) int [EricMtEinstellungAlleZuruecksetzen](#) ([EricInstanzHandle](#) instanz)
Alle Einstellungen, der übergebenen ERiC-Instanz werden auf den jeweiligen Standardwert zurück gesetzt.
- [ERICAPI_IMPORT](#) int [EricMtEinstellungLesen](#) ([EricInstanzHandle](#) instanz, const char *name, [EricRueckgabepufferHandle](#) rueckgabePuffer)
Der Wert der API-Einstellung name wird im rueckgabePuffer zurück geliefert.
- [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 zurück gesetzt.
- [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 zurück geliefert.
- [ERICAPI_IMPORT](#) int [EricMtGetErrorMessagesFromXMLAnswer](#) ([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 [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 in cryptoParameter zurückgeliefert.
- [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 [EricMtPruefeWIdNr](#) ([EricInstanzHandle](#) instanz, const [byteChar](#) *wIdNr)
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 Kapitel "Datenverarbeitung mit ERiC" im Entwicklerhandbuch. 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 Entwicklerhandbuch 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 zu entnehmen, siehe Dokumentation\Datenartversionmatrix.xml und ERiC-Entwicklerhandbuch.pdf. Dieser Parameter darf nicht NULL sein und muss zu den XML-Eingangsdaten passen.
in	<i>bearbeitungsFlags</i>	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 . Bei welchen Anwendungsfällen welche Flags möglich oder notwendig sind, ist im Entwicklerhandbuch nachzulesen.
in	<i>druckParameter</i>	Parameter, der für den PDF-Druck benötigt wird, siehe eric_druck_parameter_t . Bei welchen Anwendungsfällen der Druckparameter möglich oder notwendig ist, ist im Entwicklerhandbuch nachzulesen. Soll kein PDF-Druck erfolgen, so ist NULL zu übergeben.
in	<i>cryptoParameter</i>	Enthält die für den authentifizierten Versand benötigten Informationen und darf nur dann übergeben werden, siehe eric_verschluesselungs_parameter_t . Erfolgt kein authentifizierter Versand, so ist NULL zu übergeben.
in,out	<i>transferHandle</i>	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"). Wenn bei der Datenabholung kein Versandflag gesetzt ist (nur Validierung), darf dem <code>transferHandle</code> auch ein Nullzeiger (NULL) übergeben werden. Bei allen anderen

		Anwendungsfällen ist immer NULL zu übergeben.
out	<i>rueckgabeXmlPuffer</i>	Handle auf einen Rückgabepuffer, in den beim Versand Telenummer und Ordnungsbegriff, Hinweise oder Fehler bei der Regelprüfung geschrieben werden, siehe Inhalt des Rückgabepuffers und des Serverantwortpuffers und EricRueckgabepufferHandle .
out	<i>serverantwortXmlPuffer</i>	Handle auf einen Rückgabepuffer, in den beim Versand die Antwort des Empfangsservers geschrieben wird, siehe Inhalt des Rückgabepuffers und des Serverantwortpuffers und EricRueckgabepufferHandle .

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-Annahmeservers. Bei einer Datenabholung kann diese ausgewertet werden. Details hierzu befinden sich im Entwicklerhandbuch.

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, Kapitel "Anwendungsfälle von EricBearbeiteVorgang()"
- ERiC-Entwicklerhandbuch.pdf, Kapitel der jeweiligen Datenart
- ERiC-Entwicklerhandbuch.pdf, Kapitel "Datenabholung"
- ERiC-Entwicklerhandbuch.pdf, Kapitel "Größenbegrenzung der Eingangsdaten"
- ERiC-Entwicklerhandbuch.pdf, Kapitel "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 Entwicklerhandbuch 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 (eric_private.p12

		und eric_public.cer).
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 unter Dokumentation\Schnittstellenbeschreibungen\ nachgeschlagen werden.

- 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 zu entnehmen, siehe Dokumentation\Datenartversionmatrix.xml und ERiC-Entwicklerhandbuch.pdf. Dieser Parameter darf nicht NULL sein und muss zu den XML-Eingangsdaten passen.
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_UNTERSTUEZT](#): 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](#)
-
- **Nur bei Verwendung des neuen Personalausweises:**
- [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\(\)](#)
- [EricMtGetPinStatus\(\)](#)
- ERiC-Entwicklerhandbuch.pdf, Kap. "Authentifizierung mit dem neuen Personalausweis (nPA)"

ERICAPI_IMPORT int EricMtCreateKey (**EricInstanzHandle** instanz, const **byteChar** * pin, const **byteChar** * pfpad, 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 `pfpad` 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.p12: Enthält den privaten Schlüssel. Der Zugriff ist über die `pin` geschützt.
- eric.sfv: Enthält die Prüfsumme der Dateien eric_public.cer und eric_private.p12. 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 Entwicklerhandbuch 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>	XML-Datensatz, für den der TransferHeader erzeugt werden soll. Es kann entweder ein komplettes Elster-XML oder nur der Datenteil übergeben werden. ERiC nimmt bei diesem Parameter keine Konvertierung von Sonderzeichen in Entitätenreferenzen vor. Attribute, die in den Start-Tags der Elemente "Elster" bzw. "DatenTeil" im übergebenen XML-Datensatz definiert werden, werden nicht in das Rückgabe-XML übernommen. 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. Im Rückgabe-XML werden im Start-Tag des Elements "Elster" die URI "http://www.elster.de/elsterxml/schema/v11" als Default-Namensraum definiert. Die dem Element "DatenTeil" untergeordneten Elemente aus dem übergebenen XML-Datensatz werden unverändert übernommen. Der allgemeine Aufbau des Elster-XMLs wird im ERiC-Entwicklerhandbuch.pdf im Kapitel "Datenverarbeitung mit ERiC" beschrieben.
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 Element VersionClient im Transferheader 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-Namespaces-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>uuidRueckgabePuffer</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 abgeholten 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

		/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]/Dateinhalt.
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

- [EricMtHoleFehlerText\(\)](#)
- ERiC-Entwicklerhandbuch.pdf, Kap. "Datenabholung"

[ERICAPI_IMPORT](#) int EricMtEinstellungAlleZuruecksetzen ([EricInstanzHandle](#) instanz)

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

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` zurück geliefert.

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 Entwicklerhandbuch 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 zurück gesetzt.

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.

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 zurück geliefert.

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 Dokumentation\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 Finanzamtservers 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>returncodesUndFehlerTexteNDHXMLPuffer</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_exten.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>	<ul style="list-style-type: none"> ◦ Handle zu einem der folgenden Zertifikate:Portalzertifikat ◦ clientseitig erzeugtes Zertifikat ◦ Ad Hoc-Zertifikat für den neuen Personalausweis
out	<i>iInfoPinSupport</i>	<p>Wird in <i>iInfoPinSupport</i> ein Zeiger ungleich NULL übergeben und die Funktion mit ERIC_OK beendet, dann enthält <i>iInfoPinSupport</i> einen vorzeichenlosen Integer-Wert. In diesem Wert ist kodiert abgelegt, ob eine PIN-Eingabe erforderlich ist und welche PIN-Statusinformationen unterstützt werden. 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"> ◦ 0x00: Keine PIN-Angabe erforderlich, kein PIN-Status unterstützt. ◦ 0x01: PIN-Angabe für Signatur erforderlich. ◦ 0x02: PIN-Angabe für Entschlüsselung erforderlich. ◦ 0x04: PIN-Angabe für Verschlüsselung des Zertifikats erforderlich. ◦ 0x08: reserviert (wird derzeit nicht verwendet) ◦ 0x10: PIN-Status "Pin Ok" wird unterstützt. ◦ 0x20: PIN-Status "Der letzte Versuch der Pin-Eingabe schlug fehl" wird unterstützt. ◦ 0x40: PIN-Status "Beim nächsten fehlerhaften Versuch wird die Pin gesperrt" wird unterstützt. ◦ 0x80: PIN-Status "Pin ist gesperrt" wird unterstützt. ◦ Falls vom Aufrufer NULL übergeben wird, gibt die Funktion nichts zurück.
in	<i>pathToKeystore</i>	<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):

		<p>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> <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 Entwicklerhandbuch 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](#)
-
- **Nur bei Verwendung des neuen Personalausweises:**
- [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>	<ul style="list-style-type: none"> ◦ Mögliche Rückgabewerte: 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 in `cryptoParameter` zurückgeliefert.

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>
```

```

</FinanzamtLand>
(...)
</EricHoleFinanzamtLandNummern>

```

Rückgabe

- [ERIC_OK](#)
- [ERIC_GLOBAL_NULL_PARAMETER](#)
- [ERIC_GLOBAL_COMMONDATA_NICHT_VERFUEGBAR](#)
- [ERIC_GLOBAL_NICHT_GENUEGEND_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\EricHoleFinanzamtsdate n.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\EricHoleTestFinanzaem

		ter.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>
<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. Das Erzeugen einer ERiC-Instanz ist ressourcen- und zeitintensiv. Zum Beenden einer ERiC-Instanz ist [EricMtInstanzFreigegeben\(\)](#) 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

- [EricMtInstanzFreigegeben\(\)](#)

[ERICAPI_IMPORT int](#) EricMtInstanzFreigegeben ([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 EricMtPruefeIBAN ([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 EricMtPruefeWIdNr ([EricInstanzHandle](#) instanz, const [byteChar](#) * wIdNr)

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>wIdNr</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>	Folgende Zertifikatstypen werden unterstützt: 1. Clientseitig erzeugtes Zertifikat: Pfad zum Verzeichnis, in dem sich die Zertifikats-Datei (.cer) und die Datei mit dem privaten Schlüssel (.p12) befinden.

		<p>Diese Kryptomittel wurden mit EricMtCreateKey() erzeugt. Der Pfad zum Verzeichnis ist bei clientseitig erzeugten Zertifikaten relativ zum aktuellen Arbeitsverzeichnis oder absolut anzugeben.</p> <p>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.</p> <p>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.</p> <p>4. Signaturkarte: Pfad zur Treiberdatei, welcher einen Zugriff auf die Signaturkarte ermöglicht, siehe (2). Weitere Informationen in der Anleitung zur Signaturkarte.</p>
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 [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 Entwicklerhandbuch 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 NULL
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 NULL
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 <code>NULL</code> .
in	<i>schreibeEricLogDatei</i>	<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 <code>NULL</code> 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 entgegen nehmen, 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_GENUEGEND_ARBEITSSPEICHER](#)
- weitere, siehe [eric_fehlercodes.h](#)

Siehe auch

- [EricMtSystemCheck\(\)](#)

ericversion.h-Dateireferenz

Bereitstellung der ERiC API-Version über C-Präprozessor Makros. Die ERiC API-Version entspricht nicht unbedingt der Version des Setup-Pakets.

Makrodefinitionen

- #define [ERIC_MAJOR_VERSION](#) 41
 - #define [ERIC_MINOR_VERSION](#) 6
 - #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 41

#define ERIC_MINOR_VERSION 6

#define ERIC_PATCH_VERSION 2

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 [EtkPruefeWIdNr](#) (const char *wIdNr)
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

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 EtkPruefelBAN (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

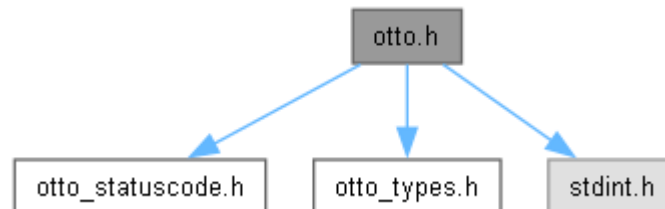
- ERiC-Entwicklerhandbuch.pdf, Kap. "Prüfung der Wirtschafts-Identifikationsnummer (W-IdNr.)"

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 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. 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.
- [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 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 `OttoEmpfang`-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 Sicherheitstoken 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 Entwicklerhandbuch 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](#) **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 NULL , 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 NULL 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 NULL 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

- [OttoInstanzFreigegeben\(\)](#)
- [OttoLogCallback](#)

[OttoStatusCode](#) [OttoInstanzFreigegeben](#) ([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 NULL 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_*` bei Problemen mit dem übergebenen Zertifikat
- weitere, siehe [otto_statuscode.h](#)

Siehe auch

- [OttoPruefsummeErzeugen\(\)](#)
- [OttoPruefsummeAktualisieren\(\)](#)
- [OttoPruefsummeFreigeben\(\)](#)

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\(\)](#)
- [OttoRueckgabepufferFreigeben\(\)](#)

[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 datenBlockGroesse)

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 NULL wird der Aufruf ignoriert.
in	<i>datenBlockGroesse</i>	Größe des Arrays <code>datenBlock</code> 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 anderen 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	instanz	Handle der Otto-Instanz, die das Zertifikatsobjekt verwenden soll.
in	<i>zertifikatsPfad</i>	<p>Pfad zum Sicherheitstoken, folgende Angaben sind möglich: 1. Clientseitig erzeugtes Zertifikat: Pfad zum Verzeichnis, in dem sich die Zertifikats-Datei (.cer) und die Datei mit dem privaten Schlüssel (.p12) befinden. Diese Sicherheitstoken wurden mit EricMtCreateKey() bzw. EricCreateKey() erzeugt. Der Pfad zum Verzeichnis ist bei clientseitig erzeugten Zertifikaten relativ zum aktuellen Arbeitsverzeichnis oder absolut anzugeben.</p> <p>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.</p> <p>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.</p> <p>4. Signaturkarte: (**) Pfad zur Treiberdatei, welcher einen Zugriff auf die Signaturkarte ermöglicht, siehe (*). Weitere Informationen in der Anleitung zur Signaturkarte.</p> <p>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> <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 Entwicklerhandbuch 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\(\)](#)

[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](#)

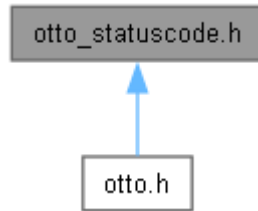
Siehe auch

- [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 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_APPLET](#) = 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_INTERN ER_FEHLER	Es trat ein interner Fehler auf, Details stehen im otto.log.
OTTO_UNBEK ANNTER_FEH LER	Es trat ein unerwarteter Fehler auf, Details stehen im otto.log.
OTTO_NPA_ZE RTIFIKATFEH LER	Es trat ein Fehler bei der Erzeugung eines Ad-hoc-Zertifikats für den nPA auf, Details stehen ggf. im otto.log.
OTTO_TRANSF ER_FEHLER	Es trat ein Fehler beim Transfer auf, Details stehen ggf. im otto.log.
OTTO_TRANSF ER_INIT	Es trat ein Fehler bei der Initialisierung des Transfers auf.

OTTO_TRANSF ER_CONNECT SERVER	Es konnte keine Verbindung zum OTTER-Server aufgebaut werden.
OTTO_TRANSF ER_CONNECT PROXY	Es konnte keine Verbindung zum Proxy aufgebaut werden.
OTTO_TRANSF ER_TIMEOUT	Bei der Kommunikation mit dem Server kam es zu einer Zeitüberschreitung.
OTTO_TRANSF ER_PROXYAU TH	Der Proxy erwartet Anmeldedaten oder der Proxy hat die Verbindung abgelehnt.
OTTO_TRANSF ER_UNAUTHO RIZED	Der Client darf die Schnittstelle nicht verwenden.
OTTO_TRANSF ER_NOT_FOU ND	Der OTTER-Server hat das Objekt nicht gefunden.
OTTO_TRANSF ER_SERVER_F EHLER	Der OTTER-Server hat einen unerwarteten Fehler gemeldet. Möglicherweise ist ein Retry sinnvoll. Details stehen im otto.log.
OTTO_TRANSF ER_DECODING	Die empfangenen Daten konnten nicht dekodiert werden.
OTTO_TRANSF ER_EID_ZERTI FIKATFEHLER	Es konnte kein Ad-hoc-Zertifikat für den Personalausweis oder den Aufenthaltstitel erzeugt bzw. gefunden werden, Details stehen ggf. im otto.log.
OTTO_TRANSF ER_EID_KEINC LIENT	Der eID-Client ist nicht erreichbar. Wahrscheinlich wurde er nicht gestartet oder die übergebene lokale URL ist nicht korrekt.
OTTO_TRANSF ER_EID_KEIN KONTO	Für die Identifikationsnummer des Benutzers existiert kein Konto bei ELSTER.
OTTO_TRANSF ER_EID_CLIEN TFEHLER	Der eID-Client hat einen Fehler gemeldet. Details zu dem Fehler finden Sie im Log des eID-Clients oder ggf. im otto.log.
OTTO_TRANSF ER_EID_NPAB LOCKIERT	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_HAN DLE	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_INSTANZEN_INKONSISTENT	Eines der übergebenen Otto-Objekte wurde mit einer anderen Otto-Instanz erstellt.
OTTO_INSTANZ_UNTEROBJEKTE_NICHT_FREIGEgeben	Mit dieser Instanz wurden Unterobjekte erzeugt, die noch nicht freigegeben worden sind.
OTTO_LOG_FEHLER	Die Protokolldatei konnte nicht erzeugt oder geöffnet werden.
OTTO_FUNKTION_NICHT_UNTERSTUETZT	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-Authentifizierungsschema angegeben.

OTTO_VERSAND_ABGESCHLOSSEN	Der Versand wurde bereits abgeschlossen.
OTTO_VERSAND_ZUGROSSDATENMENGE	Die Versandsdaten dürfen die Maximalgröße nicht überschreiten.
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_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_ESIGNER_NO_SERVICE	eSigner: Der PC/SC-Dienst ist nicht gestartet.
OTTO_ESIGNER_NO_SIGNATURE_CERTIFICATE_KEY	eSigner: Kein Signatur/Verschlüsselungszertifikat bzw. -schlüssel vorhanden.
OTTO_ESIGNER_OUT_OF_MEMORY	eSigner: Speicherallokation fehlgeschlagen.
OTTO_ESIGNER_P11_ENCRYPTION_KEY	eSigner: Fehler beim Zugriff auf Hard-Token-Entschlüsselungsschlüssel
OTTO_ESIGNER_P11_ENGINE_LOADED	eSigner: Die PKCS#11-Engine wird von einer anderen Bibliothek belegt.
OTTO_ESIGNER_P11_INITIAL_TOKEN_ACCESS_FAILED	eSigner: P11 Initialer Token-Zugriff fehlgeschlagen.

ILED	
OTTO_ESIGNER_P11_NO_ENC_CERT	eSigner: P11 Verschlüsselungszertifikat fehlt.
OTTO_ESIGNER_P11_NO_SIG_CERT	eSigner: P11 Signaturzertifikat fehlt.
OTTO_ESIGNER_P11_SIG_KEY	eSigner: Fehler beim Zugriff auf den Hard-Token-Signaturschlüssel
OTTO_ESIGNER_P11_SLOT_EMPTY	eSigner: Leere Slot-Liste, d.h. keine Karte eingesteckt.
OTTO_ESIGNER_P12_CREATE	eSigner: Temporäres PKCS#12-Token konnte nicht erzeugt werden.
OTTO_ESIGNER_P12_DECODE	eSigner: Fehler beim Dekodieren des PKCS#12-Objekts
OTTO_ESIGNER_P12_ENC_KEY	eSigner: Fehler beim Zugriff auf den Soft-PSE-Entschlüsselungsschlüssel
OTTO_ESIGNER_P12_SIG_KEY	eSigner: Fehler beim Zugriff auf den Soft-PSE-Signaturschlüssel
OTTO_ESIGNER_P12_NO_ENC_CERT	eSigner: P12 Verschlüsselungszertifikat fehlt.
OTTO_ESIGNER_P12_NO_SIG_CERT	eSigner: P12 Signaturzertifikat fehlt.
OTTO_ESIGNER_P12_READ	eSigner: Fehler beim Lesen des PKCS#12-Objekts
OTTO_ESIGNER_P7_DECODE	eSigner: Fehler beim Dekodieren des PKCS#7-Objekts
OTTO_ESIGNER_P7_READ	eSigner: Fehler beim Lesen des PKCS#7-Objekts
OTTO_ESIGNER_P7_RECEIPT	eSigner: Das Entschlüsselungszertifikat ist nicht in der Empfängerliste enthalten.
OTTO_ESIGNER_PIN_LOCKED	eSigner: Die PIN bzw. das Passwort ist gesperrt.
OTTO_ESIGNER_PIN_WRONG	eSigner: Die PIN bzw. das Passwort ist falsch.
OTTO_ESIGNER_PSE_PATH	eSigner: Der Pfad zum Soft-PSE ist falsch oder ungültig.

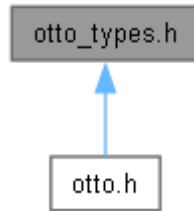
OTTO_ESIGNER_SC_ENC_KEY	eSigner: Fehler beim Zugriff auf den Stick-Entschlüsselungsschlüssel.
OTTO_ESIGNER_SC_INIT_FAILED	eSigner: Initialer Token-Zugriff auf die PC/SC-Schnittstelle fehlgeschlagen.
OTTO_ESIGNER_SC_NO_APPLET	eSigner: Kein unterstütztes Applet gefunden.
OTTO_ESIGNER_SC_NO_ENC_CERT	eSigner: PC/SC Verschlüsselungszertifikat fehlt.
OTTO_ESIGNER_SC_NO_SIG_CERT	eSigner: PC/SC Signaturzertifikat fehlt.
OTTO_ESIGNER_SC_SESSION	eSigner: Fehler in der Karten-Session.
OTTO_ESIGNER_SC_SIG_KEY	eSigner: Fehler beim Zugriff auf den Stick-Signaturschlüssel
OTTO_ESIGNER_SC_SLOT_EMPTY	eSigner: Es ist keine Karte bzw. kein Stick eingesteckt.
OTTO_ESIGNER_TOKEN_TYPE_MISMATCH	eSigner: Der Token-Typ der CA (Certification Authority, Zertifizierungsstelle) stimmt nicht mit dem internen Token-Typ überein.
OTTO_ESIGNER_USER_CANCEL	eSigner: Die Aktion wurde vom Benutzer abgebrochen.
OTTO_ESIGNER_VERIFY_CERT_CHAIN	eSigner: Die Zertifikatskette konnte nicht verifiziert werden.
OTTO_ESIGNER_DATA_NOT_INITIALIZED	eSigner: Die Datenstruktur ist nicht initialisiert.
OTTO_ESIGNER_ASN1_READ_BUFFER_TOO_SMALL	eSigner: Der Lesebuffer zum Dekodieren der ASN.1-Struktur ist zu klein.
OTTO_ESIGNER_ASN1_READ_DATA_INCOMPLETE	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_NICHTGENUEGEND_ARBEITSSPEICHER	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 soll nicht für jede Aufgabe neu erstellt und konfiguriert werden. Das Erstellen und Zerstören einer Otto-Instanz ist ressourcen- und zeitintensiv. Die Lebenszeit einer Otto-Instanz sollte beispielsweise eher der Lebenszeit eines Arbeiter-Threads in einem Pool entsprechen als der Verarbeitungsdauer einer einzelnen Aufgabe eines Arbeiter-Threads.

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 *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.

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:

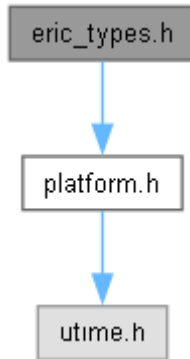
OTTOLOG_FEHLERMELDUNGEN	Fehler, die zum Abbruch der gewünschten Aktion führen.
OTTOLOG_WARNUNGEN	Hinweise auf Zustände, die zu Fehlern führen können.
OTTOLOG_INFORMATIONEN	Grobe Informationen über den Programmablauf und Werte.
OTTOLOG_DEBUGMELDUNGEN	Feingranulare Informationen über den Programmablauf und Werte.

eric_types.h-Dateireferenz

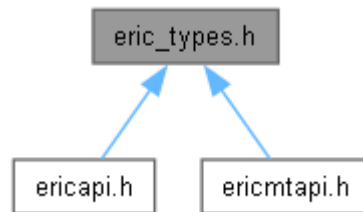
Definition von Datenstrukturen und Datentypen.

```
#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 }
 - 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 }
- Bearbeitungsflags für die Anwendungsfälle von [EricBearbeiteVorgang\(\)](#).*

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 sollen nicht für jede Aufgabe neu erstellt und konfiguriert werden. Das Erstellen und Zerstören einer ERiC-Instanz ist ressourcen- und zeitintensiv. Die Lebenszeit einer ERiC-Instanz sollte beispielsweise eher der Lebenszeit eines Arbeiter-Threads in einem Pool entsprechen, als der Verarbeitungsdauer einer einzelnen Aufgabe an einen Arbeiter-Thread.

ERiC-Instanzen können zwischen Threads ausgetauscht werden. Eine ERiC-Instanz darf aber nicht in zwei Threads gleichzeitig verwendet werden.

Siehe auch

- [EricMtInstanzErzeugen\(\)](#)
- [EricMtInstanzFreigegeben\(\)](#)

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 pdfDaten 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 entgegen nehmen, 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_FORTSC HRITTCALLBA CK_ID_EINLES EN	id , die beim Einlesen des XMLs von Fortschrittcallbacks ausgegeben wird.
ERIC_FORTSC HRITTCALLBA	id , die gemeldet wird, wenn die Daten zum Versand noch vorbereitet werden müssen.

CK_ID_VORBEREITEN	
ERIC_FORTSCHRITTCALLBACK_ID_VALIDIEREN	id , die beim Validieren der Eingangsdaten von Fortschrittcallbacks ausgegeben wird.
ERIC_FORTSCHRITTCALLBACK_ID_SENDEN	id , die beim Versand der Ausgangsdaten von Fortschrittcallbacks ausgegeben wird.
ERIC_FORTSCHRITTCALLBACK_ID_DRUCKEN	id , 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_VALIDIERE	Der Datensatz soll validiert werden.
ERIC_SENDE	Der Datensatz soll an den ELSTER Annahmeserver versendet werden.
ERIC_DRUCKE	Der Datensatz soll gedruckt werden.
ERIC_PRUEFE_HINWEISE	Der Datensatz soll auf Hinweise hin geprüft werden.
ERIC_VALIDIERE_OHNE_FREIGABEDATUM	<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_ER	Fehler, der zum Programmabbruch führt.
-------------	--

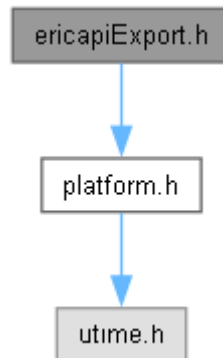
ROR	
ERIC_LOG_WARN	Hinweise auf Zustände, die zu Fehlern führen können.
ERIC_LOG_INFO	Grobe Informationen über den Programmablauf und Werte.
ERIC_LOG_DEBUG	Feingranulare Informationen über den Programmablauf und Werte.
ERIC_LOG_TRACE	Sehr feingranulare Informationen über den Programmablauf und Werte.

ericapiExport.h-Dateireferenz

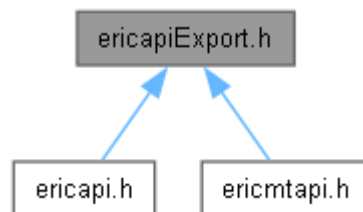
Attribute für dynamische Bibliotheken.

```
#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

ericdef.h-Dateireferenz

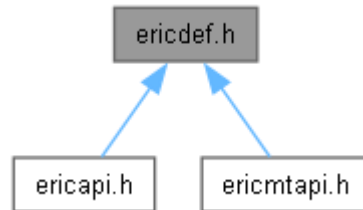
Konstanten und Definitionen für Übergabeparameter.

```
#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)
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.
- #define [EURO](#) (unsigned char)0x20AC

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.

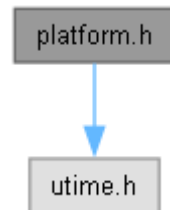
#define EURO (unsigned char)0x20AC

platform.h-Dateireferenz

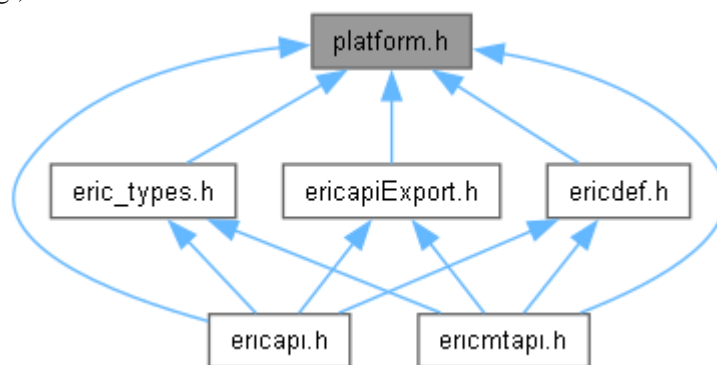
Konstanten für verschiedene Betriebssysteme.

```
#include <utime.h>
```

Include-Abhängigkeitsdiagramm für platform.h:



Dieser Graph zeigt, welche Datei direkt oder indirekt diese Datei enthält:



Makrodefinitionen

- `#define` [ATOI64](#) `atoll`
- `#define` [I64](#)(C)
- `#define` [HAS_FUTIME](#) `1`
- `#define` [UTIME_NEEDS_CLOSED_FILE](#) `0`

Typdefinitionen

- `typedef` `__plattformabhaengige_Implementierung` [uint32_t](#)
Definition eines vorzeichenlosen, 32 Bit breiten Integer-Typs.

Ausführliche Beschreibung

Konstanten für verschiedene Betriebssysteme.

Makro-Dokumentation

#define ATOI64 `atoll`

#define HAS_FUTIME `1`

#define I64(C)

Wert:

`C##LL`

#define UTIME_NEEDS_CLOSED_FILE `0`

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 11
adresse
 eric_zertifikat_parameter_t 11
ATOI64
 platform.h 161
authentifizierungsMethode
 OttoProxyKonfiguration 13
benutzerName
 OttoProxyKonfiguration 14
benutzerPasswort
 OttoProxyKonfiguration 14
beschreibung
 eric_zertifikat_parameter_t 11
byteChar
 eric_types.h 151
duplexDruck
 eric_druck_parameter_t 6
email
 eric_zertifikat_parameter_t 11
eric_bearbeitung_flag_t
 eric_types.h 155
ERIC_CRYPT_CORRUPTED
 eric_fehlercodes.h 30
ERIC_CRYPT_E_ASN1_NO_CONTENT_D
ATA
 eric_fehlercodes.h 31
ERIC_CRYPT_E_ASN1_NO_ENVELOPED_
DATA
 eric_fehlercodes.h 31
ERIC_CRYPT_E_ASN1_READ_BUFFER_T
OO_SMALL
 eric_fehlercodes.h 31
ERIC_CRYPT_E_ASN1_READ_DATA_INC
OMPLETE
 eric_fehlercodes.h 31
ERIC_CRYPT_E_BUSY
 eric_fehlercodes.h 27
ERIC_CRYPT_E_DATA_NOT_INITIALIZE
D
 eric_fehlercodes.h 31
ERIC_CRYPT_E_DECRYPT
 eric_fehlercodes.h 29
ERIC_CRYPT_E_ENCODE_ERROR
 eric_fehlercodes.h 29
ERIC_CRYPT_E_ENCODE_UNKNOWN
 eric_fehlercodes.h 28
ERIC_CRYPT_E_ENCRYPT
 eric_fehlercodes.h 29
ERIC_CRYPT_E_ESICL_EXCEPTION
 eric_fehlercodes.h 29
ERIC_CRYPT_E_ESIGNER_NICHT_GELA
DEN
 eric_fehlercodes.h 29
ERIC_CRYPT_E_INKOMPATIBLE_ESIGN
ER_VERSION
 eric_fehlercodes.h 29
ERIC_CRYPT_E_INTERN
 eric_fehlercodes.h 30
ERIC_CRYPT_E_INVALID_HANDLE
 eric_fehlercodes.h 27
ERIC_CRYPT_E_LOAD_DLL
 eric_fehlercodes.h 29
ERIC_CRYPT_E_MAX_SESSION
 eric_fehlercodes.h 27
ERIC_CRYPT_E_NO_SERVICE
 eric_fehlercodes.h 29
ERIC_CRYPT_E_NO_SIG_ENC_KEY
 eric_fehlercodes.h 29
ERIC_CRYPT_E_OUT_OF_MEM
 eric_fehlercodes.h 27
ERIC_CRYPT_E_P11_ENC_KEY
 eric_fehlercodes.h 28
ERIC_CRYPT_E_P11_ENGINE_LOADED
 eric_fehlercodes.h 29
ERIC_CRYPT_E_P11_INIT_FAILED
 eric_fehlercodes.h 31
ERIC_CRYPT_E_P11_NO_ENC_CERT
 eric_fehlercodes.h 31
ERIC_CRYPT_E_P11_NO_SIG_CERT
 eric_fehlercodes.h 31
ERIC_CRYPT_E_P11_SIG_KEY
 eric_fehlercodes.h 28
ERIC_CRYPT_E_P11_SLOT_EMPTY
 eric_fehlercodes.h 29
ERIC_CRYPT_E_P12_CREATE
 eric_fehlercodes.h 29
ERIC_CRYPT_E_P12_DECODE
 eric_fehlercodes.h 28
ERIC_CRYPT_E_P12_ENC_KEY
 eric_fehlercodes.h 28
ERIC_CRYPT_E_P12_NO_ENC_CERT
 eric_fehlercodes.h 31
ERIC_CRYPT_E_P12_NO_SIG_CERT
 eric_fehlercodes.h 31
ERIC_CRYPT_E_P12_READ
 eric_fehlercodes.h 28
ERIC_CRYPT_E_P12_SIG_KEY
 eric_fehlercodes.h 28
ERIC_CRYPT_E_P7_DECODE
 eric_fehlercodes.h 28
ERIC_CRYPT_E_P7_READ
 eric_fehlercodes.h 28
ERIC_CRYPT_E_P7_RECIPIENT
 eric_fehlercodes.h 28
ERIC_CRYPT_E_PIN_LOCKED
 eric_fehlercodes.h 28
ERIC_CRYPT_E_PIN_WRONG
 eric_fehlercodes.h 28
ERIC_CRYPT_E_PSE_PATH
 eric_fehlercodes.h 28
ERIC_CRYPT_E_SC_ENC_KEY

- eric_fehlercodes.h 31
- ERIC_CRYPT_E_SC_INIT_FAILED
 - eric_fehlercodes.h 31
- ERIC_CRYPT_E_SC_NO_APPLET
 - eric_fehlercodes.h 30
- ERIC_CRYPT_E_SC_NO_ENC_CERT
 - eric_fehlercodes.h 31
- ERIC_CRYPT_E_SC_NO_SIG_CERT
 - eric_fehlercodes.h 31
- ERIC_CRYPT_E_SC_SESSION
 - eric_fehlercodes.h 30
- ERIC_CRYPT_E_SC_SIG_KEY
 - eric_fehlercodes.h 31
- ERIC_CRYPT_E_SC_SLOT_EMPTY
 - eric_fehlercodes.h 30
- ERIC_CRYPT_E_TOKEN_TYPE_MISMATCH
 - eric_fehlercodes.h 29
- ERIC_CRYPT_E_USER_CANCEL
 - eric_fehlercodes.h 30
- ERIC_CRYPT_E_VERALTETE_ESIGNER_VERSION
 - eric_fehlercodes.h 29
- ERIC_CRYPT_E_VERIFY_CERT_CHAIN
 - eric_fehlercodes.h 29
- ERIC_CRYPT_E_XML_INIT
 - eric_fehlercodes.h 29
- ERIC_CRYPT_E_XML_PARSE
 - eric_fehlercodes.h 28
- ERIC_CRYPT_E_XML_SIG_ADD
 - eric_fehlercodes.h 28
- ERIC_CRYPT_E_XML_SIG_SIGN
 - eric_fehlercodes.h 28
- ERIC_CRYPT_E_XML_SIG_TAG
 - eric_fehlercodes.h 28
- ERIC_CRYPT_EIDKARTE_NICHT_UNTERSTUETZT
 - eric_fehlercodes.h 30
- ERIC_CRYPT_ERROR_CREATE_KEY
 - eric_fehlercodes.h 27
- ERIC_CRYPT_NICHT_UNTERSTUETZTES_PSE_FORMAT
 - eric_fehlercodes.h 30
- ERIC_CRYPT_PIN_BENOETIGT
 - eric_fehlercodes.h 30
- ERIC_CRYPT_PIN_ENTHAELT_UNGUELTIGE_ZEICHEN
 - eric_fehlercodes.h 30
- ERIC_CRYPT_PIN_STAERKE_NICHT_AUSREICHEND
 - eric_fehlercodes.h 30
- ERIC_CRYPT_SIGNATUR
 - eric_fehlercodes.h 30
- ERIC_CRYPT_ZERTIFIKAT
 - eric_fehlercodes.h 30
- ERIC_CRYPT_ZERTIFIKATSDATEI_EXISTIERT_BEREITS
 - eric_fehlercodes.h 30
- ERIC_CRYPT_ZERTIFIKATSPFAD_KEIN_VERZEICHNIS

- eric_fehlercodes.h 30
- eric_druck_parameter_t 5
- duplexDruck 6
- fussText 6
- pdfCallback 6
- pdfCallbackBenutzerdaten 6
- pdfName 6
- version 7
- vorschau 7
- ERIC_DRUCKE
 - eric_types.h 155
- eric_fehlercode
 - eric_fehlercodes.h 18
- eric_fehlercode_t
 - eric_fehlercodes.h 18
- eric_fehlercodes.h 15
- ERIC_CRYPT_CORRUPTED 30
- ERIC_CRYPT_E_ASN1_NO_CONTENT_DATA 31
- ERIC_CRYPT_E_ASN1_NO_ENVELOPE_DATA 31
- ERIC_CRYPT_E_ASN1_READ_BUFFER_TOO_SMALL 31
- ERIC_CRYPT_E_ASN1_READ_DATA_INCOMPLETE 31
- ERIC_CRYPT_E_BUSY 27
- ERIC_CRYPT_E_DATA_NOT_INITIALIZED 31
- ERIC_CRYPT_E_DECRYPT 29
- ERIC_CRYPT_E_ENCODE_ERROR 29
- ERIC_CRYPT_E_ENCODE_UNKNOWN 28
- ERIC_CRYPT_E_ENCRYPT 29
- ERIC_CRYPT_E_ESICL_EXCEPTION 29
- ERIC_CRYPT_E_ESIGNER_NICHT_GELADEN 29
- ERIC_CRYPT_E_INKOMPATIBLE_ESIGNER_VERSION 29
- ERIC_CRYPT_E_INTERN 30
- ERIC_CRYPT_E_INVALID_HANDLE 27
- ERIC_CRYPT_E_LOAD_DLL 29
- ERIC_CRYPT_E_MAX_SESSION 27
- ERIC_CRYPT_E_NO_SERVICE 29
- ERIC_CRYPT_E_NO_SIG_ENC_KEY 29
- ERIC_CRYPT_E_OUT_OF_MEM 27
- ERIC_CRYPT_E_P11_ENC_KEY 28
- ERIC_CRYPT_E_P11_ENGINE_LOADED 29
- ERIC_CRYPT_E_P11_INIT_FAILED 31
- ERIC_CRYPT_E_P11_NO_ENC_CERT 31
- ERIC_CRYPT_E_P11_NO_SIG_CERT 31
- ERIC_CRYPT_E_P11_SIG_KEY 28
- ERIC_CRYPT_E_P11_SLOT_EMPTY 29
- ERIC_CRYPT_E_P12_CREATE 29
- ERIC_CRYPT_E_P12_DECODE 28

ERIC_CRYPT_E_P12_ENC_KEY 28	ERIC_GLOBAL_BUFANR_UNBEKANNT 20
ERIC_CRYPT_E_P12_NO_ENC_CERT 31	ERIC_GLOBAL_BUNDESLAENDER_UNEINHEITLICH 24
ERIC_CRYPT_E_P12_NO_SIG_CERT 31	ERIC_GLOBAL_CHECK_CORRUPTED_NDS 22
ERIC_CRYPT_E_P12_READ 28	ERIC_GLOBAL_COMMONDATA_NICHT_VERFUEGBAR 21
ERIC_CRYPT_E_P12_SIG_KEY 28	ERIC_GLOBAL_DATEI_NICHT_GEFUNDEN 19
ERIC_CRYPT_E_P7_DECODE 28	ERIC_GLOBAL_DATEIZUGRIFF_VERWEIGERT 21
ERIC_CRYPT_E_P7_READ 28	ERIC_GLOBAL_DATENARTVERSION_UNBEKANNT 21
ERIC_CRYPT_E_P7_RECIPIENT 28	ERIC_GLOBAL_DATENARTVERSION_XML_INKONSISTENT 21
ERIC_CRYPT_E_PIN_LOCKED 28	ERIC_GLOBAL_DATENSATZ_ZU_GROSS 19
ERIC_CRYPT_E_PIN_WRONG 28	ERIC_GLOBAL_DRUCK_FUER_VERFAHREN_NICHT_ERLAUBT 22
ERIC_CRYPT_E_PSE_PATH 28	ERIC_GLOBAL_ECHTFALL_NICHT_ERLAUBT 19
ERIC_CRYPT_E_SC_ENC_KEY 31	ERIC_GLOBAL_EINSTELLUNG_NAME_UNGUELTIG 24
ERIC_CRYPT_E_SC_INIT_FAILED 31	ERIC_GLOBAL_EINSTELLUNG_WERT_UNGUELTIG 24
ERIC_CRYPT_E_SC_NO_APPLET 30	ERIC_GLOBAL_ERR_DEKODIEREN 24
ERIC_CRYPT_E_SC_NO_ENC_CERT 31	ERIC_GLOBAL_ERROR_XML_CREATE 20
ERIC_CRYPT_E_SC_NO_SIG_CERT 31	ERIC_GLOBAL_ERSTE_SEITE_DRUCK_NICHT_UNTERSTUETZT 22
ERIC_CRYPT_E_SC_SESSION 30	ERIC_GLOBAL_EWAZ_LANDESKUERZEL_UNBEKANNT 24
ERIC_CRYPT_E_SC_SIG_KEY 31	ERIC_GLOBAL_EWAZ_UNGUELTIG 24
ERIC_CRYPT_E_SC_SLOT_EMPTY 30	ERIC_GLOBAL_FEHLER_INITIALIZIERUNG 22
ERIC_CRYPT_E_TOKEN_TYPE_MISMATCH 29	ERIC_GLOBAL_FEHLERMELDUNG_NICHT_VORHANDEN 19
ERIC_CRYPT_E_USER_CANCEL 30	ERIC_GLOBAL_FUNKTION_NICHT_ERLAUBT 19
ERIC_CRYPT_E_VERALTETE_ESIGNER_VERSION 29	ERIC_GLOBAL_FUNKTION_NICHT_UNTERSTUETZT 24
ERIC_CRYPT_E_VERIFY_CERT_CHAIN 29	ERIC_GLOBAL_HERSTELLER_ID_NICHT_ERLAUBT 19
ERIC_CRYPT_E_XML_INIT 29	ERIC_GLOBAL_HINWEISE 19
ERIC_CRYPT_E_XML_PARSE 28	ERIC_GLOBAL_IBAN_FORMALER_FEHLER 23
ERIC_CRYPT_E_XML_SIG_ADD 28	ERIC_GLOBAL_IBAN_LAENDERCODE_FEHLER 23
ERIC_CRYPT_E_XML_SIG_SIGN 28	ERIC_GLOBAL_IBAN_LANDESFORMAT_FEHLER 23
ERIC_CRYPT_E_XML_SIG_TAG 28	ERIC_GLOBAL_IBAN_PRUEFZIFFER_FEHLER 23
ERIC_CRYPT_EIDKARTE_NICHT_UNTERSTUETZT 30	ERIC_GLOBAL_IDNUMMER_UNGUELTIG 24
ERIC_CRYPT_ERROR_CREATE_KEY 27	ERIC_GLOBAL_ILLEGAL_STATE 19
ERIC_CRYPT_NICHT_UNTERSTUETZTES_PSE_FORMAT 30	ERIC_GLOBAL_INKOMPATIBLE_VERSIONEN 23
ERIC_CRYPT_PIN_BENOETIGT 30	
ERIC_CRYPT_PIN_ENTHAELT_UNGUELTIGE_ZEICHEN 30	
ERIC_CRYPT_PIN_STAERKE_NICHT_AUSREICHEND 30	
ERIC_CRYPT_SIGNATUR 30	
ERIC_CRYPT_ZERTIFIKAT 30	
ERIC_CRYPT_ZERTIFIKATSDATEI_EXISTIERT_BEREITS 30	
ERIC_CRYPT_ZERTIFIKATSPFAD_KEY_N_VERZEICHNIS 30	
eric_fehlercode 18	
eric_fehlercode_t 18	
ERIC_GLOBAL_ARITHMETIKFEHLER 20	
ERIC_GLOBAL_BIC_FORMALER_FEHLER 23	
ERIC_GLOBAL_BIC_LAENDERCODE_FEHLER 23	

ERIC_GLOBAL_INTERNER_FEHLER 20	ERIC_GLOBAL_UNGUELTIGE_FLAG_ KOMBINATION 22
ERIC_GLOBAL_KEINE_DATEN_VORH ANDEN 19	ERIC_GLOBAL_UNGUELTIGE_INSTAN Z 21
ERIC_GLOBAL_LANDESNUMMER_BU FANR 20	ERIC_GLOBAL_UNGUELTIGE_PARAM ETER_VERSION 22
ERIC_GLOBAL_LANDESNUMMER_UN BEKANNT 20	ERIC_GLOBAL_UNGUELTIGER_PARA METER 22
ERIC_GLOBAL_LOG_EXCEPTION 21	ERIC_GLOBAL_UNKNOWN 18
ERIC_GLOBAL_MEHRFACHAUFRUFE_ NICHT_UNTERSTUETZT 23	ERIC_GLOBAL_UNKNOWN_PARAMET ER_ERROR 22
ERIC_GLOBAL_MEHRFACHE_INITIAL ISIERUNG 22	ERIC_GLOBAL_UPDATE_NECESSARY 24
ERIC_GLOBAL_NICHT_GENUEGEND_ ARBEITSSPEICHER 19	ERIC_GLOBAL_UTI_COUNTRY_NOT_S UPPORTED 23
ERIC_GLOBAL_NICHT_INITIALISIERT 21	ERIC_GLOBAL_VERSAND_ART_NICH T_UNTERSTUETZT 22
ERIC_GLOBAL_NO_VERSAND_IN_BET A_VERSION 19	ERIC_GLOBAL_VERSCHLUESSELUNG S_PARAMETER_NICHT_ANGEGEBE N 22
ERIC_GLOBAL_NULL_PARAMETER 24	ERIC_GLOBAL_VERSCHLUESSELUNG S_PARAMETER_NICHT_ERLAUBT 20
ERIC_GLOBAL_NUR_PORTALZERTIFI KAT_ERLAUBT 20	ERIC_GLOBAL_VERSCHLUESSELUNG SVERFAHREN_NICHT_UNTERSTUE TZT 23
ERIC_GLOBAL_NUTZDATENHEADER_ EMPFAENGER_NICHT_KORREKT 25	ERIC_GLOBAL_VORSATZ_UNGUELTI G 21
ERIC_GLOBAL_NUTZDATENHEADER VERSIONEN_UNEINHEITLICH 24	ERIC_GLOBAL_ZEITRAEUME_UNEIN HEITLICH 24
ERIC_GLOBAL_NUTZDATENTICKETS _NICHT_EINDEUTIG 24	ERIC_GLOBAL_ZULASSUNGSNUMME R_ZU_LANG 23
ERIC_GLOBAL_OEFFENTLICHER_SCH LUESSEL_UNGUELTIG 21	ERIC_IO_DATEI_INKORREKT 32
ERIC_GLOBAL_PLUGININITIALISIERU NG 23	ERIC_IO_DATENTEILENDNOTFOUND 34
ERIC_GLOBAL_PRUEF_FEHLER 19	ERIC_IO_DATENTEILNOTFOUND 34
ERIC_GLOBAL_PUFFER_UEBERLAUF 21	ERIC_IO_FALSCHES_VERFAHREN 32
ERIC_GLOBAL_PUFFER_UNGLEICHER _INSTANZ 21	ERIC_IO_FEHLER 32
ERIC_GLOBAL_PUFFER_ZUGRIFFSKO NFLIKT 20	ERIC_IO_MASTERDATENSERVICE_NI CHT_VERFUEGBAR 32
ERIC_GLOBAL_SEND_FLAG_MEHR_A LS_EINES 22	ERIC_IO_NDS_GENERIERUNG_FEHLG ESCHLAGEN 32
ERIC_GLOBAL_STEUERNUMMER_FA LSCHE_LAENGE 20	ERIC_IO_PARSE_FEHLER 32
ERIC_GLOBAL_STEUERNUMMER_NIC HT_NUMERISCH 20	ERIC_IO_READER_ANHAENGE_ZU_G ROSS 34
ERIC_GLOBAL_STEUERNUMMER_UN GUELTIG 20	ERIC_IO_READER_ANHANG_ZU_GRO SS 34
ERIC_GLOBAL_TESTMERKER_UNGUE LTIG 19	ERIC_IO_READER_ANHANG_ZU_KLEI N 34
ERIC_GLOBAL_TEXTPUFFERGROESS E_FIX 20	ERIC_IO_READER_FALSCHES_ENCOD ING 32
ERIC_GLOBAL_TRANSFERHANDLE 23	ERIC_IO_READER_FORMALE_FEHLER 32
ERIC_GLOBAL_TRANSPORTSCHLUES SEL_NICHT_ERLAUBT 21	ERIC_IO_READER_KEINE_RABEID 33
ERIC_GLOBAL_TRANSPORTSCHLUES SEL_TYP_FALSCH 21	ERIC_IO_READER_MEHRFACHE_NUT ZDATEN_ELEMENTE 32
	ERIC_IO_READER_MEHRFACHE_NUT ZDATENBLOCK_ELEMENTE 32
	ERIC_IO_READER_MEHRFACHE_STE UERFAELLE 32

ERIC_IO_READER_RABE_FEHLER	33	ERIC_PRINT_UNGUELTIGER_DATEI_P	FAD	35
ERIC_IO_READER_RABE_REFERENZI		ERIC_TRANSFER_COM_ERROR	25	
D_NICHT_ERLAUBT	33	ERIC_TRANSFER_EID_CLIENTFEHLER	27	
ERIC_IO_READER_RABE_REFERENZI		ERIC_TRANSFER_EID_FEHLENDEFEL	DER	27
DS_NICHT_EINDEUTIG	34	ERIC_TRANSFER_EID_IDENTIFIKATIO	NABGEBROCHEN	27
ERIC_IO_READER_RABE_VERIFIKATI	ONSID_UNGUELTIG	ERIC_TRANSFER_EID_IDNRNICHT	TEIN	
ERIC_IO_READER_RABEID_UNGUELT	IG	DEUTIG	27	
ERIC_IO_READER_SCHEMA_VALIDIE	RUNGSFEHLER	ERIC_TRANSFER_EID_KEINCLIENT	27	
ERIC_IO_READER_STEUERZEICHEN_I	M_NUTZDATENHEADER	ERIC_TRANSFER_EID_KEINKONTO	27	
ERIC_IO_READER_STEUERZEICHEN_I	M_TRANSFERHEADER	ERIC_TRANSFER_EID_NPABLOCKIER	T	27
ERIC_IO_READER_STEUERZEICHEN_I	N_DEN_NUTZDATEN	ERIC_TRANSFER_EID_SERVERFEHLE	R	27
ERIC_IO_READER_UNBEKANNT_XM	L_ENTITY	ERIC_TRANSFER_EID_ZERTIFIKATFE	HLE	27
ERIC_IO_READER_UNERWARTETE_E	LEMENTE	ERIC_TRANSFER_ERR_BEGINDATEN	GROESSE	25
ERIC_IO_READER_UNTERSACHBEREI	CH_UNGUELTIG	ERIC_TRANSFER_ERR_BEGINDATENL	IEFERANT	25
ERIC_IO_READER_ZU_VIELE_ANHAE	NGE	ERIC_TRANSFER_ERR_BEGINTRANSP	ORTSCHLUESSEL	25
ERIC_IO_READER_ZU_VIELE_NUTZD	ATENBLOCK_ELEMENTE	ERIC_TRANSFER_ERR_CONNECTSER	VER	26
ERIC_IO_STEUERZEICHEN_IM_NDS	32	ERIC_TRANSFER_ERR_DATENTEILEN	DNOTFOUND	25
ERIC_IO_TESTHERSTELLERID_GESPE	RRT	ERIC_TRANSFER_ERR_DATENTEILFE	HLE	26
ERIC_IO_UEBERGABEPARAMETER_F	EHLEHAF	ERIC_TRANSFER_ERR_ENDDATENGR	OESSE	25
ERIC_IO_UNBEKANNT_DATENART	33	ERIC_TRANSFER_ERR_ENDDATENLIE	FERANT	25
ERIC_IO_UNGUELTIGE_UTF8_SEQUE	NZ	ERIC_TRANSFER_ERR_ENDSIGUSER	26	
ERIC_IO_UNGUELTIGE_ZEICHEN_IN_	PARAMETER	ERIC_TRANSFER_ERR_ENDTRANSP	ORTSCHLUESSEL	25
ERIC_IO_VERSIONSINFORMATIONEN	_NICHT_GEFUNDEN	ERIC_TRANSFER_ERR_NORESPONSE	26	
ERIC_OK	18	ERIC_TRANSFER_ERR_NOTENCRYPT	ED	26
ERIC_PRINT_ABBRUCH_DRUCKVORB	EREITUNG	ERIC_TRANSFER_ERR_OTHER	26	
ERIC_PRINT_ABBRUCH_GENERIERUN	G	ERIC_TRANSFER_ERR_PARAM	25	
ERIC_PRINT_AUSGABEZIEL_UNBEKA	NNT	ERIC_TRANSFER_ERR_PROXYAUTH	26	
ERIC_PRINT_DRUCKVORLAGE_NICH	T_GEFUNDEN	ERIC_TRANSFER_ERR_PROXYCONNE	CT	26
ERIC_PRINT_FUSSTEXT_ZU_LANG	35	ERIC_TRANSFER_ERR_PROXYPORT_I	NVALID	26
ERIC_PRINT_INITIALIZIERUNG_FEHL	ERHAF	ERIC_TRANSFER_ERR_SEND	26	
ERIC_PRINT_INTERNET_FEHLER	34	ERIC_TRANSFER_ERR_SEND_INIT	26	
ERIC_PRINT_PDFCALLBACK	35	ERIC_TRANSFER_ERR_TIMEOUT	26	
ERIC_PRINT_STEUERFALL_NICHT_UN	TERSTUETZT	ERIC_TRANSFER_ERR_XML_ENCODING	26	
	35	ERIC_TRANSFER_ERR_XML_NHEADE	R	26

ERIC_TRANSFER_ERR_XML_THEADER 25	ERIC_GLOBAL_EINSTELLUNG_NAME_UNGUELTIG eric_fehlercodes.h 24
ERIC_TRANSFER_ERR_XMLTAG_NICHT_GEFUNDEN 26	ERIC_GLOBAL_EINSTELLUNG_WERT_UNGUELTIG eric_fehlercodes.h 24
ERIC_TRANSFER_VORGANG_NICHT_UNTERSTUETZT 25	ERIC_GLOBAL_ERR_DEKODIEREN eric_fehlercodes.h 24
ERIC_FORTSCHRITTCALLBACK_ID_DRUCKEN eric_types.h 155	ERIC_GLOBAL_ERROR_XML_CREATE eric_fehlercodes.h 20
ERIC_FORTSCHRITTCALLBACK_ID_EINLESEN eric_types.h 154	ERIC_GLOBAL_ERSTE_SEITE_DRUCK_NICHT_UNTERSTUETZT eric_fehlercodes.h 22
ERIC_FORTSCHRITTCALLBACK_ID_SENDEN eric_types.h 155	ERIC_GLOBAL_EWAZ_LANDESKUERZEL_UNBEKANNT eric_fehlercodes.h 24
ERIC_FORTSCHRITTCALLBACK_ID_VALIDIEREN eric_types.h 155	ERIC_GLOBAL_EWAZ_UNGUELTIG eric_fehlercodes.h 24
ERIC_FORTSCHRITTCALLBACK_ID_VORBEREITEN eric_types.h 154	ERIC_GLOBAL_FEHLER_INITIALISIERUNG eric_fehlercodes.h 22
ERIC_GLOBAL_ARITHMETIKFEHLER eric_fehlercodes.h 20	ERIC_GLOBAL_FEHLERMELDUNG_NICHT_VORHANDEN eric_fehlercodes.h 19
ERIC_GLOBAL_BIC_FORMALER_FEHLER eric_fehlercodes.h 23	ERIC_GLOBAL_FUNKTION_NICHT_ERLAUBT eric_fehlercodes.h 19
ERIC_GLOBAL_BIC_LAENDERCODE_FEHLER eric_fehlercodes.h 23	ERIC_GLOBAL_FUNKTION_NICHT_UNTERSTUETZT eric_fehlercodes.h 24
ERIC_GLOBAL_BUFANR_UNBEKANNT eric_fehlercodes.h 20	ERIC_GLOBAL_HERSTELLER_ID_NICHT_ERLAUBT eric_fehlercodes.h 19
ERIC_GLOBAL_BUNDESLAENDER_UNEINHEITLICH eric_fehlercodes.h 24	ERIC_GLOBAL_HINWEISE eric_fehlercodes.h 19
ERIC_GLOBAL_CHECK_CORRUPTED_NIDS eric_fehlercodes.h 22	ERIC_GLOBAL_IBAN_FORMALER_FEHLER eric_fehlercodes.h 23
ERIC_GLOBAL_COMMONDATA_NICHT_VERFUEGBAR eric_fehlercodes.h 21	ERIC_GLOBAL_IBAN_LAENDERCODE_FEHLER eric_fehlercodes.h 23
ERIC_GLOBAL_DATEI_NICHT_GEFUNDEN eric_fehlercodes.h 19	ERIC_GLOBAL_IBAN_LANDESFORMAT_FEHLER eric_fehlercodes.h 23
ERIC_GLOBAL_DATEIZUGRIFF_VERWEIGERT eric_fehlercodes.h 21	ERIC_GLOBAL_IBAN_PRUEFZIFFER_FEHLER eric_fehlercodes.h 23
ERIC_GLOBAL_DATENARTVERSION_UNBEKANNT eric_fehlercodes.h 21	ERIC_GLOBAL_IDNUMMER_UNGUELTIG eric_fehlercodes.h 24
ERIC_GLOBAL_DATENARTVERSION_XML_INKONSISTENT eric_fehlercodes.h 21	ERIC_GLOBAL_ILLEGAL_STATE eric_fehlercodes.h 19
ERIC_GLOBAL_DATENSATZ_ZU_GROSS eric_fehlercodes.h 19	ERIC_GLOBAL_INKOMPATIBLE_VERSIONEN eric_fehlercodes.h 23
ERIC_GLOBAL_DRUCK_FUER_VERFAHREN_NICHT_ERLAUBT eric_fehlercodes.h 22	ERIC_GLOBAL_INTERNER_FEHLER eric_fehlercodes.h 20
ERIC_GLOBAL_ECHTFALL_NICHT_ERLAUBT eric_fehlercodes.h 19	ERIC_GLOBAL_KEINE_DATEN_VORHANDEN eric_fehlercodes.h 19

ERIC_GLOBAL_LANDESNUMMER_BUFA NR eric_fehlercodes.h 20	eric_fehlercodes.h 20
ERIC_GLOBAL_LANDESNUMMER_UNB EKANNT eric_fehlercodes.h 20	ERIC_GLOBAL_STEUERNUMMER_UNG UELTIG eric_fehlercodes.h 20
ERIC_GLOBAL_LOG_EXCEPTION eric_fehlercodes.h 21	ERIC_GLOBAL_TESTMERKER_UNGUEL TIG eric_fehlercodes.h 19
ERIC_GLOBAL_MEHRFACHAUFRUFE_N ICHT_UNTERSTUETZT eric_fehlercodes.h 23	ERIC_GLOBAL_TEXTPUFFERGROESSE_ FIX eric_fehlercodes.h 20
ERIC_GLOBAL_MEHRFACHE_INITIALISI ERUNG eric_fehlercodes.h 22	ERIC_GLOBAL_TRANSFERHANDLE eric_fehlercodes.h 23
ERIC_GLOBAL_NICHT_GENUEGEND_AR BEITSSPEICHER eric_fehlercodes.h 19	ERIC_GLOBAL_TRANSPORTSCHLUESSE L_NICHT_ERLAUBT eric_fehlercodes.h 21
ERIC_GLOBAL_NICHT_INITIALISIERT eric_fehlercodes.h 21	ERIC_GLOBAL_TRANSPORTSCHLUESSE L_TYP_FALSCH eric_fehlercodes.h 21
ERIC_GLOBAL_NO_VERSAND_IN_BETA _VERSION eric_fehlercodes.h 19	ERIC_GLOBAL_UNGUELTIGE_FLAG_KO MBINATION eric_fehlercodes.h 22
ERIC_GLOBAL_NULL_PARAMETER eric_fehlercodes.h 24	ERIC_GLOBAL_UNGUELTIGE_INSTANZ eric_fehlercodes.h 21
ERIC_GLOBAL_NUR_PORTALZERTIFIK AT_ERLAUBT eric_fehlercodes.h 20	ERIC_GLOBAL_UNGUELTIGE_PARAME TER_VERSION eric_fehlercodes.h 22
ERIC_GLOBAL_NUTZDATENHEADER_E MPFAENGER_NICHT_KORREKT eric_fehlercodes.h 25	ERIC_GLOBAL_UNGUELTIGER_PARAM ETER eric_fehlercodes.h 22
ERIC_GLOBAL_NUTZDATENHEADERVE RSIONEN_UNEINHEITLICH eric_fehlercodes.h 24	ERIC_GLOBAL_UNKNOWN eric_fehlercodes.h 18
ERIC_GLOBAL_NUTZDATENTICKETS_N ICHT_EINDEUTIG eric_fehlercodes.h 24	ERIC_GLOBAL_UNKNOWN_PARAMETE R_ERROR eric_fehlercodes.h 22
ERIC_GLOBAL_OEFFENTLICHER_SCHL UESSEL_UNGUELTIG eric_fehlercodes.h 21	ERIC_GLOBAL_UPDATE_NECESSARY eric_fehlercodes.h 24
ERIC_GLOBAL_PLUGININITIALISIERUN G eric_fehlercodes.h 23	ERIC_GLOBAL_UTI_COUNTRY_NOT_SU PPORTED eric_fehlercodes.h 23
ERIC_GLOBAL_PRUEF_FEHLER eric_fehlercodes.h 19	ERIC_GLOBAL_VERSAND_ART_NICHT_ UNTERSTUETZT eric_fehlercodes.h 22
ERIC_GLOBAL_PUFFER_UEBERLAUF eric_fehlercodes.h 21	ERIC_GLOBAL_VERSCHLUESSELUNGS_ PARAMETER_NICHT_ANGEGEBEN eric_fehlercodes.h 22
ERIC_GLOBAL_PUFFER_UNGLEICHER_I NSTANZ eric_fehlercodes.h 21	ERIC_GLOBAL_VERSCHLUESSELUNGS_ PARAMETER_NICHT_ERLAUBT eric_fehlercodes.h 20
ERIC_GLOBAL_PUFFER_ZUGRIFFSKONF LIKT eric_fehlercodes.h 20	ERIC_GLOBAL_VERSCHLUESSELUNGS VERFAHREN_NICHT_UNTERSTUETZT eric_fehlercodes.h 23
ERIC_GLOBAL_SEND_FLAG_MEHR_ALS _EINES eric_fehlercodes.h 22	ERIC_GLOBAL_VORSATZ_UNGUELTIG eric_fehlercodes.h 21
ERIC_GLOBAL_STEUERNUMMER_FALS CHE_LAENGE eric_fehlercodes.h 20	ERIC_GLOBAL_ZEITRAEUME_UNEINHE ITLICH eric_fehlercodes.h 24
ERIC_GLOBAL_STEUERNUMMER_NICH T_NUMERISCH	ERIC_GLOBAL_ZULASSUNGSNUMMER_ ZU_LANG eric_fehlercodes.h 23
	ERIC_IO_DATEI_INKORREKT

eric_fehlercodes.h 32	ERIC_IO_READER_STEUERZEICHEN_IM
ERIC_IO_DATENTEILENDNOTFOUND	_NUTZDATENHEADER
eric_fehlercodes.h 34	eric_fehlercodes.h 33
ERIC_IO_DATENTEILNOTFOUND	ERIC_IO_READER_STEUERZEICHEN_IM
eric_fehlercodes.h 34	_TRANSFERHEADER
ERIC_IO_FALSCHES_VERFAHREN	eric_fehlercodes.h 33
eric_fehlercodes.h 32	ERIC_IO_READER_STEUERZEICHEN_IN
ERIC_IO_FEHLER	DEN_NUTZDATEN
eric_fehlercodes.h 32	eric_fehlercodes.h 33
ERIC_IO_MASTERDATENSERVICE_NICH	ERIC_IO_READER_UNBEKANNTE_XML
T_VERFUEGBAR	ENTITY
eric_fehlercodes.h 32	eric_fehlercodes.h 34
ERIC_IO_NDS_GENERIERUNG_FEHLGES	ERIC_IO_READER_UNERWARTETE_ELE
CHLAGEN	MENTE
eric_fehlercodes.h 32	eric_fehlercodes.h 32
ERIC_IO_PARSE_FEHLER	ERIC_IO_READER_UNTERSACHBEREIC
eric_fehlercodes.h 32	H_UNGUELTIG
ERIC_IO_READER_ANHAENGE_ZU_GRO	eric_fehlercodes.h 33
SS	ERIC_IO_READER_ZU_VIELE_ANHAEN
eric_fehlercodes.h 34	GE
ERIC_IO_READER_ANHANG_ZU_GROSS	eric_fehlercodes.h 34
eric_fehlercodes.h 34	ERIC_IO_READER_ZU_VIELE_NUTZDAT
ERIC_IO_READER_ANHANG_ZU_KLEIN	ENBLOCK_ELEMENTE
eric_fehlercodes.h 34	eric_fehlercodes.h 33
ERIC_IO_READER_FALSCHES_ENCODIN	ERIC_IO_STEUERZEICHEN_IM_NDS
G	eric_fehlercodes.h 32
eric_fehlercodes.h 32	ERIC_IO_TESTHERSTELLERID_GESPERR
ERIC_IO_READER_FORMALE_FEHLER	T
eric_fehlercodes.h 32	eric_fehlercodes.h 34
ERIC_IO_READER_KEINE_RABEID	ERIC_IO_UEBERGABEPARAMETER_FEH
eric_fehlercodes.h 33	LERHAFT
ERIC_IO_READER_MEHRFACHE_NUTZD	eric_fehlercodes.h 34
ATEN_ELEMENTE	ERIC_IO_UNBEKANNTE_DATENART
eric_fehlercodes.h 32	eric_fehlercodes.h 33
ERIC_IO_READER_MEHRFACHE_NUTZD	ERIC_IO_UNGUELTIGE_UTF8_SEQUENZ
ATENBLOCK_ELEMENTE	eric_fehlercodes.h 34
eric_fehlercodes.h 32	ERIC_IO_UNGUELTIGE_ZEICHEN_IN_PA
ERIC_IO_READER_MEHRFACHE_STEUE	RAMETER
RFAELLE	eric_fehlercodes.h 34
eric_fehlercodes.h 32	ERIC_IO_VERSIONSINFORMATIONEN_N
ERIC_IO_READER_RABE_FEHLER	ICHT_GEFUNDEN
eric_fehlercodes.h 33	eric_fehlercodes.h 32
ERIC_IO_READER_RABE_REFERENZID_	ERIC_LOG_DEBUG
NICHT_ERLAUBT	eric_types.h 156
eric_fehlercodes.h 33	ERIC_LOG_ERROR
ERIC_IO_READER_RABE_REFERENZID_	eric_types.h 155
UNGUELTIG	ERIC_LOG_INFO
eric_fehlercodes.h 33	eric_types.h 156
ERIC_IO_READER_RABE_REFERENZIDS	eric_log_level_t
_NICHT_EINDEUTIG	eric_types.h 155
eric_fehlercodes.h 34	ERIC_LOG_TRACE
ERIC_IO_READER_RABE_VERIFIKATIO	eric_types.h 156
NSID_UNGUELTIG	ERIC_LOG_WARN
eric_fehlercodes.h 33	eric_types.h 156
ERIC_IO_READER_RABEID_UNGUELTIG	ERIC_MAJOR_VERSION
eric_fehlercodes.h 33	ericversion.h 116
ERIC_IO_READER_SCHEMA_VALIDIERU	ERIC_MAX_LAENGE_FUSSTEXT
NGSFEHLER	ericdef.h 159
eric_fehlercodes.h 34	ERIC_MINOR_VERSION
	ericversion.h 116

ERIC_OK	eric_fehlercodes.h 18	ERIC_TRANSFER_EID_ZERTIFIKATFEHLER	eric_fehlercodes.h 27
ERIC_PATCH_VERSION	ericversion.h 116	ERIC_TRANSFER_ERR_BEGINDATENGR	OESSE
ERIC_PRINT_ABBRUCH_DRUCKVORBEREITUNG	eric_fehlercodes.h 35	ERIC_TRANSFER_ERR_BEGINDATENLIEFERANT	eric_fehlercodes.h 25
ERIC_PRINT_ABBRUCH_GENERIERUNG	eric_fehlercodes.h 35	ERIC_TRANSFER_ERR_BEGINTRANSPORTSCHLUESSEL	eric_fehlercodes.h 25
ERIC_PRINT_AUSGABEZIEL_UNBEKANNT	eric_fehlercodes.h 35	ERIC_TRANSFER_ERR_CONNECTSERVER	eric_fehlercodes.h 26
ERIC_PRINT_DRUCKVORLAGE_NICHT_GEFUNDEN	eric_fehlercodes.h 34	ERIC_TRANSFER_ERR_DATENTEILENDNOTFOUND	eric_fehlercodes.h 25
ERIC_PRINT_FUSSTEXT_ZU_LANG	eric_fehlercodes.h 35	ERIC_TRANSFER_ERR_DATENTEILFEHLER	eric_fehlercodes.h 26
ERIC_PRINT_INITIALIZIERUNG_FEHLERHAFT	eric_fehlercodes.h 35	ERIC_TRANSFER_ERR_ENDDATENGROESSE	eric_fehlercodes.h 25
ERIC_PRINT_INTERNER_FEHLER	eric_fehlercodes.h 34	ERIC_TRANSFER_ERR_ENDDATENLIEFERANT	eric_fehlercodes.h 25
ERIC_PRINT_PDFCALLBACK	eric_fehlercodes.h 35	ERIC_TRANSFER_ERR_ENDSIGUSER	eric_fehlercodes.h 26
ERIC_PRINT_STEUERFALL_NICHT_UNTERSTUETZT	eric_fehlercodes.h 35	ERIC_TRANSFER_ERR_ENDTRANSPORTSCHLUESSEL	eric_fehlercodes.h 25
ERIC_PRINT_UNGUELTIGER_DATEI_PFAD	eric_fehlercodes.h 35	ERIC_TRANSFER_ERR_NORESPONSE	eric_fehlercodes.h 26
ERIC_PRUEFE_HINWEISE	eric_types.h 155	ERIC_TRANSFER_ERR_NOTENCRYPTED	eric_fehlercodes.h 26
ERIC_SENDE	eric_types.h 155	ERIC_TRANSFER_ERR_OTHER	eric_fehlercodes.h 26
ERIC_TESTMERKER_CLEARINGSTELLE	ericdef.h 159	ERIC_TRANSFER_ERR_PARAM	eric_fehlercodes.h 25
ERIC_TESTMERKER_ECC	ericdef.h 159	ERIC_TRANSFER_ERR_PROXYAUTH	eric_fehlercodes.h 26
ERIC_TRANSFER_COM_ERROR	eric_fehlercodes.h 25	ERIC_TRANSFER_ERR_PROXYCONNECT	eric_fehlercodes.h 26
ERIC_TRANSFER_EID_CLIENTFEHLER	eric_fehlercodes.h 27	ERIC_TRANSFER_ERR_PROXYPORT_INVALID	eric_fehlercodes.h 26
ERIC_TRANSFER_EID_FEHLENDEFELDER	eric_fehlercodes.h 27	ERIC_TRANSFER_ERR_SEND	eric_fehlercodes.h 26
ERIC_TRANSFER_EID_IDENTIFIKATIONABGEBROCHEN	eric_fehlercodes.h 27	ERIC_TRANSFER_ERR_SEND_INIT	eric_fehlercodes.h 26
ERIC_TRANSFER_EID_IDNRNICHTEINDUTIG	eric_fehlercodes.h 27	ERIC_TRANSFER_ERR_TIMEOUT	eric_fehlercodes.h 26
ERIC_TRANSFER_EID_KEINCLIENT	eric_fehlercodes.h 27	ERIC_TRANSFER_ERR_XML_ENCODING	eric_fehlercodes.h 26
ERIC_TRANSFER_EID_KEINKONTO	eric_fehlercodes.h 27	ERIC_TRANSFER_ERR_XML_NHEADER	eric_fehlercodes.h 26
ERIC_TRANSFER_EID_NPABLOCKIERT	eric_fehlercodes.h 27	ERIC_TRANSFER_ERR_XML_HEADER	eric_fehlercodes.h 25
ERIC_TRANSFER_EID_SERVERFEHLER	eric_fehlercodes.h 27		

ERIC_TRANSFER_ERR_XMLTAG_NICHT_GEFUNDEN
 eric_fehlercodes.h 26
 ERIC_TRANSFER_VORGANG_NICHT_UNTERSTUETZT
 eric_fehlercodes.h 25
 eric_types.h 150
 byteChar 151
 eric_bearbeitung_flag_t 155
 ERIC_DRUCKE 155
 ERIC_FORTSCHRITTCALLBACK_ID_DRUCKEN 155
 ERIC_FORTSCHRITTCALLBACK_ID_EINLESEN 154
 ERIC_FORTSCHRITTCALLBACK_ID_SENDEN 155
 ERIC_FORTSCHRITTCALLBACK_ID_VALIDIEREN 155
 ERIC_FORTSCHRITTCALLBACK_ID_VALIDIEREN_ORBEREITEN 154
 ERIC_LOG_DEBUG 156
 ERIC_LOG_ERROR 155
 ERIC_LOG_INFO 156
 eric_log_level_t 155
 ERIC_LOG_TRACE 156
 ERIC_LOG_WARN 156
 ERIC_PRUEFE_HINWEISE 155
 ERIC_SENDE 155
 ERIC_VALIDIERE 155
 ERIC_VALIDIERE_OHNE_FREIGABEDATUM 155
 EricFortschrittCallback 151
 EricInstanzHandle 152
 EricLogCallback 152
 EricPdfCallback 153
 EricRueckgabepufferHandle 153
 EricTransferHandle 154
 EricZertifikatHandle 154
 ERIC_VALIDIERE
 eric_types.h 155
 ERIC_VALIDIERE_OHNE_FREIGABEDATUM
 eric_types.h 155
 eric_verschluesselungs_parameter_t 8
 pin 8
 version 8
 zertifikatHandle 9
 eric_zertifikat_parameter_t 10
 abteilung 11
 adresse 11
 beschreibung 11
 email 11
 land 11
 name 11
 organisation 12
 ort 12
 version 12
 ericapi.h 36
 EricBearbeiteVorgang 40
 EricBeende 44
 EricChangePassword 44
 EricCheckXML 45
 EricCloseHandleToCertificate 45
 EricCreateKey 46
 EricCreateTH 47
 EricCreateUUID 49
 EricDekodiereDaten 49
 EricEinstellungAlleZuruecksetzen 50
 EricEinstellungLesen 50
 EricEinstellungSetzen 51
 EricEinstellungZuruecksetzen 51
 EricEntladePlugins 52
 EricFormatEWaz 52
 EricFormatStNr 52
 EricGetAuswahlListen 53
 EricGetErrorMessageFromXMLAnswer 54
 EricGetHandleToCertificate 55
 EricGetPinStatus 57
 EricGetPublicKey 58
 EricHoleFehlerText 58
 EricHoleFinanzaemter 59
 EricHoleFinanzamtLandNummern 59
 EricHoleFinanzamtsdaten 60
 EricHoleTestfinanzaemter 61
 EricHoleZertifikatEigenschaften 61
 EricHoleZertifikatFingerabdruck 62
 EricInitialisiere 63
 EricMakeElsterEWaz 64
 EricMakeElsterStnr 64
 EricPruefeBIC 65
 EricPruefeBuFaNummer 65
 EricPruefeEWaz 66
 EricPruefeIBAN 66
 EricPruefeIdentifikationsMerkmal 67
 EricPruefeSteuernummer 67
 EricPruefeWIdNr 67
 EricPruefeZertifikatPin 68
 EricRegistriereFortschrittCallback 69
 EricRegistriereGlobalenFortschrittCallback 70
 EricRegistriereLogCallback 70
 EricRueckgabepufferErzeugen 71
 EricRueckgabepufferFreigeben 72
 EricRueckgabepufferInhalt 72
 EricRueckgabepufferLaenge 72
 EricSystemCheck 73
 EricVersion 73
 ERICAPI_IMPORT
 ericapiExport.h 157
 ericapiExport.h 157
 ERICAPI_IMPORT 157
 EricBearbeiteVorgang
 ericapi.h 40
 EricBeende
 ericapi.h 44
 EricChangePassword
 ericapi.h 44
 EricCheckXML
 ericapi.h 45

EricCloseHandleToCertificate	ericapi.h 45	EricLogCallback	eric_types.h 152
EricCreateKey	ericapi.h 46	EricMakeElsterEWaz	ericapi.h 64
EricCreateTH	ericapi.h 47	EricMakeElsterStnr	ericapi.h 64
EricCreateUUID	ericapi.h 49	ericmtapi.h 75	
ericdef.h 158		EricMtBearbeiteVorgang	79
ERIC_MAX_LAENGE_FUSSTEXT	159	EricMtChangePassword	83
ERIC_TESTMERKER_CLEARINGSTELL	E 159	EricMtCheckXML	84
ERIC_TESTMERKER_ECC	159	EricMtCloseHandleToCertificate	85
EURO	159	EricMtCreateKey	85
EricDekodiereDaten	ericapi.h 49	EricMtCreateTH	86
EricEinstellungAlleZuruecksetzen	ericapi.h 50	EricMtCreateUUID	88
EricEinstellungLesen	ericapi.h 50	EricMtDekodiereDaten	88
EricEinstellungSetzen	ericapi.h 51	EricMtEinstellungAlleZuruecksetzen	89
EricEinstellungZuruecksetzen	ericapi.h 51	EricMtEinstellungLesen	89
EricEntladePlugins	ericapi.h 52	EricMtEinstellungSetzen	90
EricFormatEWaz	ericapi.h 52	EricMtEinstellungZuruecksetzen	90
EricFormatStNr	ericapi.h 52	EricMtEntladePlugins	91
EricFortschrittCallback	eric_types.h 151	EricMtFormatEWaz	91
EricGetAuswahlListen	ericapi.h 53	EricMtFormatStNr	92
EricGetErrormessagesFromXMLAnswer	ericapi.h 54	EricMtGetAuswahlListen	92
EricGetHandleToCertificate	ericapi.h 55	EricMtGetErrormessagesFromXMLAnswer	93
EricGetPinStatus	ericapi.h 57	EricMtGetHandleToCertificate	94
EricGetPublicKey	ericapi.h 58	EricMtGetPinStatus	97
EricHoleFehlerText	ericapi.h 58	EricMtGetPublicKey	97
EricHoleFinanzaemter	ericapi.h 59	EricMtHoleFehlerText	98
EricHoleFinanzamtLandNummern	ericapi.h 59	EricMtHoleFinanzaemter	98
EricHoleFinanzamtsdaten	ericapi.h 60	EricMtHoleFinanzamtLandNummern	99
EricHoleTestfinanzaemter	ericapi.h 61	EricMtHoleFinanzamtsdaten	100
EricHoleZertifikatEigenschaften	ericapi.h 61	EricMtHoleTestfinanzaemter	100
EricHoleZertifikatFingerabdruck	ericapi.h 62	EricMtHoleZertifikatEigenschaften	101
EricInitialisiere	ericapi.h 63	EricMtHoleZertifikatFingerabdruck	102
EricInstanzHandle	eric_types.h 152	EricMtInstanzErzeugen	103
		EricMtInstanzFreigeben	103
		EricMtMakeElsterEWaz	104
		EricMtMakeElsterStnr	104
		EricMtPruefeBIC	105
		EricMtPruefeBuFaNummer	105
		EricMtPruefeEWaz	106
		EricMtPruefeIBAN	106
		EricMtPruefeIdentifikationsMerkmal	107
		EricMtPruefeSteuernummer	107
		EricMtPruefeWIdNr	108
		EricMtPruefeZertifikatPin	108
		EricMtRegistriereFortschrittCallback	110
		EricMtRegistriereGlobalenFortschrittCallba	ck 110
		EricMtRegistriereLogCallback	111
		EricMtRueckgabepufferErzeugen	112
		EricMtRueckgabepufferFreigeben	112
		EricMtRueckgabepufferInhalt	113
		EricMtRueckgabepufferLaenge	113
		EricMtSystemCheck	114
		EricMtVersion	114
		EricMtBearbeiteVorgang	ericmtapi.h 79
		EricMtChangePassword	

ericmtapi.h 83
 EricMtCheckXML
 ericmtapi.h 84
 EricMtCloseHandleToCertificate
 ericmtapi.h 85
 EricMtCreateKey
 ericmtapi.h 85
 EricMtCreateTH
 ericmtapi.h 86
 EricMtCreateUUID
 ericmtapi.h 88
 EricMtDekodiereDaten
 ericmtapi.h 88
 EricMtEinstellungAlleZuruecksetzen
 ericmtapi.h 89
 EricMtEinstellungLesen
 ericmtapi.h 89
 EricMtEinstellungSetzen
 ericmtapi.h 90
 EricMtEinstellungZuruecksetzen
 ericmtapi.h 90
 EricMtEntladePlugins
 ericmtapi.h 91
 EricMtFormatEWaz
 ericmtapi.h 91
 EricMtFormatStNr
 ericmtapi.h 92
 EricMtGetAuswahlListen
 ericmtapi.h 92
 EricMtGetErrormessagesFromXMLAnswer
 ericmtapi.h 93
 EricMtGetHandleToCertificate
 ericmtapi.h 94
 EricMtGetPinStatus
 ericmtapi.h 97
 EricMtGetPublicKey
 ericmtapi.h 97
 EricMtHoleFehlerText
 ericmtapi.h 98
 EricMtHoleFinanzaemter
 ericmtapi.h 98
 EricMtHoleFinanzamtLandNummern
 ericmtapi.h 99
 EricMtHoleFinanzamtsdaten
 ericmtapi.h 100
 EricMtHoleTestfinanzaemter
 ericmtapi.h 100
 EricMtHoleZertifikatEigenschaften
 ericmtapi.h 101
 EricMtHoleZertifikatFingerabdruck
 ericmtapi.h 102
 EricMtInstanzErzeugen
 ericmtapi.h 103
 EricMtInstanzFreigeben
 ericmtapi.h 103
 EricMtMakeElsterEWaz
 ericmtapi.h 104
 EricMtMakeElsterStnr
 ericmtapi.h 104
 EricMtPruefeBIC

ericmtapi.h 105
 EricMtPruefeBuFaNummer
 ericmtapi.h 105
 EricMtPruefeEWaz
 ericmtapi.h 106
 EricMtPruefeIBAN
 ericmtapi.h 106
 EricMtPruefeIdentifikationsMerkmal
 ericmtapi.h 107
 EricMtPruefeSteuernummer
 ericmtapi.h 107
 EricMtPruefeWIdNr
 ericmtapi.h 108
 EricMtPruefeZertifikatPin
 ericmtapi.h 108
 EricMtRegistriereFortschrittCallback
 ericmtapi.h 110
 EricMtRegistriereGlobalenFortschrittCallback
 ericmtapi.h 110
 EricMtRegistriereLogCallback
 ericmtapi.h 111
 EricMtRueckgabepufferErzeugen
 ericmtapi.h 112
 EricMtRueckgabepufferFreigeben
 ericmtapi.h 112
 EricMtRueckgabepufferInhalt
 ericmtapi.h 113
 EricMtRueckgabepufferLaenge
 ericmtapi.h 113
 EricMtSystemCheck
 ericmtapi.h 114
 EricMtVersion
 ericmtapi.h 114
 EricPdfCallback
 eric_types.h 153
 EricPruefeBIC
 ericapi.h 65
 EricPruefeBuFaNummer
 ericapi.h 65
 EricPruefeEWaz
 ericapi.h 66
 EricPruefeIBAN
 ericapi.h 66
 EricPruefeIdentifikationsMerkmal
 ericapi.h 67
 EricPruefeSteuernummer
 ericapi.h 67
 EricPruefeWIdNr
 ericapi.h 67
 EricPruefeZertifikatPin
 ericapi.h 68
 EricRegistriereFortschrittCallback
 ericapi.h 69
 EricRegistriereGlobalenFortschrittCallback
 ericapi.h 70
 EricRegistriereLogCallback
 ericapi.h 70
 EricRueckgabepufferErzeugen
 ericapi.h 71
 EricRueckgabepufferFreigeben

- ericapi.h 72
- EricRueckgabepufferHandle
 - eric_types.h 153
- EricRueckgabepufferInhalt
 - ericapi.h 72
- EricRueckgabepufferLaenge
 - ericapi.h 72
- EricSystemCheck
 - ericapi.h 73
- erictoolkit.h 117
 - ETKAPI_DECL 118
 - EtkHoleDateiVersion 118
 - EtkHoleProduktVersion 118
 - EtkPruefeBIC 118
 - EtkPruefeBuFaNummer 118
 - EtkPruefeEWaz 119
 - EtkPruefeIBAN 119
 - EtkPruefeIdentifikationsMerkmal 120
 - EtkPruefeSteuernummer 120
 - EtkPruefeWIdNr 120
- EricTransferHandle
 - eric_types.h 154
- EricVersion
 - ericapi.h 73
- ericversion.h 116
 - ERIC_MAJOR_VERSION 116
 - ERIC_MINOR_VERSION 116
 - ERIC_PATCH_VERSION 116
- EricZertifikatHandle
 - eric_types.h 154
- ETKAPI_DECL
 - erictoolkit.h 118
- EtkHoleDateiVersion
 - erictoolkit.h 118
- EtkHoleProduktVersion
 - erictoolkit.h 118
- EtkPruefeBIC
 - erictoolkit.h 118
- EtkPruefeBuFaNummer
 - erictoolkit.h 118
- EtkPruefeEWaz
 - erictoolkit.h 119
- EtkPruefeIBAN
 - erictoolkit.h 119
- EtkPruefeIdentifikationsMerkmal
 - erictoolkit.h 120
- EtkPruefeSteuernummer
 - erictoolkit.h 120
- EtkPruefeWIdNr
 - erictoolkit.h 120
- EURO
 - ericdef.h 159
- fussText
 - eric_druck_parameter_t 6
- HAS_FUTIME
 - platform.h 161
- I64
 - platform.h 161
- land
 - eric_zertifikat_parameter_t 11
- name
 - eric_zertifikat_parameter_t 11
- organisation
 - eric_zertifikat_parameter_t 12
- ort
 - eric_zertifikat_parameter_t 12
- otto.h 122
 - OttoDatenAbholen 124
 - OttoEmpfangBeenden 126
 - OttoEmpfangBeginnen 126
 - OttoEmpfangBeginnenAbholzertifikat 127
 - OttoEmpfangFortsetzen 128
 - OttoHoleFehlertext 128
 - OttoInstanzErzeugen 128
 - OttoInstanzFreigeben 129
 - OttoProxyKonfigurationSetzen 129
 - OttoPruefsummeAktualisieren 130
 - OttoPruefsummeErzeugen 130
 - OttoPruefsummeFreigeben 131
 - OttoPruefsummeSignieren 131
 - OttoRueckgabepufferErzeugen 132
 - OttoRueckgabepufferFreigeben 132
 - OttoRueckgabepufferGroesse 132
 - OttoRueckgabepufferInhalt 133
 - OttoVersandAbschliessen 133
 - OttoVersandBeenden 134
 - OttoVersandBeginnen 134
 - OttoVersandFortsetzen 135
 - OttoVersion 135
 - OttoZertifikatOeffnen 136
 - OttoZertifikatSchliessen 137
- OTTO_DEKOMPRESSION_FEHLGESCHL
 - AGEN
 - otto_statuscode.h 145
- OTTO_EMPFANG_VORZEITIG_BEENDET
 - otto_statuscode.h 141
- OTTO_ENTSCHLUESSELN_FEHLGESCHL
 - AGEN
 - otto_statuscode.h 145
- OTTO_ESIGNER_ASN1_NO_CONTENT_D
 - ATA
 - otto_statuscode.h 145
- OTTO_ESIGNER_ASN1_NO_ENVELOPED
 - _DATA
 - otto_statuscode.h 144
- OTTO_ESIGNER_ASN1_READ_BUFFER_
 - TOO_SMALL
 - otto_statuscode.h 144
- OTTO_ESIGNER_ASN1_READ_DATA_IN
 - COMPLETE
 - otto_statuscode.h 144
- OTTO_ESIGNER_BUSY
 - otto_statuscode.h 142
- OTTO_ESIGNER_DATA_NOT_INITIALIZE
 - D
 - otto_statuscode.h 144
- OTTO_ESIGNER_DECRYPT
 - otto_statuscode.h 142
- OTTO_ESIGNER_ENCODE_ERROR
 - otto_statuscode.h 142

OTTO_ESIGNER_ENCODE_UNKNOWN	OTTO_ESIGNER_PSE_PATH
otto_statuscode.h 142	otto_statuscode.h 143
OTTO_ESIGNER_ENCRYPT	OTTO_ESIGNER_SC_ENC_KEY
otto_statuscode.h 142	otto_statuscode.h 144
OTTO_ESIGNER_ESICL_EXCEPTION	OTTO_ESIGNER_SC_INIT_FAILED
otto_statuscode.h 142	otto_statuscode.h 144
OTTO_ESIGNER_INKOMPATIBEL	OTTO_ESIGNER_SC_NO_APPLET
otto_statuscode.h 141	otto_statuscode.h 144
OTTO_ESIGNER_INVALID_HANDLE	OTTO_ESIGNER_SC_NO_ENC_CERT
otto_statuscode.h 142	otto_statuscode.h 144
OTTO_ESIGNER_LOAD_DLL	OTTO_ESIGNER_SC_NO_SIG_CERT
otto_statuscode.h 142	otto_statuscode.h 144
OTTO_ESIGNER_MAX_SESSION	OTTO_ESIGNER_SC_SESSION
otto_statuscode.h 142	otto_statuscode.h 144
OTTO_ESIGNER_NICHT_GELADEN	OTTO_ESIGNER_SC_SIG_KEY
otto_statuscode.h 141	otto_statuscode.h 144
OTTO_ESIGNER_NO_SERVICE	OTTO_ESIGNER_SC_SLOT_EMPTY
otto_statuscode.h 142	otto_statuscode.h 144
OTTO_ESIGNER_NO_SIG_ENC_KEY	OTTO_ESIGNER_TOKEN_TYPE_MISMATCH
otto_statuscode.h 142	otto_statuscode.h 144
OTTO_ESIGNER_OUT_OF_MEM	OTTO_ESIGNER_USER_CANCEL
otto_statuscode.h 142	otto_statuscode.h 144
OTTO_ESIGNER_P11_ENC_KEY	OTTO_ESIGNER_VERALTET
otto_statuscode.h 142	otto_statuscode.h 141
OTTO_ESIGNER_P11_ENGINE_LOADED	OTTO_ESIGNER_VERIFY_CERT_CHAIN
otto_statuscode.h 142	otto_statuscode.h 144
OTTO_ESIGNER_P11_INIT_FAILED	OTTO_FUNKTION_NICHT_UNTERSTUETZT
otto_statuscode.h 142	otto_statuscode.h 141
OTTO_ESIGNER_P11_NO_ENC_CERT	OTTO_INIDATEI_LESEFEHLER
otto_statuscode.h 143	otto_statuscode.h 145
OTTO_ESIGNER_P11_NO_SIG_CERT	OTTO_INSTANZ_UNTEROBJEKTE_NICHT_FREIGEgeben
otto_statuscode.h 143	otto_statuscode.h 141
OTTO_ESIGNER_P11_SIG_KEY	OTTO_INSTANZEN_INKONSISTENT
otto_statuscode.h 143	otto_statuscode.h 141
OTTO_ESIGNER_P11_SLOT_EMPTY	OTTO_INTERNER_FEHLER
otto_statuscode.h 143	otto_statuscode.h 139
OTTO_ESIGNER_P12_CREATE	OTTO_LOG_FEHLER
otto_statuscode.h 143	otto_statuscode.h 141
OTTO_ESIGNER_P12_DECODE	OTTO_MEHRFACHAUFRUFE_NICHT_UNTERSTUETZT
otto_statuscode.h 143	otto_statuscode.h 140
OTTO_ESIGNER_P12_ENC_KEY	OTTO_NICHT_GENUEGEND_ARBEITSSPEICHER
otto_statuscode.h 143	otto_statuscode.h 145
OTTO_ESIGNER_P12_NO_ENC_CERT	OTTO_NPA_ZERTIFIKATFEHLER
otto_statuscode.h 143	otto_statuscode.h 139
OTTO_ESIGNER_P12_NO_SIG_CERT	OTTO_OK
otto_statuscode.h 143	otto_statuscode.h 139
OTTO_ESIGNER_P12_READ	OTTO_PROXY_AUTHSCHEMA
otto_statuscode.h 143	otto_statuscode.h 141
OTTO_ESIGNER_P12_SIG_KEY	OTTO_PROXY_PORT
otto_statuscode.h 143	otto_statuscode.h 141
OTTO_ESIGNER_P7_DECODE	OTTO_PROXY_URL
otto_statuscode.h 143	otto_statuscode.h 141
OTTO_ESIGNER_P7_READ	OTTO_PRUEFSUMME_FINALISIERT
otto_statuscode.h 143	otto_statuscode.h 141
OTTO_ESIGNER_P7_RECIPIENT	OTTO_SIGNIEREN_FEHLGESCHLAGEN
otto_statuscode.h 143	
OTTO_ESIGNER_PIN_LOCKED	
otto_statuscode.h 143	
OTTO_ESIGNER_PIN_WRONG	
otto_statuscode.h 143	

otto_statuscode.h 145	OTTO_ESIGNER_P7_READ 143
otto_statuscode.h 138	OTTO_ESIGNER_P7_RECIPIENT 143
OTTO_DEKOMPRESSION_FEHLGESCHLAGEN 145	OTTO_ESIGNER_PIN_LOCKED 143
OTTO_EMPFANG_VORZEITIG_BEENDET 141	OTTO_ESIGNER_PIN_WRONG 143
OTTO_ENTSCHLUESSELN_FEHLGESCHLAGEN 145	OTTO_ESIGNER_PSE_PATH 143
OTTO_ESIGNER_ASN1_NO_CONTENT_DATA 145	OTTO_ESIGNER_SC_ENC_KEY 144
OTTO_ESIGNER_ASN1_NO_ENVELOPED_DATA 144	OTTO_ESIGNER_SC_INIT_FAILED 144
OTTO_ESIGNER_ASN1_READ_BUFFER_TOO_SMALL 144	OTTO_ESIGNER_SC_NO_APPLET 144
OTTO_ESIGNER_ASN1_READ_DATA_INCOMPLETE 144	OTTO_ESIGNER_SC_NO_ENC_CERT 144
OTTO_ESIGNER_BUSY 142	OTTO_ESIGNER_SC_NO_SIG_CERT 144
OTTO_ESIGNER_DATA_NOT_INITIALIZED 144	OTTO_ESIGNER_SC_SESSION 144
OTTO_ESIGNER_DECRYPT 142	OTTO_ESIGNER_SC_SIG_KEY 144
OTTO_ESIGNER_ENCODE_ERROR 142	OTTO_ESIGNER_SC_SLOT_EMPTY 144
OTTO_ESIGNER_ENCODE_UNKNOWN 142	OTTO_ESIGNER_TOKEN_TYPE_MISMATCH 144
OTTO_ESIGNER_ENCRYPT 142	OTTO_ESIGNER_USER_CANCEL 144
OTTO_ESIGNER_ESICL_EXCEPTION 142	OTTO_ESIGNER_VERALTET 141
OTTO_ESIGNER_INKOMPATIBEL 141	OTTO_ESIGNER_VERIFY_CERT_CHAIN 144
OTTO_ESIGNER_INVALID_HANDLE 142	OTTO_FUNKTION_NICHT_UNTERSTUETZT 141
OTTO_ESIGNER_LOAD_DLL 142	OTTO_INIDATEI_LESEFEHLER 145
OTTO_ESIGNER_MAX_SESSION 142	OTTO_INSTANZ_UNTEROBJEKTE_NICHT_FREIGEgeben 141
OTTO_ESIGNER_NICHT_GELADEN 141	OTTO_INSTANZEN_INKONSISTENT 141
OTTO_ESIGNER_NO_SERVICE 142	OTTO_INTERNER_FEHLER 139
OTTO_ESIGNER_NO_SIG_ENC_KEY 142	OTTO_LOG_FEHLER 141
OTTO_ESIGNER_OUT_OF_MEM 142	OTTO_MEHRFACHAUFRUFE_NICHT_UNTERSTUETZT 140
OTTO_ESIGNER_P11_ENC_KEY 142	OTTO_NICHT_GENUEGEND_ARBEITS_SPEICHER 145
OTTO_ESIGNER_P11_ENGINE_LOADED 142	OTTO_NPA_ZERTIFIKATFEHLER 139
OTTO_ESIGNER_P11_INIT_FAILED 142	OTTO_OK 139
OTTO_ESIGNER_P11_NO_ENC_CERT 143	OTTO_PROXY_AUTHSCHEMA 141
OTTO_ESIGNER_P11_NO_SIG_CERT 143	OTTO_PROXY_PORT 141
OTTO_ESIGNER_P11_SIG_KEY 143	OTTO_PROXY_URL 141
OTTO_ESIGNER_P11_SLOT_EMPTY 143	OTTO_PRUEFSUMME_FINALISIERT 141
OTTO_ESIGNER_P12_CREATE 143	OTTO_SIGNIEREN_FEHLGESCHLAGE N 145
OTTO_ESIGNER_P12_DECODE 143	OTTO_TRANSFER_CONNECTPROXY 140
OTTO_ESIGNER_P12_ENC_KEY 143	OTTO_TRANSFER_CONNECTSERVER 140
OTTO_ESIGNER_P12_NO_ENC_CERT 143	OTTO_TRANSFER_DECODING 140
OTTO_ESIGNER_P12_NO_SIG_CERT 143	OTTO_TRANSFER_EID_CLIENTFEHLER 140
OTTO_ESIGNER_P12_READ 143	OTTO_TRANSFER_EID_KEINCLIENT 140
OTTO_ESIGNER_P12_SIG_KEY 143	OTTO_TRANSFER_EID_KEINKONTO 140
OTTO_ESIGNER_P7_DECODE 143	OTTO_TRANSFER_EID_NPABLOCKIERT 140
	OTTO_TRANSFER_EID_ZERTIFIKATFEHLER 140

OTTO_TRANSFER_FEHLER 139
 OTTO_TRANSFER_INIT 139
 OTTO_TRANSFER_NOT_FOUND 140
 OTTO_TRANSFER_PROXYAUTH 140
 OTTO_TRANSFER_SERVER_FEHLER 140
 OTTO_TRANSFER_TIMEOUT 140
 OTTO_TRANSFER_UNAUTHORIZED 140
 OTTO_UNBEKANNTER_FEHLER 139
 OTTO_UNGUELTIGE_HERSTELLERID 141
 OTTO_UNGUELTIGER_PARAMETER 140
 OTTO_UNGUELTIGES_HANDLE 140
 OTTO_VERSAND_ABGESCHLOSSEN 142
 OTTO_VERSAND_GERINGE_DATENMENGE 141
 OTTO_VERSAND_ZU_GROSSE_DATENMENGE 142
 OTTO_ZERTIFIKAT_DEFEKT 145
 OTTO_ZERTIFIKAT_FINGERABDRUCK_FEHLER 145
 OTTO_ZERTIFIKAT_LESEFEHLER 145
 OTTO_ZERTIFIKAT_NICHT_ERKANNT 141
 OTTO_ZERTIFIKAT_PFAD_FALSCH 141
 OTTO_ZERTIFIKAT_PIN_FALSCH 141
 OttoStatusCode 139
 OTTO_TRANSFER_CONNECTPROXY otto_statuscode.h 140
 OTTO_TRANSFER_CONNECTSERVER otto_statuscode.h 140
 OTTO_TRANSFER_DECODING otto_statuscode.h 140
 OTTO_TRANSFER_EID_CLIENTFEHLER otto_statuscode.h 140
 OTTO_TRANSFER_EID_KEINCLIENT otto_statuscode.h 140
 OTTO_TRANSFER_EID_KEINKONTO otto_statuscode.h 140
 OTTO_TRANSFER_EID_NPABLOCKIERT otto_statuscode.h 140
 OTTO_TRANSFER_EID_ZERTIFIKATFEHLER otto_statuscode.h 140
 OTTO_TRANSFER_FEHLER otto_statuscode.h 139
 OTTO_TRANSFER_INIT otto_statuscode.h 139
 OTTO_TRANSFER_NOT_FOUND otto_statuscode.h 140
 OTTO_TRANSFER_PROXYAUTH otto_statuscode.h 140
 OTTO_TRANSFER_SERVER_FEHLER otto_statuscode.h 140
 OTTO_TRANSFER_TIMEOUT otto_statuscode.h 140
 OTTO_TRANSFER_UNAUTHORIZED otto_statuscode.h 140
 otto_types.h 146
 OttoEmpfangHandle 147
 OttoInstanzHandle 147
 OTTOLOG_DEBUGMELDUNGEN 149
 OTTOLOG_FEHLERMELDUNGEN 149
 OTTOLOG_INFORMATIONEN 149
 OTTOLOG_WARNUNGEN 149
 OttoLogCallback 147
 OttoLogEbene 148
 OttoPruefsummeHandle 147
 OttoRueckgabepufferHandle 148
 OttoVersandHandle 148
 OttoZertifikatHandle 148
 OTTO_UNBEKANNTER_FEHLER otto_statuscode.h 139
 OTTO_UNGUELTIGE_HERSTELLERID otto_statuscode.h 141
 OTTO_UNGUELTIGER_PARAMETER otto_statuscode.h 140
 OTTO_UNGUELTIGES_HANDLE otto_statuscode.h 140
 OTTO_VERSAND_ABGESCHLOSSEN otto_statuscode.h 142
 OTTO_VERSAND_GERINGE_DATENMENGE otto_statuscode.h 141
 OTTO_VERSAND_ZU_GROSSE_DATENMENGE otto_statuscode.h 142
 OTTO_ZERTIFIKAT_DEFEKT otto_statuscode.h 145
 OTTO_ZERTIFIKAT_FINGERABDRUCK_FEHLER otto_statuscode.h 145
 OTTO_ZERTIFIKAT_LESEFEHLER otto_statuscode.h 145
 OTTO_ZERTIFIKAT_NICHT_ERKANNT otto_statuscode.h 141
 OTTO_ZERTIFIKAT_PFAD_FALSCH otto_statuscode.h 141
 OTTO_ZERTIFIKAT_PIN_FALSCH otto_statuscode.h 141
 OttoDatenAbholen otto.h 124
 OttoEmpfangBeenden otto.h 126
 OttoEmpfangBeginnen otto.h 126
 OttoEmpfangBeginnenAbholzertifikat otto.h 127
 OttoEmpfangFortsetzen otto.h 128
 OttoEmpfangHandle otto_types.h 147
 OttoHoleFehlertext otto.h 128
 OttoInstanzErzeugen otto.h 128

OttoInstanzFreigeben	otto.h 129	OttoVersandBeenden	otto.h 133
OttoInstanzHandle	otto_types.h 147	OttoVersandBeginnen	otto.h 134
OTTOLOG_DEBUGMELDUNGEN	otto_types.h 149	OttoVersandFortsetzen	otto.h 135
OTTOLOG_FEHLERMELDUNGEN	otto_types.h 149	OttoVersandHandle	otto_types.h 148
OTTOLOG_INFORMATIONEN	otto_types.h 149	OttoVersion	otto.h 135
OTTOLOG_WARNUNGEN	otto_types.h 149	OttoZertifikatHandle	otto_types.h 148
OttoLogCallback	otto_types.h 147	OttoZertifikatOeffnen	otto.h 136
OttoLogEbene	otto_types.h 148	OttoZertifikatSchliessen	otto.h 137
OttoProxyKonfiguration	13	pdfCallback	eric_druck_parameter_t 6
authentifizierungsmethode	13	pdfCallbackBenutzerdaten	eric_druck_parameter_t 6
benutzerName	14	pdfName	eric_druck_parameter_t 6
benutzerPasswort	14	pin	eric_verschluesselungs_parameter_t 8
url	14	platform.h	160
version	14	ATOI64	161
OttoProxyKonfigurationSetzen	otto.h 129	HAS_FUTIME	161
OttoPruefsummeAktualisieren	otto.h 130	I64	161
OttoPruefsummeErzeugen	otto.h 130	uint32_t	161
OttoPruefsummeFreigeben	otto.h 131	UTIME_NEEDS_CLOSED_FILE	161
OttoPruefsummeHandle	otto_types.h 147	Start	2
OttoPruefsummeSignieren	otto.h 131	uint32_t	platform.h 161
OttoRueckgabepufferErzeugen	otto.h 132	url	OttoProxyKonfiguration 14
OttoRueckgabepufferFreigeben	otto.h 132	UTIME_NEEDS_CLOSED_FILE	platform.h 161
OttoRueckgabepufferGroesse	otto.h 132	version	eric_druck_parameter_t 7
OttoRueckgabepufferHandle	otto_types.h 148	eric_verschluesselungs_parameter_t	8
OttoRueckgabepufferInhalt	otto.h 133	eric_zertifikat_parameter_t	12
OttoStatusCode	otto_statuscode.h 139	OttoProxyKonfiguration	14
OttoVersandAbschliessen		vorschau	eric_druck_parameter_t 7
		zertifikatHandle	eric_verschluesselungs_parameter_t 9