

ERiC 43.3.2.0

ERiC-Tutorial

Inhalt

1	EINLEITUNG	4
1.1	Typographische Konventionen	4
1.2	Zweck des Dokuments und Zielgruppe	5
1.3	Was ist die ERiC API?	5
1.4	Wie kann die ERiC API verwendet werden?	5
1.5	Aufbau des ERiC-Tutorials	6
1.6	Abkürzungen.....	6
2	VORBEREITENDE MAßNAHMEN	7
2.1	Voraussetzungen unter Windows.....	7
2.2	Registrieren als Softwarehersteller.....	7
2.3	Anmelden im Entwicklerbereich (Login)	8
2.4	Beantragen der Hersteller-ID bzw. Hersteller-IDs	9
2.5	Herunterladen der ERiC API	10
2.6	Installieren der ERiC API.....	12
2.7	Die XML-Dateien und die PDF-Dokumente des ERiC-Tutorials	13
2.8	Eingeben der zugeteilten Hersteller-ID in die XML-Dateien.....	14
2.9	Die Beispielprogramme „ericdemo“ und „ottodemo“ kompilieren und ausführen.....	14
2.10	Das Beispielprogramm „ericdemo“ und die Kernfunktionen der ERiC API	16
2.11	Verwenden der ERiC API-Referenz	20
2.12	ELSTER Forum für Entwickler.....	21
3	VERARBEITUNG VON JAHRESSTEUERN AM BEISPIEL EST	22
3.1	Erste Aufgabe: Die ELSTER-XML Eingangsdaten am Beispiel ESt erstellen	23
3.1.1	Von den Daten des Steuerpflichtigen zu den ELSTER Feldern	24
3.1.2	Die Struktur der ELSTER eXML-Daten für die ESt.....	26
3.1.3	Bedeutung der Feldkennungen, -formate und -texte	30
3.1.4	Die ESt eXML Daten erstellen	31
3.2	Zweite Aufgabe: Verarbeitung der Steuerdaten	37
3.2.1	Schritt 1 - Implementieren Sie das empfohlene Vorgehen der Initialisierung beim Laden der ERiC Komponenten	38
3.2.2	Schritt 2 - Aufruf der ERiC API.....	38
3.2.3	Schritt 3 - Plausibilitätsprüfungen	43
3.2.4	Schritt 4 - Übermittlung an den ELSTER Annahmeserver.....	46
3.2.5	Schritt 5 - PDF-Erstellung	48
3.3	Dritte Aufgabe: Neue und geänderte Felder in die Steuersoftware integrieren	49
3.3.1	Beispiel zum VZ-Wechsel für die ESt von 2016 auf 2017	50
3.4	Vierte Aufgabe: Den TransferHeader mit EricCreateTH() erstellen	52
3.4.1	Schritt 1 - XML-Eingabedatei erstellen	52
3.4.2	Schritt 2 - EricCreateTH() in ericdemo aufrufen.....	53
4	ERIC OPTIMAL IN DIE STEUERSOFTWARE AM BEISPIEL EST_2020 INTEGRIEREN	56
5	FORTSCHRITTCALLBACKS AM BEISPIEL ERICDEMO IMPLEMENTIEREN	59
6	DATENABHOLUNG: ALLGEMEINER ABHOL-PROZESS (NICHT FÜR VAST-BELEGE)	62
6.1	XML-Anfrage erstellen, ob Daten abgeholt werden können	63
6.2	Serverantwort auswerten.....	65
6.3	Bereitgestellte Daten vom OTTER-Server herunterladen	66


6.4	XML-Bestätigung senden, dass Daten erfolgreich abgeholt wurden	67
7	ZUSAMMENFASSUNG.....	69
8	TABELLENVERZEICHNIS	70
9	ABBILDUNGSVERZEICHNIS.....	71

1 Einleitung

1.1 Typographische Konventionen

In diesem Dokument werden besondere Aspekte durch Formatierung hervorgehoben.

Tabelle 1-1 Typographische Konventionen

Formatierungskonvention	Informationstyp
<div style="border: 1px solid black; padding: 2px; display: inline-block;">ERICAPI DECL eric fehler t</div>	Code(-Beispiele) und Daten(-Strukturen) befinden sich in einem grauen Kasten.
<div style="border-top: 1px solid black; border-bottom: 1px solid black; padding: 5px; display: inline-block;"> HINWEIS: Hinweistext</div>	Hinweistexte werden oben und unten von einer Linie begrenzt. An der linken Seite zeigt ein Hand-Symbol auf den fett gedruckten Text „ HINWEIS: “.
https://www.elster.de	Hyperlinks von Internet-Adressen und Querverweise zu anderen Kapiteln sind blau dargestellt und unterstrichen.

1.2 Zweck des Dokuments und Zielgruppe

Das vorliegende Tutorial führt Sie anhand eines Beispiels einer Einkommensteuer (**ESt**) Schritt für Schritt durch die Steuersoftwareentwicklung mit ERiC (ElsterRichClient).

Das Tutorial wendet sich an Softwareentwickler, die erstmalig eine Steuersoftware erstellen und hierzu ERiC einsetzen. Grundkenntnisse in der Softwareentwicklung werden vorausgesetzt, die verwendete Programmiersprache in diesem Tutorial ist C++.

Das Tutorial ist weitestgehend plattformunabhängig. Das **ESt**-Beispiel wird an der gebräuchlichsten Desktopplattform Windows demonstriert, Linux-, AIX- und macOS-Entwickler können analog vorgehen.

1.3 Was ist die ERiC API?

Die ERiC API ist eine Schnittstellenspezifikation, die die Erstellung von ELSTER-kompatibler Clientsoftware für Steuer-, Finanz- und Lohnbuchhaltung unterstützt. Die ERiC API steht allen auf www.elster.de registrierten Softwareherstellern zur Verfügung und ist für die Betriebssystemplattformen Microsoft Windows, macOS, Linux und AIX verfügbar.

Lesen Sie im [ERiC-Entwicklerhandbuch.pdf](#) (EHB¹), welche Steuerverfahren und -arten von der ERiC API unterstützt werden.

1.4 Wie kann die ERiC API verwendet werden?

Die ERiC API kann von der eingesetzten Programmiersprache verwendet werden, wenn diese das Einbinden von C-Bibliotheken unterstützt, das sind z. B. C++ oder Java mit JNA. Zur Integration in eine Anwendung können die ERiC Bibliotheken einerseits vom Linker der jeweiligen Entwicklungsumgebung eingebunden werden oder andererseits dynamisch unter Verwendung von [LoadLibrary\(\)](#) bzw. [dlopen\(\)](#) geladen werden. Das dynamische Laden können Sie am C++ Beispielprogramm „ericdemo“ sehen.

¹ siehe EHB, Kap. „Unterstützte Fachverfahren und Daten- / Steuerarten“

1.5 Aufbau des ERiC-Tutorials

Das Kap. [2 Vorbereitende Maßnahmen](#) beschreibt die Voraussetzungen, damit Sie erfolgreich mit ERiC ein Steuersoftwareprogramm erstellen können. Das ist beispielsweise die Installation und Konfiguration der IDE.

Das Kap. [3 Verarbeitung von Jahressteuern am Beispiel ESt](#) führt Sie mit einem **ESt**-Beispiel durch die Integration des ERiC in die Steuersoftware. Die Unterkapitel enthalten die Aufgaben:

- Kap. [3.1 Erste Aufgabe: Die ELSTER-XML Eingangsdaten am Beispiel ESt erstellen](#)
- Kap. [3.2 Zweite Aufgabe: Verarbeitung der Steuerdaten](#)
- Kap. [3.3 Dritte Aufgabe: Neue und geänderte Felder in die Steuersoftware integrieren](#)
- Kap. [3.4 Vierte Aufgabe: Den TransferHeader mit EricCreateTH\(\) erstellen](#)

Das Kap. [4 ERiC optimal in die Steuersoftware am Beispiel ESt_2020 integrieren](#) zeigt am Beispiel **ESt_2020**, welche ERiC Plugins und Bibliotheken bei der Integration in die Steuersoftware notwendig sind bzw. weggelassen werden können und welche Vorteile sich daraus ergeben.

Das Kap. [5 Fortschrittcallbacks am Beispiel ericdemo implementieren](#) demonstriert die Verwendung von Fortschrittcallbacks.

Das Kap. [6 Datenabholung: Allgemeiner Abhol-Prozess \(nicht für VaSt-Belege\)](#) demonstriert die asynchrone Datenabholung mit Anhängen.

Das Tutorial wird vom Kap. [7 Zusammenfassung](#) abgeschlossen.

1.6 Abkürzungen

EHB	ERiC-Entwicklerhandbuch.pdf
ERiC	ElsterRichClient
ESt	Einkommenssteuer
eXML	einheitliches ELSTER-XML Siehe EHB, Kap. „Einführung in die neue Nutzdatenstruktur“.
VZ	Veranlagungszeitraum

2 Vorbereitende Maßnahmen

2.1 Voraussetzungen unter Windows

Es wird vorausgesetzt, dass auf dem Rechner des Steuersoftware-Entwicklers folgende Software² installiert ist:

- Microsoft Windows als Betriebssystem
- das benötigte Visual C++ Redistributable Package
- eine Microsoft Visual Studio Edition

2.2 Registrieren als Softwarehersteller

1. Rufen Sie www.elster.de/eportal/start auf.
2. Klicken Sie auf **Benutzergruppen** → **Entwickler** → **Registrierung als Hersteller / Entwickler**.

Abbildung 2-1 Initiale Registrierung



3. Füllen Sie das Formular **Registrierung für Entwickler** aus und klicken Sie auf **Absenden**.
4. Bitte lesen Sie auch alle weiteren Punkte auf der Webseite unter der Überschrift **Registrierung als Hersteller/Entwickler**. Siehe [A] in Abbildung oben.
5. Bitte beachten Sie, dass Sie Ihre persönlichen Benutzerdaten (Benutzername und Passwort) erst einige Tage nach der Registrierung per E-Mail erhalten.

² Siehe EHB ([ERiC-Entwicklerhandbuch.pdf](#)), Kap. „Mindestanforderungen an benötigte Software“

2.3 Anmelden im Entwicklerbereich (Login)



HINWEIS:

Bitte beachten Sie, dass Sie Ihre persönlichen Benutzerdaten (Benutzername und Passwort) erst einige Tage nach der Registrierung per E-Mail erhalten.

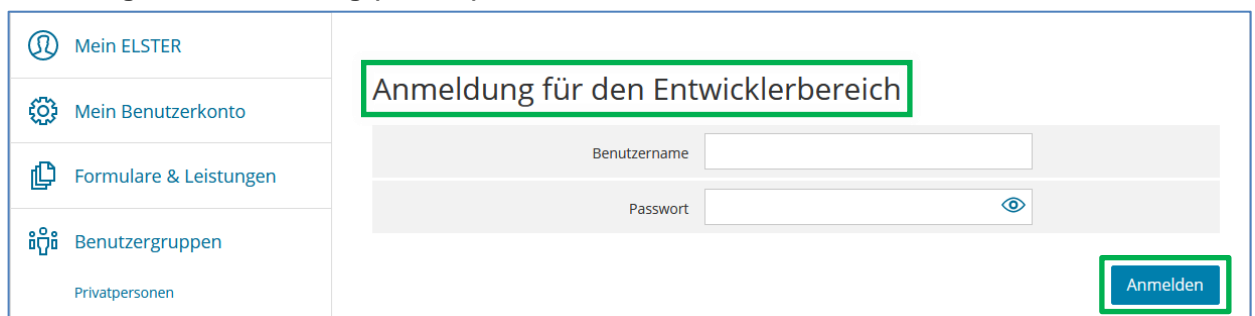
1. Rufen Sie www.elster.de/eportal/start auf.
2. Klicken Sie auf **Benutzergruppen** → **Entwickler** → **ERiC (ELSTER Rich Client)**:

Abbildung 2-2 Anmeldung (1 von 2)



3. Geben Sie auf der Seite **Anmeldung für den Entwicklerbereich** Ihren Benutzernamen und Passwort ein und klicken Sie auf **Anmelden**:

Abbildung 2-3 Anmeldung (2 von 2)



Die Seite **ELSTER Rich Client (ERiC)** wird angezeigt.

Nächster Vorbereitungsschritt:

Siehe Kap. [Beantragen der Hersteller-ID bzw. Hersteller-IDs](#).

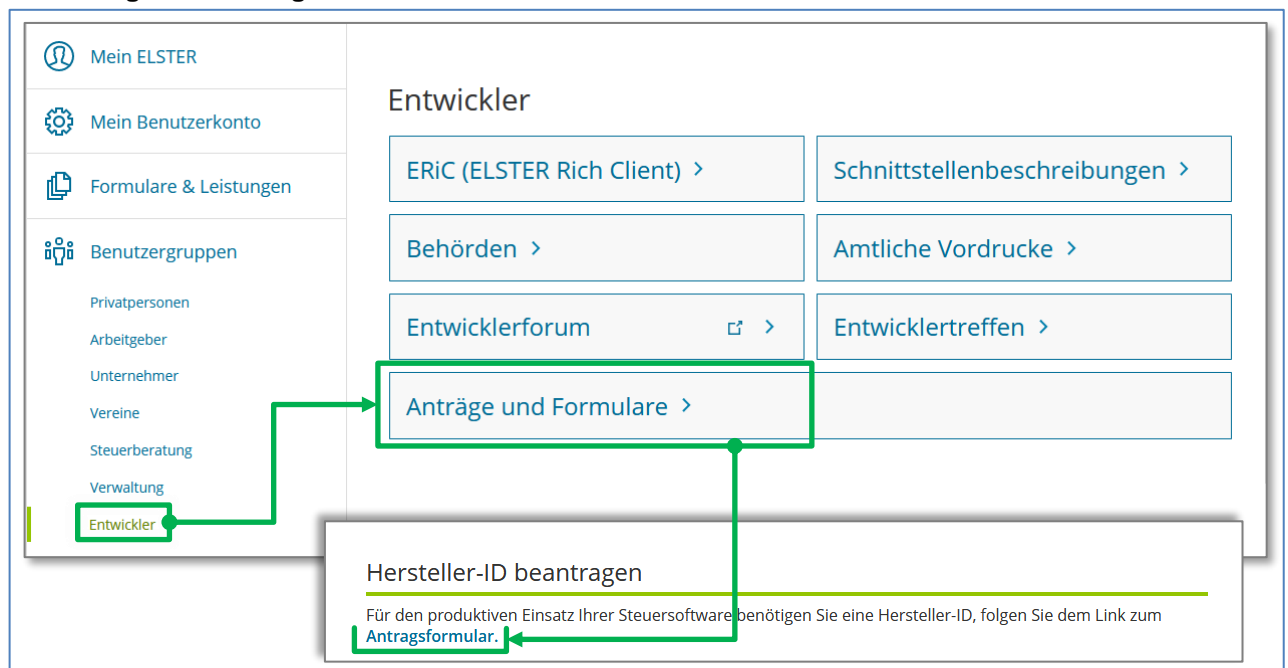
2.4 Beantragen der Hersteller-ID bzw. Hersteller-IDs

Ihr Unternehmen benötigt mindestens eine Hersteller-ID³.

In der folgenden Beschreibung wird davon ausgegangen, dass Sie sich gerade eingeloggt haben (Kap. [2.3](#)) und nun eine Hersteller-ID beantragen wollen.

1. Klicken Sie auf **Entwickler** → **Anträge und Formulare** → **Antragsformular**:

Abbildung 2-4 Antragsformular für eine Hersteller-ID aufrufen



2. Füllen Sie das Formular **Antragsformular für eine Hersteller-ID** aus und klicken Sie auf **Absenden**.



HINWEIS:

Bitte beachten Sie, dass die Hersteller-ID produktbezogen ist. Sofern Sie mehrere Software-Produkte anbieten, beantragen Sie bitte für jedes Produkt eine eigene Hersteller-ID.

Nächster Vorbereitungsschritt:

Die ERiC API-Dateien herunterladen, siehe Kap. [Herunterladen der ERiC API](#).

³ Zum Verwendungszweck der Hersteller-IDs siehe EHB, Kap. „Hersteller-ID bzw. Hersteller-IDs“

2.5 Herunterladen der ERiC API

In der folgenden Beschreibung wird davon ausgegangen, dass Sie sich gerade eingeloggt haben (Kap. 2.3) und nun die Seite **ELSTER Rich Client (ERiC)** angezeigt wird.

1. Klicken Sie für das gewünschte Release auf **Lizenzvertrag anzeigen**, lesen Sie ihn und klicken Sie dann auf **Lizenzvertrag akzeptieren**.
2. Laden Sie für das vorliegende ERiC-Tutorial die Dokumentation [A], die Schemadokumentation [B], ein 64-Bit ERiC-Release für Windows [C] und die amtlichen Vordrucke [D] herunter:⁴

Abbildung 2-5 Downloads (Teil 1 von 2)

ERiC-Release 42

Dokumentation

Dokumentationen	Stand
A ERiC-<version>-Dokumentation.zip	
B ERiC-<version>-Schemadokumentation.zip	

ERiC-Pakete

ERiC für Windows	Version und Datum der Bibliotheken	Stand
C ERiC-<version>-Windows-x86_64.jar	Versionsinformationen	

Amtliche Vordrucke

Amtliche Vordrucke	Größe	Stand
D Vordrucke_archive_ERiC-<version>.zip (2014 - 2023)	1 GB	

[A] [ERiC-<version>-Dokumentation.zip](#): Das Dokumentationspaket beinhaltet im Verzeichnis Dokumentation\ das [ERiC-Entwicklerhandbuch.pdf](#) (EHB). Auf das EHB wird in diesem Tutorial häufig verwiesen. Das EHB ist die zentrale Dokumentation und beinhaltet detaillierte Beschreibungen für alle Themen, die für die Anwendungsentwicklung mit ERiC relevant sind.

[B] [ERiC-<version>-Schemadokumentation.zip](#): Das Schemadokumentationspaket.

⁴ siehe EHB, Kap. „Inhalt des ERiC“

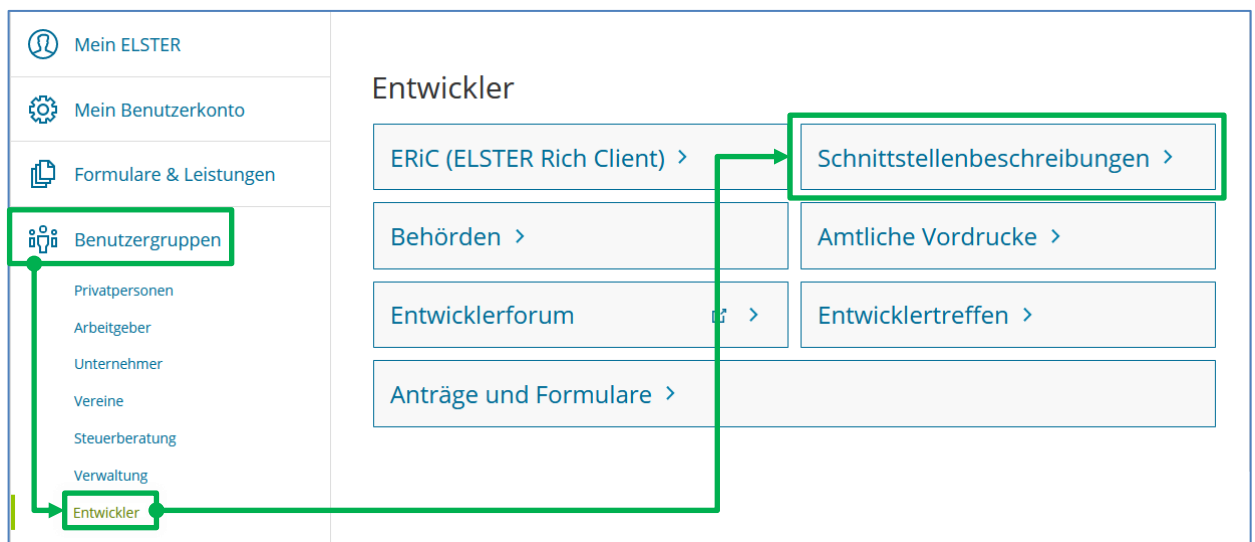
[C] [ERiC-<version>-Windows-x86_64.jar](#): Die Windows jar-Datei ist ein Archiv und keine Bibliothek. Ein Entpackprogramm, z. B. 7-Zip, wird benötigt, um die darin enthaltenen ERiC-Bibliotheken zu extrahieren.

Die anderen ERiC Softwarepakete für Linux, macOS und AIX sind in der obigen Abbildung aus Platzgründen nicht dargestellt.

[D] [Vordrucke_archive_ERiC-<version>.zip](#): Die im Tutorial verwendeten amtlichen Vordrucke sind in der *.zip-Datei zu finden. Nachfolgend werden die **Est**-Vordrucke für den Veranlagungszeitraum (VZ) 2020 benötigt.

3. Navigieren Sie zu **Schnittstellenbeschreibungen**:

Abbildung 2-6 Downloads (Teil 2 von 2)



4. Laden Sie auf der Seite **Schnittstellenbeschreibungen und Dokumentationen** die folgenden Dokumentationen herunter:

- ELSTER-Fehlerliste
- Dokumentation Steuernummern- und Identifikationsnummernprüfung

Zum Durcharbeiten des Tutorials benötigen Sie weitere Softwarepakete. Die passende Version der weiteren Softwarepakete kann im EHB, Kap. „Ergänzende Downloadpakete“ nachgelesen werden.



HINWEIS:

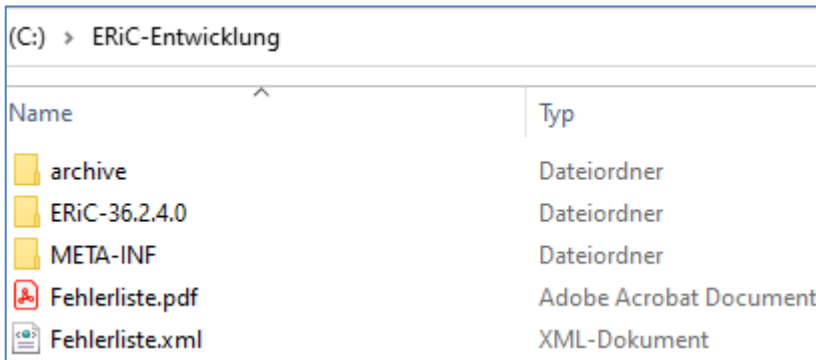
Halten Sie zusätzlich das EHB ([ERiC-Entwicklerhandbuch.pdf](#)) bereit, da sich das Tutorial vertiefend darauf bezieht.

Sie finden die Datei unter: Dokumentation\ERiC-Entwicklerhandbuch.pdf

2.6 Installieren der ERiC API

Entpacken Sie die heruntergeladenen Pakete in dasselbe Arbeitsverzeichnis, z. B. C:\ERiC-Entwicklung

Abbildung 2-7 Nach dem Entpacken



Name	Typ
archive	Dateiordner
ERiC-36.2.4.0	Dateiordner
META-INF	Dateiordner
Fehlerliste.pdf	Adobe Acrobat Document
Fehlerliste.xml	XML-Dokument



HINWEIS:

Im weiteren Verlauf des Tutorials werden nur noch relative Pfade ausgehend von Ihrem gewählten Arbeitsverzeichnis angegeben.

Beispiel:

- Gesamter Pfad: „C:\ERiC-Entwicklung\ERIC-<version>\Dokumentation“
- Relativer Pfad: „~~\Dokumentation“

2.7 Die XML-Dateien und die PDF-Dokumente des ERiC-Tutorials

Zum ERiC-Tutorial der ERiC API gehören folgende Dateien und Dokumente:

Tabelle 2-1 Dateien und Dokumente des ERiC-Tutorials

Datei / Dokument	Verzeichnis Beschreibung
ERiC-Tutorial.pdf	~~\Dokumentation\Tutorial\ Dieses Dokument.
ESt_2020-Beispiel_Loesung.xml	~~\Dokumentation\Tutorial\Beispiele\ XML-Lösungsdatei des ESt -Beispiels.
ericprint.pdf	~~\Dokumentation\Tutorial\Beispiele\ PDF-Lösungsdatei des ESt -Beispiels.
PostfachAnfrage.xml PostfachBestaetigung.xml	~~\Dokumentation\Tutorial\Beispiele\ XML-Lösungsdateien des ElsterDatenabholung -Beispiels.



HINWEIS:

Halten Sie zusätzlich das EHB ([ERiC-Entwicklerhandbuch.pdf](#)) bereit, da sich das Tutorial vertiefend darauf bezieht.

Sie finden die Datei unter: `~~\Dokumentation\ERiC-Entwicklerhandbuch.pdf`

2.8 Eingeben der zugeteilten Hersteller-ID in die XML-Dateien

Bitte geben Sie in die XML-Dateien des ERiC-Tutorials Ihre Hersteller-ID in das Element **<HerstellerID>** ein. Die betroffenen XML-Dateien listet die [Tabelle 2-1](#) oben auf.

Die Hersteller-ID 74931 in den XML-Dateien des ERiC-Tutorials dient nur zu Illustrationszwecken.



HINWEIS:

Wenn Sie noch keine eigene Hersteller-ID beantragt haben, so holen Sie dies jetzt bitte nach. Siehe [Beantragen der Hersteller-ID bzw. Hersteller-IDs](#).

Sie benötigen die Ihnen zugeteilte Hersteller-ID nicht nur für die XML-Dateien im Tutorial, sondern auch für die XML-Dateien der Test- und Echtfälle!

2.9 Die Beispielprogramme „ericdemo“ und „ottodemo“ kompilieren und ausführen

Kompilieren und Linken von „ericdemo“ bzw. „ottodemo“:

1. Navigieren Sie im Windows-Explorer nach:
 - `~~\Windows-x86_64\Beispiel\ericdemo-cpp\`
 - `~~\Windows-x86_64\Beispiel\ottodemo-cpp\`
2. Doppelklicken Sie auf die jeweilige *.sln-Datei, um das Beispielprojekt in der Microsoft IDE zu öffnen:
 - [ericdemo-2017-x64.sln](#)
 - [ottodemo-2017-x64.sln](#)
3. Übersetzen Sie das Beispielprojekt „ericdemo“ bzw. „ottodemo“.
4. Überprüfen Sie, ob das ausführbare Programm im Debug-Verzeichnis zu finden ist:
 - `~~\Windows-x86_64\Beispiel\ericdemo-cpp\x64\Debug\ericdemo.exe`
 - `~~\Windows-x86_64\Beispiel\ottodemo-cpp\x64\Debug\ottodemo.exe`

„ericdemo“ ausführen:

1. Navigieren Sie in das Verzeichnis `~~\Windows-x86_64\Beispiel\ericdemo-cpp\`
2. Geben Sie in die Datei `ESt_2020.xml` Ihre Hersteller-ID ein.
Siehe [Eingeben der zugeteilten Hersteller-ID in die XML-Dateien](#).
3. Doppelklicken Sie die Batchdatei `starte-ericdemo.bat`.

**HINWEIS:**

Achten Sie bei Problemen auf die Programmausgaben und prüfen Sie die Logdatei [eric.log](#) auf Fehlermeldungen.

Verifizieren Sie die korrekte Ausführung vorangegangener Schritte und beachten Sie die Hinweise in der Datei [Liesmich.txt](#).

„ottodemo“ ausführen:

1. Navigieren Sie in das Verzeichnis `~~\Windows-x86_64\Beispiel\ottodemo-cpp\`
2. Doppelklicken Sie die Batchdatei `starte-ottodemo.bat`.
3. Geben Sie im Konsolenfenster Ihre Hersteller-ID ein. Erst dann kann die ottodemo ausgeführt werden:

Abbildung 2-8 Hersteller-ID eingeben zum Ausführen der ottodemo

```
C:\WINDOWS\system32\cmd.exe
Starte ottodemo ...
Bitte geben Sie Ihre Hersteller-ID ein: 
*** Hole Datei von OTTER ***
Objekt-ID: 7090bc69-be5e-4fd0-b91a-2128021295d6
Speichere Daten in: beispiel.xml
Empfange Daten ..
547 Byte erfolgreich abgeholt.

Bitte druecken Sie die Eingabetaste
```

**HINWEIS:**

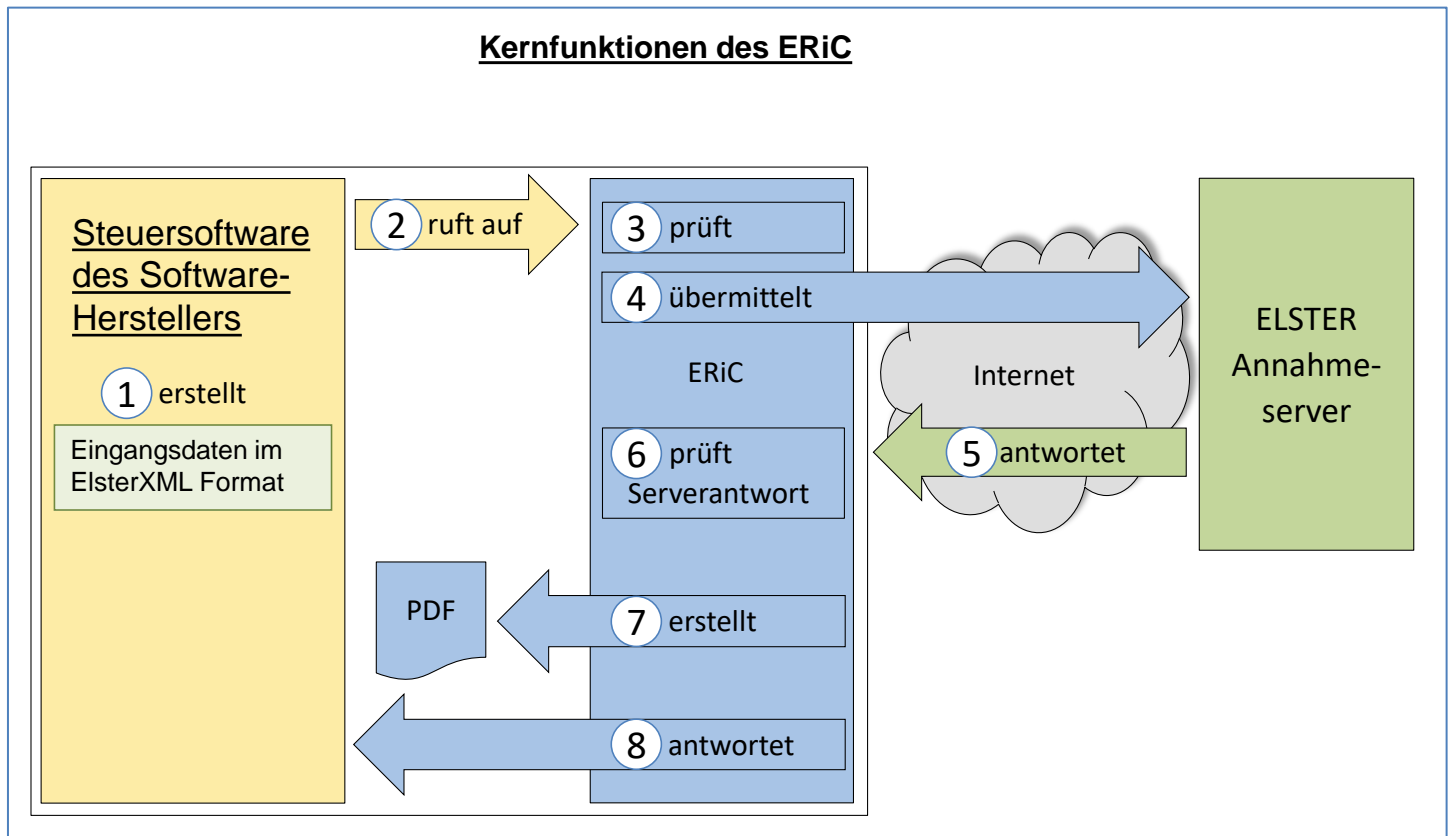
Achten Sie bei Problemen auf die Programmausgaben und prüfen Sie die Logdatei [otto.log](#) auf Fehlermeldungen.

Verifizieren Sie die korrekte Ausführung vorangegangener Schritte und beachten Sie die Hinweise in der Datei [Liesmich.txt](#).

2.10 Das Beispielprogramm „ericdemo“ und die Kernfunktionen der ERiC API

Dieser Abschnitt vermittelt Ihnen am C++ Programmbeispiel „ericdemo“ die Kernfunktionen der ERiC API. Die in folgender Abbildung visualisierten Abläufe können Sie in der anschließenden Beschreibung und in den Codebeispielen nachvollziehen.

Abbildung 2-9 Die Kernfunktionen des ERiC



(1) Steuersoftware des Software-Herstellers erstellt ElsterXML-Datei

Die Steuersoftware stellt der ERiC API die Eingangsdaten im Elster-XML-Format⁵ bereit. Beim Beispielprogramm „ericdemo“ ist dies die Datei [ESt_2020.xml](#).

1. Gehen Sie hierzu in das Verzeichnis `~\Windows-x86_64\Beispiel\ericdemo-cpp\`.
2. Öffnen Sie die Datei [ESt_2020.xml](#) mit einem Text- oder XML-Viewer.
 - Das Element `<DatenArt>` mit dem Wert „EST“ wird vom übergeordneten Element `<TransferHeader>` ummantelt.

⁵ siehe EHB, Kap. „Testunterstützung bei der ERiC-Anbindung“ und Dokumentation\Schnittstellenbeschreibungen\ElsterBasisSchema\Dokumentation\Einheitliche_Datenschnittstelle_XML_11.pdf

- ERiC unterstützt Sie beim Testen durch die Bereitstellung von Test-Steuernummern und Test-Finanzämtern⁶. Das Element `<Testmerker>`, im Beispiel ist es der Wert „700000004“, muss in den Testdaten gesetzt sein.

Abbildung 2-10 Auszug aus der Datei „ESt_2020.xml“

```
<Elster>
  <TransferHeader version="11">
    ...
    <DatenArt>ESt</DatenArt>
    <Testmerker>700000004</Testmerker>
    ...
  </TransferHeader>
  <DatenTeil>
    ...
    ...
  </DatenTeil>
</Elster>
```

(2) Steuersoftware des Software-Herstellers ruft ERiC auf

Die XML-Eingangsdaten in der Datei `ESt_2020.xml` werden neben anderen Funktionsparametern im Beispielprogramm „ericdemo“ der ERiC API-Funktion ***EricBearbeiteVorgang()*** übergeben.

⁶ siehe EHB, Kap. „Testunterstützung bei der ERiC-Anbindung“

(3) und (4) Prüfen und übermitteln an den ELSTER Annahmeserver

Alle Daten, die von der Steuersoftware an die ERiC-Schnittstelle übergeben werden, müssen eine Plausibilitätsprüfung durchlaufen.

Bei der Plausibilitätsprüfung werden die Daten auf formale Korrektheit und Vollständigkeit geprüft. Die formale Korrektheit bezieht sich auf die Prüfung des Formates, der maximalen Längen und der Gültigkeit des Feldes im Kontext der Steuerart und des Veranlagungszeitraums.

Nachdem ERiC die Funktionsparameter überprüft und die XML-Eingangsdaten erfolgreich auf Plausibilität geprüft hat, erfolgt der verschlüsselte Datenversand an den ELSTER Annahmeserver.

1. Öffnen Sie die Datei [ericvorgang.cpp](#).
2. Sehen Sie sich in der Methode `EricVorgang::ausfuehren()` folgenden API-Funktionsaufruf an:

Abbildung 2-11 Ausschnitt aus ericvorgang.cpp: API-Funktionsaufruf

```
const int rc = ericAdapter.EricBearbeiteVorgang(
    xmlDaten.c_str(), argParser.getDatenartVersion().c_str(),
    bearbeitungsFlags, &druckEinstellungen, verschluesselungsParameter,
    argParser.getHatTransferHandle() ? &transferHandle : nullptr,
    ergebnisPuffer.handle(), serverantwortPuffer.handle() );
```

(5) ELSTER Annahmeserver antwortet

Der ELSTER Annahmeserver antwortet. Das Ergebnis der Plausibilitätsprüfung und die möglichen Fehlermeldungen der Serverantwort werden über den jeweiligen ERiC-Rückgabepuffer (`ergebnisPuffer` und `serverantwortPuffer`) von **`EricBearbeiteVorgang()`** an den Aufrufer über ein Handle zurückgegeben. Zur Erzeugung, Verwendung und Freigabe von Rückgabepuffern siehe `EricRueckgabepufferHandle` in der API-Referenz⁷.

Der Rückgabewert `ERIC_OK` signalisiert Erfolg, andere Werte einen Fehler. Die möglichen Fehlerwerte mit Fehlertexten⁸ können Sie in der API-Referenz und im Softwarepaket in der Datei [eric_fehlercodes.h](#)⁹ nachlesen.

⁷ Sie lernen die API-Referenz im nächsten Kap. [2.11 Verwenden der ERiC API-Referenz](#) kennen.

⁸ Eine Fehlerauswertung mit Fehlerbehandlung in der Steuersoftware ist damit möglich.

⁹ Windows-x86_64\include\

(6) ERiC prüft Serverantwort

ERiC prüft anhand der Serverantwort, ob bei der serverseitigen Verarbeitung ein Fehler aufgetreten ist. Falls ja, wird die Bearbeitung im ERiC abgebrochen und der Funktionsaufruf mit einer entsprechenden Fehlermeldung beendet. Die Steuerungssoftware muss in diesem Fall die im Parameter *serverantwortPuffer* zurückgelieferte Serverantwort auswerten.

(7) ERiC erstellt PDF-Datei

Nachdem die ERiC API mit **ERIC_OK** geantwortet hat, ist eine PDF-Datei von ERiC erstellt worden.

Sie sehen im nächsten Codebeispiel aus „ericdemo“, Datei [ericvorgang.cpp](#) in der Struktur **eric_druck_parameter_t**, wie die ERiC Druckeinstellungen und der PDF-Dateiname gesetzt sind.

Abbildung 2-12 Ausschnitt aus [ericvorgang.cpp](#): ERiC Druckeinstellungen

```
/** @brief Dies sind die Standardeinstellungen des Beispiels. */
eric_druck_parameter_t holeDruckeinstellungen()
{
    eric_druck_parameter_t druckEinstellungen = {};
    druckEinstellungen.version      = 4;
    druckEinstellungen.vorschau    = 0;
    druckEinstellungen.duplexDruck = 0;
    druckEinstellungen.pdfName     = "ericprint.pdf";
    druckEinstellungen.fussText    = nullptr;
    druckEinstellungen.pdfCallback = nullptr;
    druckEinstellungen.pdfCallbackBenutzerdaten = nullptr;
    return druckEinstellungen;
}
```

Näheres über die zulässigen Datenfelder und Werte, siehe API-Referenz, [eric_types.h](#)
Struktur **eric_druck_parameter_t**

(8) ERiC antwortet

Als Ergebnis des Funktionsaufrufs **EricBearbeiteVorgang()** wird die Serverantwort in den Parameter *serverantwortPuffer* zurückgeliefert. Siehe [Abbildung 2-11](#).

Im Fall von Fehlern bei der Plausibilitätsprüfung werden die Fehler in den Parameter *ergebnisPuffer* zurückgeliefert. Siehe [Abbildung 2-11](#).

2.11 Verwenden der ERiC API-Referenz

Die ERiC API-Referenzdokumentation wird im HTML- und PDF-Format ausgeliefert.

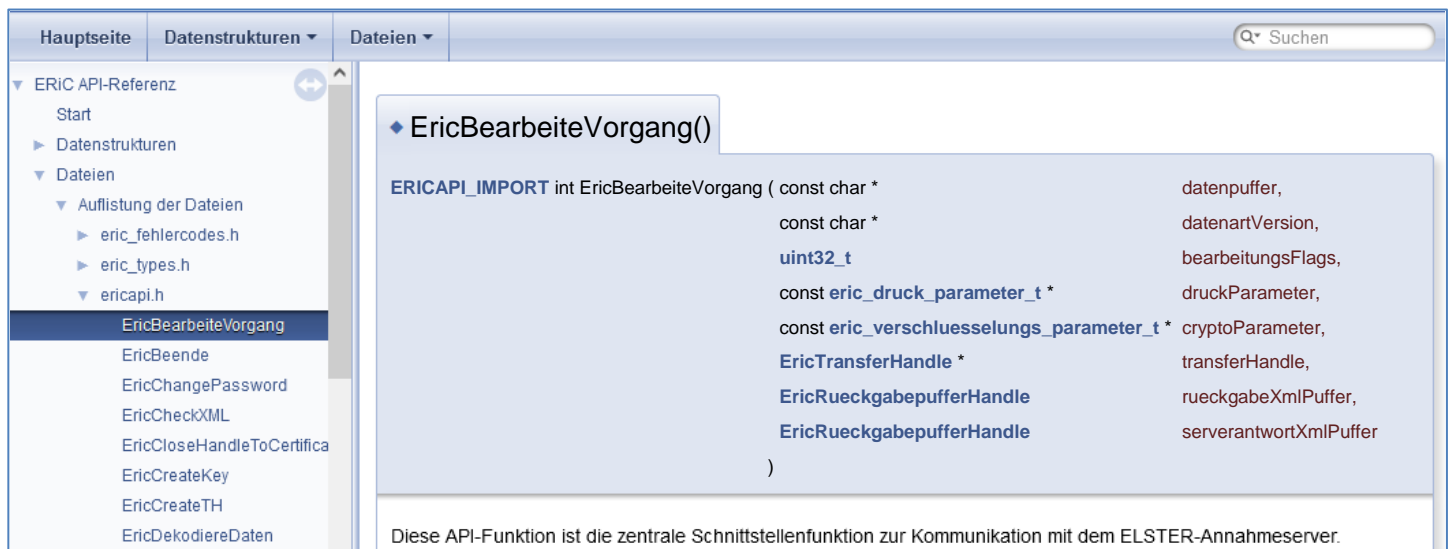
1. Öffnen Sie die HTML-Hauptseite Dokumentation\API-Referenz\HTML\index.html in einem Web-Browser Ihrer Wahl.

Über den oberen oder linken Navigationsbereich gelangen Sie zu den von ERiC verwendeten Headerdateien, Datenstrukturen und API-Funktionen.

2. Öffnen Sie beispielsweise über den linken Navigationsbereich die Datei [ericapi.h](#), siehe [Abbildung 2-13](#).

Hier sehen Sie die Funktionssignatur und Referenzdokumentation zur API-Funktion *EricBearbeiteVorgang()*.

Abbildung 2-13 ERiC API-Referenz (html): Beispiel einer Funktionssignatur



The screenshot shows the ERiC API-Referenz (html) interface. On the left, there is a navigation pane with tabs for 'Hauptseite', 'Datenstrukturen', and 'Dateien'. The 'Dateien' tab is selected, showing a list of files including 'ericapi.h'. The main content area displays the function signature for 'EricBearbeiteVorgang()'. The signature is as follows:

```
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
                                           )
```

Below the signature, a description states: 'Diese API-Funktion ist die zentrale Schnittstellenfunktion zur Kommunikation mit dem ELSTER-Annahmeserver.'

2.12 ELSTER Forum für Entwickler

Im Herstellerforum können Sie nach weiterführenden Informationen zu Problemen suchen, die andere schon gelöst haben.¹⁰

1. Rufen Sie im Browser die Seite <https://forum.elster.de/herstellerforum> auf.
2. Melden Sie sich mit Ihren Hersteller-Logindaten an.

Abbildung 2-14 ELSTERFORUM für Entwickler: Startseite nach dem Login



¹⁰ siehe EHB, Kap. „Probleme und mögliche Ursachen“

3 Verarbeitung von Jahressteuern am Beispiel ESt

Am Beispiel der Einkommensteuer für den Veranlagungszeitraum 2020 mit ausgewählten Anlagen führt das nachfolgende Kapitel Schritt für Schritt durch die Integration des ERiC in die Steuersoftware:

- Kap. [3.1 Erste Aufgabe: Die ELSTER-XML Eingangsdaten am Beispiel ESt erstellen](#)
- Kap. [3.2 Zweite Aufgabe: Verarbeitung der Steuerdaten](#) (auf Plausibilität prüfen, Versenden, PDF-Druck)
- Kap. [3.3 Dritte Aufgabe: Neue und geänderte Felder in die Steuersoftware integrieren](#)
- Kap. [3.4 Vierte Aufgabe: Den TransferHeader mit EricCreateTH\(\) erstellen](#)

Anschließend erfahren Sie, wie Sie Ihre Steuersoftware an ein neues ERiC-Release anpassen können.

**HINWEIS:**

Denken Sie daran, nur die Ihnen zugeteilte Hersteller-ID im Element `<HerstellerID>` zu verwenden!

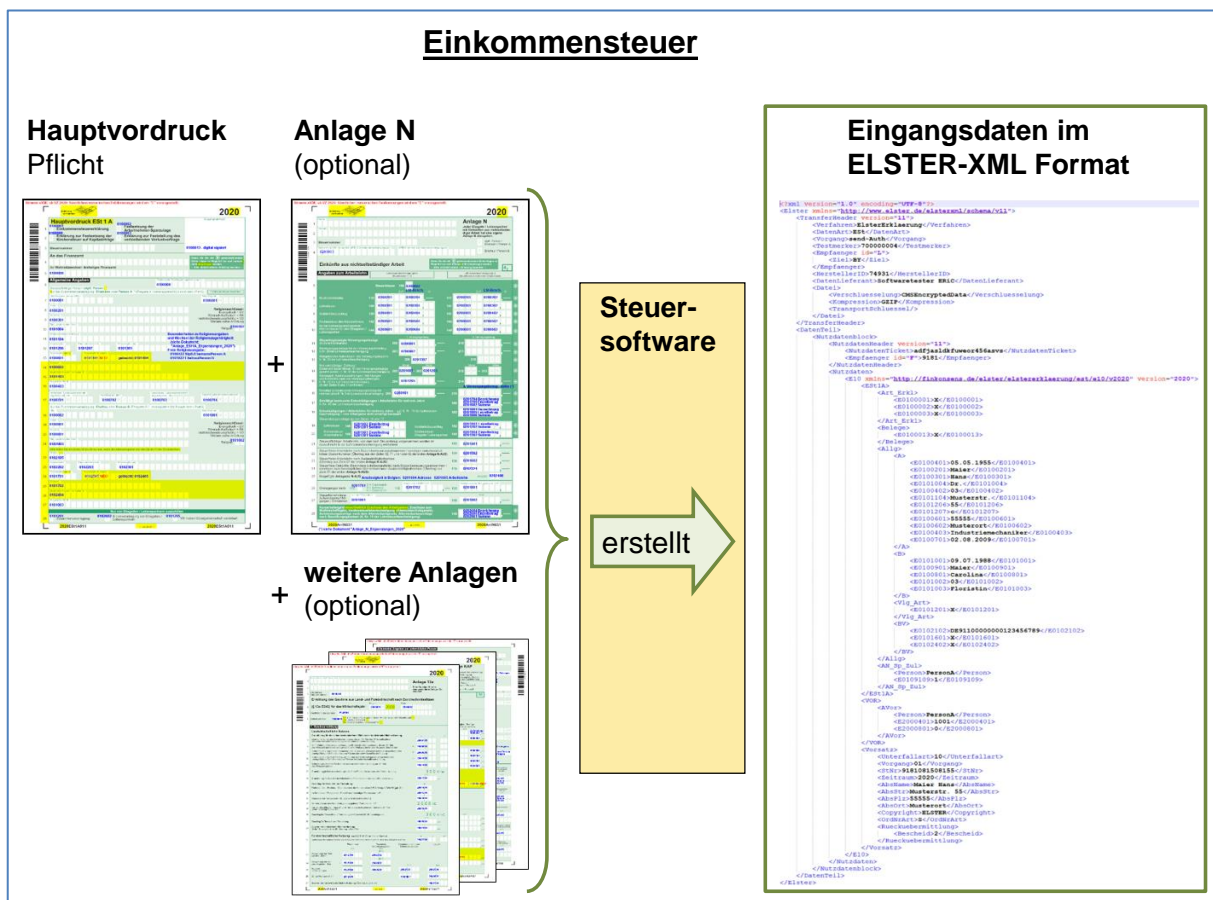
Siehe Kap. [2.4 Beantragen der Hersteller-ID bzw. Hersteller-IDs](#).

3.1 Erste Aufgabe: Die ELSTER-XML Eingangsdaten am Beispiel ESt erstellen

Steuersoftware, die Datensätze mit ELSTER bzw. mit ERiC an den ELSTER Annahmeserver übertragen möchte, muss aus den Eingaben des Benutzers einen gültigen, vollständigen ELSTER-XML Datensatz aufbauen und an ERiC übergeben. Dazu ist die Kenntnis der in einem VZ gültigen Felder für den Hauptvordruck mit optionalen Anlagen nötig.

Abbildung 3-1 stellt schematisch die Aufgaben der Steuersoftware, ein **EST** ELSTER-XML zu erstellen, dar.

Abbildung 3-1 Erstellung der ESt XML-Eingangsdaten



In den folgenden Abschnitten wird erklärt,

- wie die Eingaben des Steuerpflichtigen einem ELSTER-Feld zugeordnet werden, siehe Kap. [3.1.1 Von den Daten des Steuerpflichtigen zu den ELSTER Feldern](#)
- wie die **Est** eXML-Daten für den VZ 2020 strukturiert sind, siehe Kap. [3.1.2 Die Struktur der ELSTER eXML-Daten für die Est](#)
- wie Feldkennungen, -formate und -texte aufgebaut sind, siehe Kap. [3.1.3 Bedeutung der Feldkennungen, -formate und -texte](#)
- wie und wo Sie die benötigte Information zu gültigen ELSTER-Feldern und den zugehörigen Plausibilitätsprüfungen finden, siehe Kap. [3.1.4 Die Est eXML Daten erstellen](#)

3.1.1 Von den Daten des Steuerpflichtigen zu den ELSTER Feldern

Ihr Steuerprogramm, in das Sie ERiC integrieren wollen, verwendet elektronische Formulare, in die der Steuerpflichtige seine Steuerdaten einträgt. Wie genau Ihre elektronischen Formulare aussehen, ist unerheblich. Aber die Formulare werden mindestens die Pflichtfelder der amtlichen Vordrucke für die **Est** beinhalten.

In diesem Tutorial werden die (Pflicht-)Felder aus dem **Est** Hauptvordruck und einige weitere Anlagen verwendet, um die ELSTER-XML Eingangsdaten aufzubauen.

In diesem Beispiel wird u.a. der Hauptvordruck, Dateiname [2020Est1A011.png](#), verwendet. Öffnen Sie den Hauptvordruck jetzt. Die Vordrucke finden Sie im Verzeichnis 2020\Erklaerungssteuern\Est\Grafiken_und_Erweiterungen_E10\

Abbildung 3-2 Ausschnitt aus „2020Est1A011.png“

Hinweise eXML ab VZ 2020: Sämtlichen numerischen Feldkennungen wird ein "E" vorangestellt; Markierungen "-----" von nicht umgesetzten Feldern entfallen.

A Anleitung vorhanden

2020

Hauptvordruck Est 1 A

1 **0100001** Einkommensteuererklärung **0100002** Festsetzung der Arbeitnehmer-Sparzulage

2 **0100009** Erklärung zur Festsetzung der Kirchensteuer auf Kapitalerträge **0100003** Erklärung zur Feststellung des verbleibenden Verlustvortrags

3 Steuernummer **0100013 - digital signiert**

An das Finanzamt

Bei Wohnsitzwechsel: bisheriges Finanzamt

5 **0100006**

Allgemeine Angaben

6 Steuerpflichtige Person (stpfl. Person) **0100008** Telefonische Rückfragen tagsüber unter Nr.

Nur bei Zusammenveranlagung: Ehemann oder Person A *) (Ehegatte A / Lebenspartner[in] A nach dem LPartG) *) Bitte Anleitung beachten.

Identifikationsnummer (IdNr.) Geburtsdatum

B **0100081** **0100401** M J J J J

Name

8 **0100201**

Vorname

9 **0100301**

Titel, akademischer Grad

10 **0101004** Religionsschlüssel: Evangelisch = EV Römisch-Katholisch = RK nicht kirchensteuerpflichtig = VD Weitere siehe Anleitung Religion **0100402**

11 **0101104** **F** Besonderheiten zu Religionsangaben und Wechsel der Religionszugehörigkeit: (siehe Dokument: "Anlage_Est1A_Ergaenzungen_2020") Freie Religionsangabe: 0100422 Stpfl./Ehemann/Person A 0101022 Ehefrau/Person B

12 **0101206** Hausnummer Hausnummerzusatz **0101207** Adressergänzung **0101301**

13 **0100601** Postleitzahl (Inland) Postleitzahl (Ausland) **0101405 NEU!** gelöscht: **0101404**

14 **0100602** Wohnort **D** **E**

15 **0101403** Staat (falls Anschrift im Ausland)

- [A]** Beachten Sie den roten Hinweis in der Kopfzeile des Vordrucks. Die **Est** für den Veranlagungszeitraum 2020 verwendet die neue Nutzdatenstruktur (kurz eXML). Lesen Sie hierzu die ausführlichen Informationen zu eXML im EHB, Kap. „Einführung in die neue Nutzdatenstruktur“.
- [B]** Die amtlichen Vordrucke sind um Informationen für den Softwareentwickler erweitert (annotiert). Die auszufüllenden Felder sind auf den Vordrucken mit blauen Nummern, den Feldkennungen versehen.
- [C]** Änderungen zum Vorjahr sind gelb markiert.

[D] Falls Felder erstmalig in diesem Vordruck vorhanden sind, erfolgt eine rote Markierung mit dem Wort „**NEU!**“.

Beispielsweise ist die Feldkennung „0101405“ erstmalig verwendet und deshalb mit dem roten Wort „**NEU!**“ markiert.

[E] Gelöschte Feldkennungen sind unter „gelöscht:“ aufgezählt.

[F] Hinweise auf ergänzende Dokumente werden bei Bedarf aufgedruckt, hier z. B. „Besonderheiten zu Religionsangaben“.

Die eindeutigen Feldkennungen sind zusammen mit den zugehörigen, vom Steuerpflichtigen eingegebenen Daten (Werte) des Steuerfalls in die eXML Eingangsdaten zu übernehmen.

Über die Feldkennungen werden die vom Steuerpflichtigen erklärten Werte von den verarbeitenden Systemen der Finanzbehörde inhaltlich zugeordnet.

Wie die Ihnen nun bekannten Feldkennungen bei der Erstellung der eXML Eingangsdaten verwendet werden, erfahren Sie in den nächsten Abschnitten.

3.1.2 Die Struktur der ELSTER eXML-Daten für die ESt

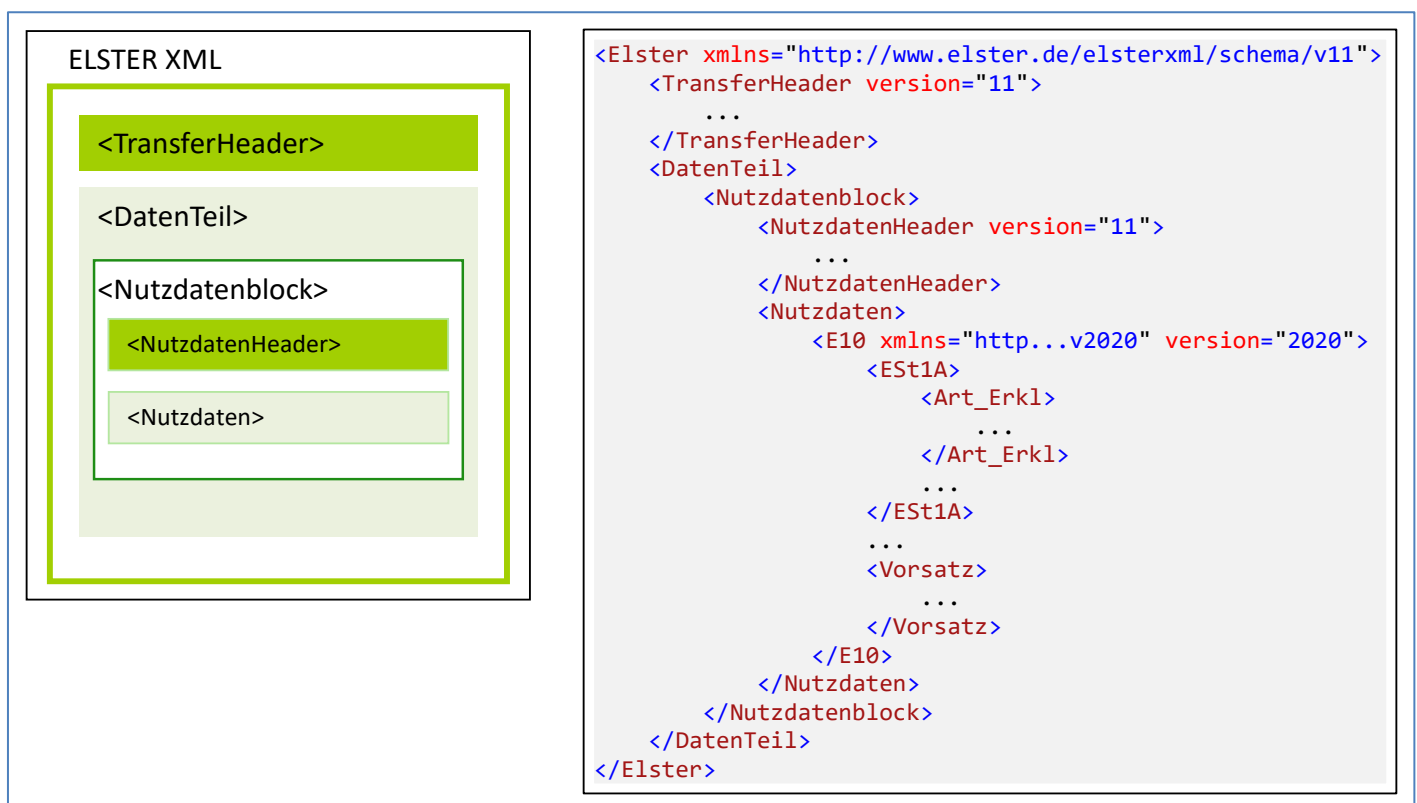
Dieser und die folgenden Abschnitte setzen das installierte und lauffähige Beispielprogramm „ericdemo“ voraus. Bitte lesen Sie ggf. das Kap. [2 Vorbereitende Maßnahmen](#).

Die zu erstellende eXML-Datei mit den Steuerdaten beinhalten sowohl die Daten des Hauptvordrucks als auch die Daten der optionalen Anlagen. Sie sehen dies im Beispieldatensatz „Windows-x86_64\Beispiel\ericdemo-cpp\ESt_2020.xml“.

Die zugrundeliegende, annotierte eXML-Schemabeschreibung ist ausgehend von Dokumentation\Schnittstellenbeschreibungen\Erklärungssteuern\ESt_10_2020\Schema\elster11_E10_2020_extern.xsd in E10-2020.xsd zu finden.

Der grundsätzliche, strukturelle Aufbau eines ELSTER eXML Eingangsdatensatzes ist in der folgenden Abbildung dargestellt. Zur besseren Lesbarkeit sind nur wesentliche Elemente enthalten.

Abbildung 3-3 XML-Beispiel: Struktur der ELSTER eXML-Daten für die ESt



Alle Erklärungssteuern folgen dieser Struktur.

Die Elemente **<TransferHeader>** und **<DatenTeil>** mit dem **<Nutzdatenblock>** und dem **<NutzdatenHeader>** sind für alle Datenarten gleich, nur das darunterliegende Element **<Nutzdaten>** mit seinem Inhalt ist spezifisch für Erklärungssteuern. Es wird empfohlen, den **<TransferHeader>** mit der API-Funktion **EricCreateTH()** zu erstellen, siehe Kap. [3.4](#).

3.1.2.1 Die Nutzdatenstruktur für eXML-Daten

Um sich mit der eXML Nutzdatenstruktur vertraut zu machen, öffnen Sie die Beispieldatei von ericdemo, [ESt_2020.xml](#)¹¹. Sie sehen, dass jeder Feldkennung das Präfix „E“ vorangestellt ist. Das Präfix „E“ ist ein typisches Merkmal für eXML-Daten.

Abbildung 3-4 XML-Beispiel: Nutzdatenstruktur für eXML-Daten

```
...
<Nutzdaten>
  <E10 xmlns="http://finkonsens.de/elster/elstererklaerung/est/e10/v2020" version="2020">
    <Est1A>
      <Art_Erkl>
        <E0100001>X</E0100001>
        <E0100002>X</E0100002>
        <E0100003>X</E0100003>
      </Art_Erkl>
      <Belege>
        <E0100013>X</E0100013>
      </Belege>
      <Allg>
        <A>
          <E0100401>05.05.1955</E0100401>
          <E0100201>Maier</E0100201>
          <E0100301>Hans</E0100301>
        </A>
      </Allg>
    </Est1A>
  </E10>
</Nutzdaten>
...
```

Das Schema und die Schemadokumentation sind für die Integration in die Steuersoftware unverzichtbare Hilfsmittel. Die Schemadokumentation wird im HTML-Format bereitgestellt und kann mit einem Internetbrowser, z. B. Firefox, verwendet werden.

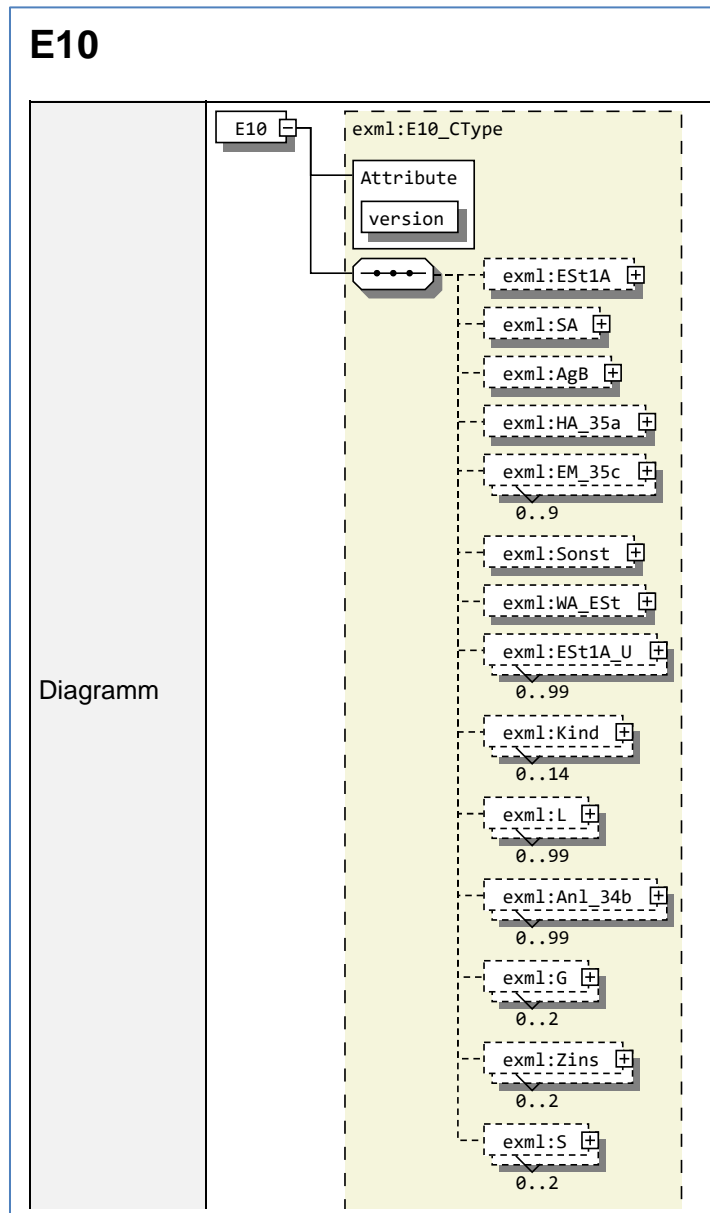
¹¹ Windows-x86_64\Beispiel\ericdemo-cpp\

Öffnen Sie für dieses Beispiel jetzt

„Dokumentation\Schnittstellenbeschreibungen\Erklaerungssteuern\Est_10_2020\SchemaDokumentation\EST10-2020.html“

Klicken im linken oberen Bereich auf das Element „E10“, um sich mit der eXML Struktur vertraut zu machen:

Abbildung 3-5 Ausschnitt aus E10-2020.html: Schema für ESt_2020



Damit sollte der Inhalt der Datei [EST_2020.xml](#) verständlich sein.

Beachten Sie, dass die Reihenfolge der Elemente strikt dem Schema folgt!

Aus dem annotierten, amtlichen Vordruck [2020ESt1A011.png](#) entnehmen Sie, dass die Feldkennung „E0100401“ das Geburtsdatum der steuerpflichtigen Person beinhaltet. Vom Steuerpflichtigen wurde der Wert „05.05.1955“ angegeben. Ist das hier verwendete Format „dd.mm.yyyy“ richtig? Das eXML-Schema sagt darüber nichts aus.

Die Antwort auf diese Frage liefert das Feldformat. Es wird im folgenden Abschnitt in der Dokumentation der Plausibilitätsprüfungen vorgestellt. Sie beinhaltet Informationen über formale Prüfungen (Felder, Formatdefinitionen) und inhaltliche Prüfungen (Regeln, Werteberechnungen), die Sie benötigen.

Öffnen Sie das Dokument [Zusatzinformationen_zur_Plausibilitaetspruefung.pdf](#)¹², denn es erklärt das Grundkonzept der Felder, Regeln und Kennzahlen und erläutert die Sprachkonstrukte für die Regelbedingungen, die Feldtypen und formale Prüfungen. Mit diesem Wissen ist das Verständnis für die Jahresdokumentation gegeben. Öffnen Sie nun die **ES**t Jahresdokumentation für den VZ 2020 [Jahresdokumentation_10_2020.xml](#)¹³ mit Excel oder einem kompatiblen Programm.

Abbildung 3-6 Ausschnitt aus der ESt Jahresdokumentation 2020

Sie erfahren in der Spalte:

- ¹³ Dokumentation\Plausipruefungen\Erklaerungssteuern\ESt\UFA10\

3.1.4 Die ESt eXML Daten erstellen

Das Wissen der vorangegangenen Abschnitte ist notwendig, damit Sie in diesem Kapitel ausgehend von den **ESt** Vordrucken, die **ESt** eXML Daten erstellen können.

Verwenden Sie zum Erstellen der XML-Daten einen Texteditor Ihrer Wahl.

Die Einzelschritte sind:

- Kap. [3.1.4.1 Vom Vordruck zu den ESt eXML Daten](#)
- Kap. [3.1.4.2 Felder, die mehrfach angegeben werden können](#)

3.1.4.1 Vom Vordruck zu den ESt eXML Daten

Am Beispiel der Feldkennungen und -werte aus dem **ESt_2020** Hauptvordruck und den eXML-Daten¹⁴ aus ericdemo, wird exemplarisch gezeigt, wie Sie Schritt für Schritt einen **ESt** eXML Datensatz erstellen können.

Öffnen Sie die Seite 1 des annotierten amtlichen Hauptvordrucks [2020ESt1A011.png](#)¹⁵.

Entnehmen Sie die benötigten Feldkennungen (rot und grau markiert):

¹⁴eXML-Datei: Windows-x86_64\Beispiel\ericdemo-cpp\ESt_2020.xml

¹⁵Vordruck: 2020\Erklaerungssteuern\ESt\Grafiken_und_Erweiterungen_E10\

Abbildung 3-7 Hauptvordruck ESt

Hinweise eXML ab VZ 2020: Sämtlichen numerischen Feldkennungen wird ein "E" vorangestellt; Markierungen "-----" von nicht umgesetzten Feldern entfallen.

Anleitung vorhanden

2020

Hauptvordruck ESt 1 A

0100001 Einkommensteuererklärung

0100002 Festsetzung der Arbeitnehmer-Sparzulage

0100009 Erklärung zur Festsetzung der Kirchensteuer auf Kapitalerträge

0100003 Erklärung zur Feststellung des verbleibenden Verlustvortrags

Steuernummer

0100013 - digital signiert

An das Finanzamt

Bei Wohnsitzwechsel: bisheriges Finanzamt

0100006

Allgemeine Angaben

Telefonische Rückfragen tagsüber unter Nr. 0100008

Steuerpflichtige Person (stpf. Person)

Nur bei Zusammenveranlagung: Ehemann oder Person A *) (Ehegatte A / Lebenspartner[in] A nach dem LPartG)

*) Bitte Anleitung beachten.

Identifikationsnummer (IdNr.)

0100081

Geburtsdatum

0100401 M J J J J

Name

0100201

Vorname

0100301

Titel, akademischer Grad

0101004

Straße (derzeitige Adresse)

0101104

Hausnummer

0101206

Hausnummerzusatz

0101207

Adressergänzung

0101301

Postleitzahl (Inland)

0100601

Postleitzahl (Ausland)

0101405 NEU!

gelöscht: 0101404

Wohnort

0100602

Staat (falls Anschrift im Ausland)

0101403

Religionsschlüssel:
Evangelisch = EV
Römisch-Katholisch = RK
nicht kirchensteuerpflichtig = VD
Weitere siehe Anleitung

Religion

0100402

Besonderheiten zu Religionsangaben und Wechsel der Religionszugehörigkeit: (siehe Dokument: "Anlage_ESt1A_Ergaenzungen_2020")
Freie Religionsangabe:
0100422 Stpf./Ehemann/Person A
0101022 Ehefrau/Person B

Wie die Felder im eXML Datensatz zu verschachteln sind, entnehmen Sie dem eXML Schema¹⁶ und der Schemadokumentation¹⁷. Dabei werden Sie feststellen:

- Die in der obigen Abbildung rot eingekreisten Felder sind im XML-Element `<Nutzdatenblock>` einzutragen, und zwar:
 - `Nutzdatenblock/Nutzdaten/E10/Est1A/Art_Erk1/E0100001`
 - `Nutzdatenblock/Nutzdaten/E10/Est1A/Allg/A/E0100201` und `E0100301`
- Das grau eingekreiste Feld **Steuernummer** ist im XML-Element `<Nutzdatenblock>` einzutragen, und zwar: `Nutzdatenblock/Nutzdaten/E10/Vorsatz/StrNr`

¹⁶ Dokumentation\Schnittstellenbeschreibungen\Erklaerungssteuern\ESt_10_2020\Schema\

¹⁷ Dokumentation\Schnittstellenbeschreibungen\Erklaerungssteuern\ESt_10_2020\SchemaDokumentation\

**HINWEIS:**

Die Lösungsdatei finden Sie unter

Dokumentation\Tutorial\Beispiele\EST_2020-Beispiel_Loesung.xml

Das Feld **Steuernummer** kann nicht 1:1 in den eXML-Datensatz übernommen werden, es muss zuerst in das 13-stellige ELSTER-Steuernummerformat konvertiert werden. Hierzu gehen Sie wie folgt vor:

- Öffnen Sie das Dokument [Pruefung_der_Steuer_und_Steueridentifikatsnummer.pdf](#)¹⁸ und machen sich mit dem Inhalt vertraut, insbesondere mit Kap. 3 „Aufbau der Steuernummer im ELSTER-Steuernummerformat“.
- Entnehmen Sie dem Dokument [Finanzamtsdaten.xlsx](#)¹⁹, je nach Finanzamtsnummer, welche BuFa (Bundesfinanzamtsnummer) der Steuernummer zuzuordnen ist.
- Entnehmen Sie dem PDF-Dokument die Regel der 5ten Stelle der Steuernummer.
- Bilden Sie nun die ELSTER-Steuernummer und tragen Sie diese jetzt in die eXML Daten ein.

Programmatisch kann mit der API-Funktion [EricMakeElsterStnr\(\)](#) eine Steuernummer im Format des Steuerbescheides in eine 13-stellige Steuernummer des ELSTER-Steuernummerformats umgewandelt werden.

Den Aufbau der 13-stelligen ELSTER-Steuernummer finden Sie auch im XML-Schema und der Schemadokumentation.

¹⁸ befindet sich im Downloadbereich für Entwickler:

<https://www.elster.de/elsterweb/entwickler/infoseite/schnittstellenbeschreibungen>

¹⁹ Dokumentation\Finanzamtsdaten.xlsx

3.1.4.2 Felder, die mehrfach angegeben werden können

Bisher konnte ein Feld des Hauptvordrucks einem Wert eindeutig zugeordnet werden. Es gibt Felder, die mehrmals verwendet werden. Die Felder von „Spenden und Mitgliedsbeiträge“ sind solche Felder. Eine eindeutige Feldzuordnung wird mit dem sog. „Kontext“ hergestellt, die „maximale Wiederholbarkeit“ gibt an, wie oft das Feld verwendet werden kann. Lesen Sie jetzt die weiterführenden Informationen zum „Kontext“ in [Zusatzinformationen_zur_Plausibilitaetspruefung.pdf](#)²⁰ und im EHB, Kap. „Einführung in die neue Nutzdatenstruktur“.

Wie Sie ein mehrmals verwendetes Feld in die eXML Daten aufnehmen, lernen Sie anhand der folgenden Beispieldaten:

Tabelle 3-1 Beispieldaten für 2x „Spenden und Mitgliedsbeiträge“

Zeile	Feldkennung	Feldwert
6	E0104704	Musikverein
6	E0104705	200
6	E0104708	200
6	E0104704	Schützenverein
6	E0104708	100
6	E0104702	200
6	E0104703	300

²⁰ Dokumentation\Plausipruefungen\

Die Felder „Spenden und Mitgliedsbeiträge“ befinden sich auf dem annotierten amtlichen Vordruck [2020AnlSonderausgaben401.png](#)²¹. Öffnen Sie nun diesen Vordruck:

Abbildung 3-8 Vordruck

Zuwendungen (Spenden und Mitgliedsbeiträge)									
Spenden und Mitgliedsbeiträge (ohne Beträge in den Zeilen 9 bis 12)				lt. Bestätigungen EUR		lt. Betriebsfinanzamt EUR			
5	zur Förderung steuerbegünstigter Zwecke an Empfänger im Inland	123	0108105			124	0108106		
6	zur Förderung steuerbegünstigter Zwecke an Empfänger im EU- / EWR-Ausland	133	0104702			134	0104703		
7	an politische Parteien (§§ 34g, 10b EStG)	127	0108701			128	0108702		
8	an unabhängige Wählervereinigungen (§ 34g EStG)	129	0108801			130	0108802		
siehe (*)									
Spenden in das zu erhaltende Vermögen (Vermögensstock) einer Stiftung				stift. Person / Ehemann / Person A EUR		Ehefrau / Person B EUR			
9	2020 geleistete Spenden an Empfänger im Inland (lt. Bestätigungen / lt. Betriebsfinanzamt)	208	0108405			209			
10	2020 geleistete Spenden (lt. Bestätigungen / lt. Betriebsfinanzamt) an Empfänger im EU- / EWR-Ausland	224	0105502			225			
11	Von den Spenden in den Zeilen 9 und 10 sollen 2020 berücksichtigt werden	212	0108509			213			
12	2020 zu berücksichtigende Spenden aus Vorjahren in das zu erhaltende Vermögen (Vermögensstock) einer Stiftung, die bisher noch nicht berücksichtigt wurden	214	0108607			215			
0105902 - Summe der Umsätze, Löhne und Gehälter zur Ermittlung des Spendenhöchstbetrages (Kz 108)									

Die gewünschten Felder, z. B. „E0104708“ können im Abschnitt „Spenden und Mitgliedsbeiträge“ nicht gefunden werden, jedoch befindet sich am rechten Rand des Vordrucks für die Zeilen 5 – 8 eine Referenz auf die Fußnote „siehe (*)“.

Öffnen Sie nun das in der Fußnote referenzierte Dokument

[Anlage_Sonderausgaben_Ergaenzungen_2020.pdf](#)²¹:

Abbildung 3-9 Anlage Sonderausgaben

Ergänzungen Anlage Sonderausgaben 2020						
Hinweis eXML ab VZ 2020: Sämtlichen numerischen Feldkennungen wird ein "E" vorangestellt.						
Zeile	Spenden und Mitgliedsbeiträge (ohne Spenden in das zu erhaltende Vermögen einer Stiftung)	Beschreibung	Betrag lt. Bestätigungen	Betrag lt. Nachweis Betriebsfinanzt	Summe der Beträge lt. Bestätigungen	Summe der Beträge lt. Nachweis Betriebsfinanzamt
5	zur Förderung steuerbegünstigter Zwecke an Empfänger im Inland	0108102	0108103	0108104	0108105 Kz 52.123	0108106 Kz 52.124
6	zur Förderung steuerbegünstigter Zwecke an Empfänger im EU- / EWR-Ausland	0104704	0104705	0104708	0104702 Kz 52.133	0104703 Kz 52.134
7	an politische Parteien (§§ 34g, 10b EStG)	0108716	0108717	0108718	0108701 Kz 52.127	0108702 Kz 52.128
8	an unabhängige Wählervereinigungen	0108804	0108805	0108806	0108801 Kz 52.129	0108802 Kz 52.130

²¹ Vordrucke: 2020\Erklärungssteuern\ESt\Grafiken_und_Erweiterungen_E10\

Sie finden das gesuchte Feld E0104708 in Zeile 6, Spalte „Betrag lt. Nachweis Betriebsfinanzamt“ und stellen fest, dass sich dort keine weitere Information zu dem Feld befindet.

In diesem Beispiel sollen diese Feldwerte zwei Mal angegeben werden, aber wie oft dürfen die Felder maximal verwendet werden? Die Antwort liefert der Wert „max. Wiederholbarkeit“ für den Kontext des Feldes aus der Jahresdokumentation²². Öffnen Sie die Jahresdokumentation und suchen Sie nach dem Feld E0104708. Sie finden es auf dem Tabellenblatt „SA - Felder“ und können dort den zugehörigen „Kontext“ „Zuw/Sp_MB/Foerd_st_beg_Zw_EU_EWR/Einz“ entnehmen. Mit diesem Kontext können Sie auf dem Tabellenblatt „SA - Kontexte“ den zugehörigen Wert „max. Wiederholbarkeit“ = 99 entnehmen. Das Feld E0104708 kann somit 99-mal angegeben werden.

Damit die Felder und Feldwerte an die richtige Position in das eXML eingefügt werden können, öffnen Sie das Schema und entnehmen die richtige Reihenfolge, siehe [Abbildung 3-5 Schema für ESt 2020](#).

Beachten Sie die Formate und Regeln aus der Jahresdokumentation für die Feldwerte.

Damit sind alle benötigten Informationen vorhanden, erweitern Sie nun die eXML-Daten und testen Sie diese z. B. mit ericdemo. Vergleichen Sie bei Bedarf die erstellten eXML-Daten mit der Beispiellösung [ESt_2020-Beispiel_Loesung.xml](#).

Falls beim Erstellen der eXML-Daten die Reihenfolge nicht eingehalten wird, erhalten Sie folgende Fehlermeldung ([eric.log](#)): `element 'SA' is not allowed for content model`
'SA' bezieht sich auf die hier verwendeten Beispieldaten in der Anlage Sonderausgaben.

Im folgenden Kap. [3.2 Zweite Aufgabe: Verarbeitung der Steuerdaten](#) lesen Sie, wie die eXML Daten auf Plausibilität geprüft und an den ELSTER Annahmeserver übermittelt werden. Das sog. Freizeichnungsdokument wird im PDF-Format erstellt.

²² Dokumentation\Plausipruefungen\Erklaerungssteuern\ESt\UFA10\
[Jahresdokumentation_10_2020.xml](#)

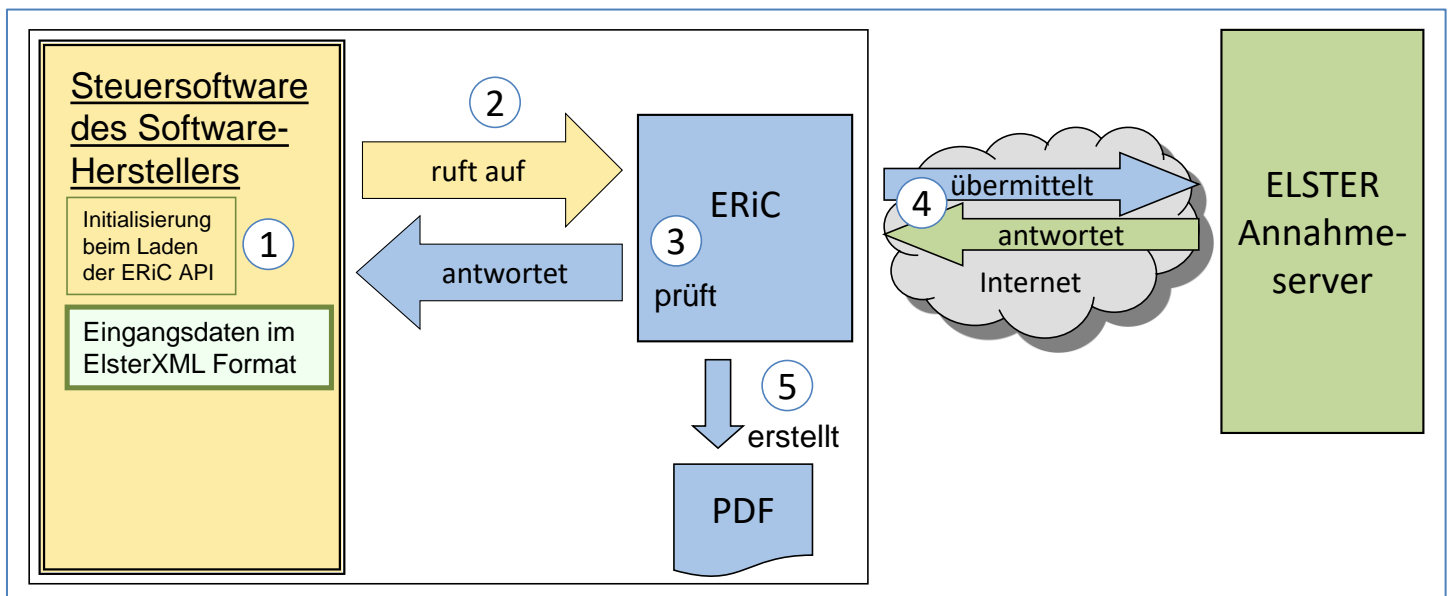
3.2 Zweite Aufgabe: Verarbeitung der Steuerdaten

Das von Ihnen in den vorangegangenen Abschnitten erweiterte **ES**t eXML werden Sie in diesem Kapitel mit ERiC verarbeiten. Sie erfahren:

- Wie Sie in Ihrer Software das empfohlene Vorgehen der Initialisierung beim Laden der ERiC Komponenten umsetzen können im Kap. [3.2.1 Schritt 1 - Implementieren Sie das empfohlene Vorgehen der Initialisierung beim Laden der ERiC Komponenten](#)
- Wie Sie die ERiC API aufrufen im Kap. [3.2.2 Schritt 2 - Aufruf der ERiC API](#)
- Wie Sie die eXML Steuerdaten mit ERiC auf Plausibilität prüfen im Kap. [3.2.3 Schritt 3 - Plausibilitätsprüfungen](#)
- Wie Sie die eXML Steuerdaten an den ELSTER Annahmeserver übermitteln im Kap. [3.2.4 Schritt 4 - Übermittlung an den ELSTER Annahmeserver](#)
- Wie das sog. Freizeichnungsdokument als PDF-Datei erstellt wird im Kap. [3.2.5 Schritt 5 - PDF-Erstellung](#)

In der folgenden Abbildung, die sich an der [Abbildung 2-9](#) der ERiC API-Kernfunktionen orientiert, ist der schematische Ablauf der Verarbeitung der Steuerdaten mit ERiC dargestellt. Für sehr viele Anwendungsfälle ist dieser Ablauf gleich. Falls in einem gewählten Anwendungsfall die PDF-Datei nicht erstellt werden soll, entfällt (5).

Abbildung 3-10 Datenverarbeitung mit ERiC



3.2.1 Schritt 1 - Implementieren Sie das empfohlene Vorgehen der Initialisierung beim Laden der ERiC Komponenten

Mit Ihrer zu erstellenden Steuersoftware sind die ERiC Basisbibliotheken und Plugins²³ mit auszuliefern, siehe EHB, Kap. „Empfohlenes Vorgehen bei der Initialisierung“.

Die folgenden Punkte zeigen die zur Initialisierung notwendigen Schritte des ERiC am Beispiel der Implementierung in „ericdemo“. Öffnen Sie hierzu [eric.cpp](#)²⁴.

1. Funktion `ladeEricApi(homeDir)` lädt **dynamisch** die Bibliothek „ericapi“ und ermittelt die Funktionszeiger:

Abbildung 3-11 Ausschnitt aus eric.cpp

```
libEricApi =  
    Resolve::library(System::dateiPfad(apiVerzeichnis, ericapiDateiname).c_str());  
...  
EricInitialisierePtr =  
    ladeFunktion<EricInitialisiereFun>("EricInitialisiere",  
        libEricApi);  
  
EricBearbeiteVorgangPtr =  
    ladeFunktion<EricBearbeiteVorgangFun>("EricBearbeiteVorgang",  
        libEricApi);  
...
```

2. Die ERiC API-Funktion [EricInitialisiere\(\)](#) sucht im Verzeichnis „homeDir“ rekursiv nach den ERiC Plugins und „logDir“ setzt das Verzeichnis, in das die Logdatei [eric.log](#) geschrieben wird:

Abbildung 3-12 Ausschnitt aus eric.cpp

```
statusCode = EricInitialisiere(homeDir.c_str(), logDir.c_str());
```

Zum Beenden der Anwendung ist [EricBeende\(\)](#) aufzurufen, um alle ERiC Plugins geordnet zu entladen.

3.2.2 Schritt 2 - Aufruf der ERiC API

Sie haben im Kap. [3.1](#) die **ES**t eXML-Daten²⁵ für den VZ 2020 erweitert und im Kap. [3.2.1](#) das empfohlene Vorgehen bei der Initialisierung implementiert. Damit sind alle Vorbereitungen zum Aufruf der ERiC API-Funktionen abgeschlossen.

²³ Windows-x86_64\dll* und Windows-x86_64\dll\plugins*

²⁴ Windows-x86_64\Beispiel\ericdemo-cpp\ericdemo\eric.cpp

²⁵ Alternativ kann die Lösungsdatei [ES_t_2020-Beispiel_Loesung.xml](#) verwendet werden.

3.2.2.1 Bedeutung und Aufruf der ERiC API-Funktion `EricBearbeiteVorgang()`

Die ERiC API-Funktion `EricBearbeiteVorgang()` nimmt Steuerdaten im ELSTER-XML Format entgegen, prüft sie auf Plausibilität und versendet sie, gibt mit dem Rückgabewert `rc` und dem `ergebnisPuffer` sowie dem `serverantwortPuffer` Auskunft über den Status der Einlieferung am ELSTER Annahmeserver und erstellt je nach Anwendungsfall eine PDF-Datei.

`EricBearbeiteVorgang()` wird für alle von ERiC unterstützten Datenarten verwendet. Deshalb lernen Sie diese ERiC API-Funktion jetzt näher kennen.

Schauen wir uns an, mit welchen Parametern `EricBearbeiteVorgang()` aufgerufen wird. Öffnen Sie hierfür die Datei `ericvorgang.cpp`²⁶, Methode `EricVorgang::ausfuehren()`, des Beispielprogramms „ericdemo“.

Abbildung 3-13 Ausschnitt aus `ericvorgang.cpp`: Aufruf von `EricBearbeiteVorgang()`

```
const int rc = ericAdapter.EricBearbeiteVorgang(
    xmlDaten.c_str(), argParser.getDatenartVersion().c_str(),
    bearbeitungsFlags, &druckEinstellungen, verschluesselungsParameter,
    argParser.getHatTransferHandle() ? &transferHandle : nullptr,
    ergebnisPuffer.handle(), serverantwortPuffer.handle() );
```

Die Funktionsparameter für das Beispiel sind:

- Der Eingangsparameter `xmlDaten` enthält die aus Kap. 3.1 erstellten **ES**t Steuerdaten im ELSTER XML-Format.
- Aus der Datenartversionmatrix²⁷ suchen Sie mit den Vorgaben:
Datenart = **ES**t und VZ = ‚2020‘ die `datenartVersion` = **ES**t_2020 heraus. Im Beispiel mit „ericdemo“ übergeben Sie die `datenartVersion` als Kommandozeilenparameter. Die `datenartVersion` muss zu den XML-Eingangsdaten passen.
- Mit den Bearbeitungsflags `bearbeitungsFlags` spezifizieren Sie, wie die `xmlDaten` verarbeitet werden sollen. Für eine Prüfung auf Plausibilität weisen Sie den Wert **ERIC_VALIDIERE**²⁸ zu. Mehrere Werte werden durch oder-Verknüpfungen zugewiesen.
- `&druckEinstellungen` ist ein Zeiger auf die Struktur `eric_druck_parameter_t` und definiert die ERiC Druckfunktionalität, z. B. hier ohne Vorschau druck:

²⁶ Windows-x86_64\Beispiel\ericdemo-cpp\ericdemo\ericvorgang.cpp

²⁷ siehe Dokumentation\Datenartversionmatrix.xml und EHB, Kap. „datenartVersion – Definition und Verwendung“

²⁸ Für weitere Werte der Bearbeitungsflags, siehe `eric_types.h` in der API-Referenz.

Abbildung 3-14 Ausschnitt aus `ericvorgang.cpp`: Definition der ERiC Druckfunktionalität

```
druckEinstellungen.version      = 4;
druckEinstellungen.vorschau    = 0;
druckEinstellungen.duplexDruck = 0;
druckEinstellungen.pdfName     = "ericprint.pdf";
druckEinstellungen.fussText    = nullptr;
druckEinstellungen.pdfCallback = nullptr;
druckEinstellungen.pdfCallbackBenutzerdaten = nullptr;
```

Weitere Informationen zum PDF-Druck erfahren Sie im EHB, Kap. „Anwendungsfälle von `EricBearbeiteVorgang()`“, Abschnitt „Druckkennzeichnung der Anwendungsfälle“.

- `verschluesselungsParameter` ist ein Zeiger auf die Struktur `eric_verschluesselungs_parameter_t`. Mit ihr kann ein Zertifikat (Zertifikatshandle und PIN) für einen authentifizierten Versand übergeben werden. Wird kein Zertifikat benötigt, ist der Parameter ein `nullptr`.
Ob eine Datenart authentifiziert („send-Auth“) versendet werden kann oder sogar muss, ist der Eigenschaften-Tabelle der jeweiligen Datenart im EHB zu entnehmen.
Für den Testfall **EST** 2020 benötigen Sie ein Zertifikat. Ein Testzertifikat können Sie aus dem Downloadbereich [Schnittstellenbeschreibungen](#) herunterladen. Siehe Kap. [3.2.2.2 Download des Test-Zertifikats](#).
- Der Parameter `transferHandle` ist nur in dem Sonderfall einer **ElsterDatenabholung** zu setzen. In allen anderen Fällen ist einfach ein `nullptr` zu übergeben. Daher wird das Transferhandle in diesem Beispiel nicht näher betrachtet.
- Die Parameter `ergebnisPuffer` und `serverantwortPuffer` sind vom Typ `EricRueckgabepufferHandle`. Beide müssen vor Übergabe an `EricBearbeiteVorgang()` mit `EricRueckgabepufferErzeugen()` angelegt werden. Im Code-Ausschnitt aus dem Beispielprogramm "ericdemo" geschieht dies durch die Instanziierung der Helferklasse `EricPuffer`, welche ebenfalls Teil des "ericdemo"-Projektes ist.

Mit den vorbereiteten Parametern können Sie `EricBearbeiteVorgang()` aufrufen und den Rückgabewert sowie die Inhalte von `ergebnisPuffer` und `serverantwortPuffer` auswerten. Zugriff auf die Pufferinhalte ermöglicht die API-Funktion `EricRueckgabepufferInhalt()`, im Code-Beispiel ausgeführt von der Methode `inhalt()` der Helferklasse `EricPuffer`.

Im `serverantwortPuffer` ist nach dem Versand von Daten die Antwort des ELSTER Annahmeservers enthalten. Mögliche serverseitige Fehlermeldungen können mit der API-Funktion `EricGetErrormessagesFromXMLAnswer()` aus dieser Antwort extrahiert werden.

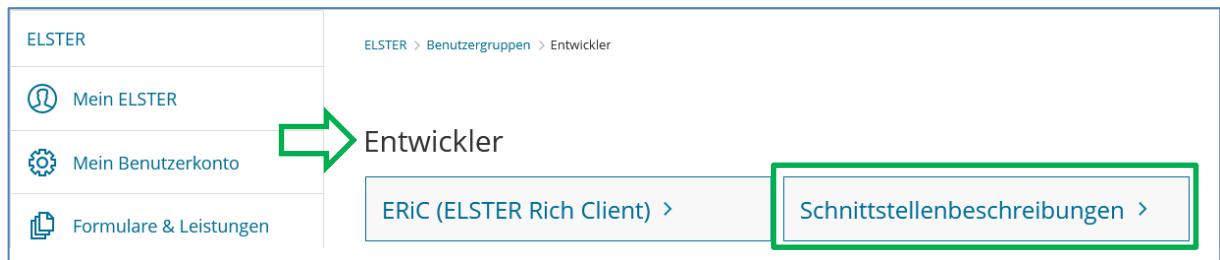
3.2.2.2 Download des Test-Zertifikats

Für den Testfall **Est_2020** benötigen Sie ein Test-Zertifikat. In den folgenden Schritten wird beschrieben, wo Sie Test-Zertifikate herunterladen können.

Hier finden Sie die Test-Zertifikate:

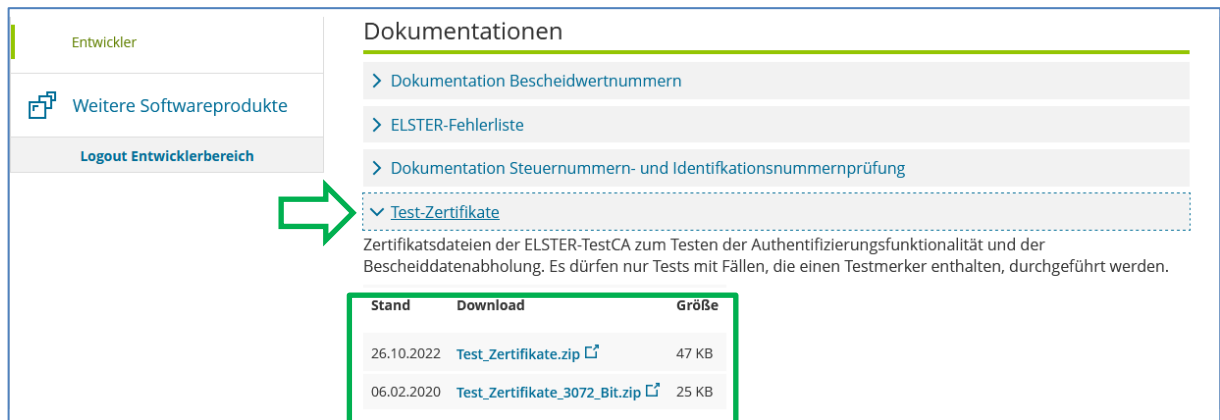
1. Klicken Sie auf <https://www.elster.de/elsterweb/entwickler/login> und melden Sie sich mit Ihren Hersteller-Logindaten im Entwicklerbereich an.

Abbildung 3-15 Nach Login im Entwicklerbereich



2. Klicken Sie auf **Schnittstellenbeschreibungen**.
3. Scrollen Sie zum Abschnitt **Dokumentationen** und laden Sie dort die **Test-Zertifikate** herunter.

Abbildung 3-16 Download der Test-Zertifikate



3.2.2.3 Fehlerbehandlung am Beispiel der ERiC API-Funktion `EricBearbeiteVorgang()`

Wenn ein Fehler bei der Verarbeitung aufgetreten ist, dann ist der Rückgabewert, in diesem Beispiel „rc“, ungleich `ERIC_OK`.

Wenn der Rückgabewert gleich `ERIC_GLOBAL_PRUEF_FEHLER` ist, dann enthält der Rückgabepuffer Fehlermeldungen aus der Plausibilitätsprüfung der Steuerdaten. Der Themenbereich „Plausibilitätsprüfung“ ist im nächsten Kap. [3.2.3](#) ausführlich beschrieben.

Mit der ERiC API-Funktion `EricHoleFehlerText()` können Sie den Fehlertext zum Fehlercode ermitteln. Verschaffen Sie sich einen Überblick über die ERiC Fehlercodes, indem Sie in der API-Referenz navigieren zu: **Dateien > Auflistung der Dateien > [eric_fehlercodes.h](#)**:

Abbildung 3-17 API-Referenz (html): ERiC Fehlercodes

Typdefinitionen	
typedef enum <code>eric_fehlercode</code> <code>eric_fehlercode_t</code>	
Aufzählungen	
enum	<pre> 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 , ... etc. </pre>

Ein optionales Element `<ElsterInfo>` in der XML-Serverantwort enthält Hinweise für den Endanwender, beispielsweise über den Ablauf des verwendeten Zertifikats. Diese können Sie in Ihrer Steuersoftware dem Endanwender zur Verfügung stellen.

3.2.3 Schritt 3 - Plausibilitätsprüfungen

In diesem Kapitel lernen Sie den Umgang mit fehlerhaften Datensätzen. Im nachfolgenden Beispiel werten Sie den API-Rückgabewert und den *ergebnisPuffer*, der die Fehlermeldung enthält, aus.

Zu Demozwecken einen Plausibilitätsfehler anlegen:

1. Ändern Sie in Ihren erstellten eXML-Daten²⁹ das Geburtsdatum im Hauptvordruck des Steuerpflichtigen wie folgt, damit ein Plausibilitätsfehler auftritt:

Abbildung 3-18 XML-Beispiel: Plausibilitätsfehler bei Geburtsdatum

```
<E0100401>05.17.1955</E0100401>
```

2. Wenn Sie jetzt „ericdemo“ in der Kommandozeile³⁰ ausführen, erhalten Sie folgende Fehlermeldung im *ergebnisPuffer*:

Abbildung 3-19 Konsolenausgabe

```
Sendestatus: Fehler während der Plausibilitätsprüfung, Datensatz nicht plausibel. Zur
Ermittlung der fehlgeschlagenen Plausibilitätsprüfungen muss der Rückgabepuffer
(Parameter "rueckgabeXmlPuffer") ausgewertet werden.

Rückgabe:
<?xml version="1.0" encoding="UTF-8"?>
<EricBearbeiteVorgang xmlns="http://www.elster.de/EricXML/1.1/EricBearbeiteVorgang">
  <FehlerRegelpruefung>
    <Nutzdatenticket>adfjasldkfuweor456asvs</Nutzdatenticket>
    [A] <Feldidentifikator>/Est1A[1]/Allg[1]/A[1]/E0100401[1]</Feldidentifikator>
    <Mehrfachzeilenindex>1</Mehrfachzeilenindex>
    <LfdNrVordruck>1</LfdNrVordruck>
    <VordruckZeilennummer>7</VordruckZeilennummer>
    <RegelName>formalePruefung</RegelName>
    [B] <FachlicheFehlerId>datumFormatFalsch</FachlicheFehlerId>
    <Text>Das Datum im Feld '$/Est1A[1]/Allg[1]/A[1]/E0100401[1]$_' ist nicht gültig.
      (Erforderliches Format: TT.MM.JJJJ).</Text>
  </FehlerRegelpruefung>
</EricBearbeiteVorgang>
```

[A] Kontext mit Feldkennung = „E0100401“

[B] Fehlercode = „datumFormatFalsch“


²⁹ Alternativ kann die Lösungsdatei [ESt_2020-Beispiel_Loesung.xml](#) verwendet werden.

³⁰ Unter Windows: `start-ericdemo.bat /x ESt_2020-Beispiel_Loesung.xml`

Umgang mit dem Plausibilitätsfehler:

1. Zunächst öffnen Sie die Logdatei [eric.log](#). Hier sehen Sie, dass der Aufruf mit dem Returncode „610001002“ beendet wurde.

Abbildung 3-20 eric.log: Fehlertext und Returncode

ERROR: Fehler während der Plausibilitätsprüfung, Datensatz nicht plausibel. Zur Ermittlung der fehlgeschlagenen Plausibilitätsprüfungen muss der Rückgabepuffer (Parameter "rueckgabeXmlPuffer") ausgewertet werden. 610001002 

2. Den korrespondierenden Fehlertext (rot markiert) zum Returncode „610001002“ sehen Sie in der API-Referenz bzw. in der Header-Datei [eric_fehlercodes.h](#).

Abbildung 3-21 API-Referenz (html): Korrespondierender Fehlertext und Returncode

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.

3. Programmatisch ist [EricHoleFehlerText\(\)](#) zu verwenden.

Wissenswertes zum Plausibilitätsfehler:

Beim Fehlercode = „datumFormatFalsch“ handelt es sich um einen formalen Fehler. Im Dokument [Zusatzinformationen_zur_Plausibilitaetspruefung.pdf](#)³¹ finden Sie die dazugehörige Prüfbedingung und Fehlerbeschreibung.

Die Plausibilitätsfehler werden unterschieden in:

- Formale Fehler innerhalb eines Feldes:
 - [Formale_Plausipruefungen_Steuerartuebergreifend.xml](#)³¹
 - [Zusatzinformationen_zur_Plausibilitaetspruefung.pdf](#)³¹
 - [Formale_Plausipruefungen.xml](#)³²
- Inhaltliche Fehler / Plausibilitäten:
 - [Dokumentation\Plausipruefungen\Erklaerungssteuern\ES\UFA10*.xml](#)

Das Dokument [Zusatzinformationen_zur_Plausibilitaetspruefung.pdf](#) informiert Sie auch über die in der obigen Dokumentation verwendeten Terminologie und Struktur, bzw. über die Regelsprache und Regeln.

³¹ Dokumentation\Plausipruefungen\

³² Dokumentation\Plausipruefungen\Erklaerungssteuern\

Öffnen Sie jetzt die Datei [Jahresdokumentation_10_2020.xml](#)³³, um einen Überblick über die inhaltlichen Fehler / Plausibilitätsfehler der **EST** für den Veranlagungszeitraum 2020 zu bekommen.

Für den Hauptvordruck und jede **EST** Anlage sind jeweils fünf Tabellenblätter (Kontexte, Felder, Regeln, Kennzahlen und Texte) vorhanden.

Das drittletzte Tabellenblatt „Allg. Information“ enthält eine Liste der Vordrucke mit der Angabe des maximal zulässigen Wertes für lfd. Nr. Vordruck, eine Legende über die verwendete farbliche Kennzeichnung und eine Liste inkl. Beschreibung der sogenannten Formatkennzeichen³⁴.

Das vorletzte Tabellenblatt „Formale Fehler“ listet die Fehler-IDs, Beschreibung und Fehlertexte der in der Regelsprache auftretenden formalen Fehler für diesen VZ auf.

Die Vielzahl der Plausibilitätsfehler in der Jahresdokumentation resultiert aus der Menge der Felder und deren Abhängigkeiten zueinander.

³³ Dokumentation\Plausipruefungen\Erklaerungssteuern\EST\UFA10

³⁴ siehe EHB, Kap. „Das ERiC Dokumentationspaket“

3.2.4 Schritt 4 - Übermittlung an den ELSTER Annahmeserver

Ist die Plausibilitätsprüfung fehlerfrei abgeschlossen, übermittelt der ERiC abhängig vom Anwendungsfall die Daten des Steuerpflichtigen zum ELSTER Annahmeserver. Dieser nimmt ebenfalls Dateneingangsprüfungen vor und meldet dann die Annahme oder Abweisung der Daten zurück an den ERiC. Die Antwort wird nun über den Returncode bzw. im Fehlerfall über den Rückgabepuffer der Funktion *EricBearbeiteVorgang()* an den Aufrufer gemeldet.

Öffnen Sie jetzt die zuvor verwendeten eXML-Daten und korrigieren Sie den Fehler im Datum. Führen Sie „ericdemo“ von der Kommandozeile erneut aus:

Abbildung 3-22 Konsolenausgabe

```

Sendestatus: Verarbeitung fehlerfrei.

Rückgabe:
<?xml version="1.0" encoding="UTF-8"?>
<EricBearbeiteVorgang xmlns="http://www.elster.de/EricXML/1.1/EricBearbeiteVorgang">
  <Erfolg>
    <Telenummer>SNN</Telenummer>
  </Erfolg>
  <Transfers>
    <Transfer>
      <TransferTicket>eh3122731gdbb250a8vcn7aacyu334rt</TransferTicket>
    </Transfer>
  </Transfers>
</EricBearbeiteVorgang>

Serverantwort:
<?xml version="1.0" encoding="UTF-8"?><Elster
xmlns="http://www.elster.de/elsterxml/schema/v11"><TransferHeader version="11">
<Verfahren>ElsterErklaerung</Verfahren><DatenArt>EST</DatenArt> <Vorgang>send-Auth</Vorgang>
<TransferTicket>eh3122731gdbb250a8vcn7aacyu334rt</TransferTicket>
<Testmerker>700000004</Testmerker><Empfaenger id="L"><Ziel>BY</Ziel>

```

- [A] War der Versand fehlerfrei (ERIC_OK), antwortet „ericdemo“ mit „Sendestatus: Verarbeitung fehlerfrei.“. Der *rueckgabeXMLPuffer* besteht im Erfolgsfall aus:
- der Telenummer als Zuordnungskriterium im Finanzamt (dreistellige Zeichenkette)
 - dem optionalen Ordnungsbegriff (nur bei Neuaufnahme³⁵)
- [B] Der Parameter *serverantwortXMLPuffer* besteht aus der Serverantwort im XML-Format (im obigen Screenshot abgeschnitten, Details im nächsten Kap. [3.2.4.1](#)).
- [C] Sie sehen, dass der Versand mit Zertifikat („send-Auth“) erfolgte.

Die Einträge in [eric.log](#) bestätigen Ihnen ebenfalls die erfolgreiche Datenübermittlung.

³⁵ Bei der Neuaufnahme werden Steuerdaten ohne Steuernummer übermittelt. Welche Datenarten dies unterstützen und welche Besonderheiten zu beachten sind, siehe EHB, Kap. „Neuaufnahmen (Steuerdaten ohne Steuernummer)“

3.2.4.1 Auswertung der Serverantwort im XML-Format

Im Beispiel des *serverantwortXMLPuffer* sehen Sie, dass kein Fehler vorliegt, der `<Code>` ist 0 und im `<Text>` wird bestätigt, dass die Daten erfolgreich angenommen wurden.

Abbildung 3-23 Serverantwort im XML-Format

```
<?xml version="1.0" encoding="UTF-8"?>
<Elster xmlns="http://www.elster.de/elsterxml/schema/v11">
  <TransferHeader version="11">
    ...
    ...
    <RC>
      <Rueckgabe>
        <Code>0</Code>
        <Text>Daten wurden erfolgreich angenommen.</Text>
      </Rueckgabe>
    ...
    ...
  </RC>
</TransferHeader>
<DatenTeil></DatenTeil>
</Elster>
```

Tritt ein Fehler auf und ist der Rückgabewert `ERIC_TRANSFER_ERR_XML_THEADER` oder `ERIC_TRANSFER_ERR_XML_NHEADER`, dann müssen die Logdatei `eric.log` und der *serverantwortXMLPuffer* ausgewertet werden. Hierzu kann entweder die Serverantwort selbst geparkt werden oder Sie verwenden *EricGetErrormessagesFromXMLAnswer()*³⁶. Diese Funktion liefert Ihnen alle Fehlermeldungen, den Returncode und das Transferticket. Mit dem Transferticket lässt sich jede Steuererklärung identifizieren, sie ist auch auf dem `ericprint.pdf`³⁷ aufgedruckt.

³⁶ Die API-Referenz enthält die Funktionssignatur und Funktionsbeschreibung.

³⁷ In der mitgelieferten Beispieldatei `ericprint.pdf` finden Sie das Transferticket auf jeder Seite am linken Rand vertikal aufgedruckt.

3.2.5 Schritt 5 - PDF-Erstellung

Nach einer fehlerfreien Verarbeitung wird abhängig vom Anwendungsfall ein PDF-Dokument erstellt. Stellen Sie sicher, dass ein schreibender Zugriff in das Arbeitsverzeichnis³⁸ möglich ist.

Öffnen Sie jetzt das erstellte PDF-Dokument [ericprint.pdf](#) im Arbeitsverzeichnis. Die Seite 1 sollte wie folgt aussehen:

Abbildung 3-24 Beispiel: ericprint.pdf

Finanzamt München (181) Überschusseinkünfte		Steuernummer 181/815/08155	Datum der Ausfertigung: 16.11.2023 Seite 1 von 4
*** Testfall ***		*** Testfall ***	2020
Hauptvordruck ESt 1 A			
Art der Erklärung			
Einkommensteuererklärung			
Festsetzung der Arbeitnehmer - Sparzulage			
Erklärung zur Feststellung des verbleibenden Verlustvortrags			
Allgemeine Angaben			
Steuerpflichtige Person. Nur bei Zusammenveranlagung: Ehemann oder Person A (Ehegatte A / Lebenspartner(in) A nach dem LPartG)			
7	Geburtsdatum	05.05.1955	
8	Name	Maier	
9	Vorname	Hans	
10	Titel, akademischer Grad	Dr.	
11	Religion	Römisch-katholisch	
12	Straße (derzeitige Adresse)	Musterstr.	
13	Hausnummer	55	
14	Hausnummerzusatz	c	
15	Postleitzahl (Inland)	55555	
16	Wohnort	Musterort	
17	Ausgeübter Beruf	Industriemechaniker	
18	Verheiratet / Lebenspartnerschaft begründet seit dem	02.08.2009	
Nur bei Zusammenveranlagung: Ehefrau oder Person B (Ehegatte B / Lebenspartner(in) B nach dem LPartG)			
Bitte füllen Sie die Zeilen 22 bis 26 nur aus, wenn die Adressangaben von den Zeilen 11 bis 15 abweichen			
19	Geburtsdatum	09.07.1988	
20	Name	Maier	
21	Vorname	Carolina	
22	Religion	Römisch-katholisch	
23	Ausgeübter Beruf	Floristin	
Bei Ehegatten / Lebenspartnern: Veranlagungsart			
24	Zusammenveranlagung		
Bankverbindung			
25	IBAN (inländisches Geldinstitut)	DE911000000001234567	
26		89	
27	Kontoinhaber ist die steuerpflichtige Person, nur bei Zusammenveranlagung: Ehemann / Person A		
28	Kontoinhaber ist die Ehefrau / Person B		
Antrag auf Festsetzung der Arbeitnehmer-Sparzulage: Steuerpflichtige Person / Ehemann / Person A			
29	Für alle vom Anbieter übermittelten Vermögensbildungsbescheinigungen wird die Festsetzung der Arbeitnehmer-Sparzulage beantragt	1 (= Ja)	
ELSTER		gedruckt mit ERiC-Print 39.2.4/ESr_2020 39.2.2	

Im Kap. [2.10](#) haben wir einen Testmerker in die eXML-Daten eingetragen, um den Datensatz als Testfall zu kennzeichnen. Der Testmerker führt auch dazu, dass das erzeugte PDF mit dem in Rot gedruckten „*** Testfall ***“ eine Kennzeichnung als Testfall erhält.

³⁸ Kann mit [EricInitialisiere\(\)](#) gesetzt werden, siehe API-Referenz.

3.3 Dritte Aufgabe: Neue und geänderte Felder in die Steuersoftware integrieren

Damit Sie Ihre Steuersoftware an einen neuen Veranlagungszeitraum oder Anmeldungszeitraum (AZ) anpassen können, sehen Sie sich folgende Dokumente an:

- In der Deltadokumentation³⁹ werden die Änderungen gegenüber dem letzten ERiC Update bzw. Patch dargestellt. Eine Deltadokumentation für einen VZ/AZ existiert aber nur dann, wenn es an dem VZ/AZ fachliche Änderungen zur vorangegangenen ERiC-Version gegeben hat.
- Die Jahresdokumentation⁴⁰ hingegen enthält die Änderungen gegenüber dem Vorjahres-Zeitraum aus dem letzten Patch des ERiC-Vorjahres-Release. Dabei sind sämtliche Änderungen der bisher für dieses ERiC-Release ausgelieferten Updates oder Patches mit enthalten.

Alle Änderungen (Feld- und Regeländerungen) in der Delta- oder/und der Jahresdokumentation sind farbig hervorgehoben. Eine programmatische Auswertung ist über die Tabellenspalte „Änderungsinformation“ möglich. Die Änderungen sind auch in den annotierten amtlichen Vordrucken farbig abgebildet.

³⁹ Dokumentation\Deltadokumentation*

⁴⁰ Dokumentation\Plausipruefungen*

3.3.1 Beispiel zum VZ-Wechsel für die ESt von 2016 auf 2017



HINWEIS:

Das Beispiel in diesem Kapitel ist nur dazu gedacht, das Prinzip zu veranschaulichen, und nicht als praktische Übung im Rahmen des Tutorial-Ablaufs.

Öffnen Sie den Vordruck [2017AnlKind021.png](#)⁴¹ (Anlage Kind, Seite 1), um Änderungen anhand der farbigen Markierung zu identifizieren. Beispielsweise ist die Feldkennung „0501513“ und „0501610“ neu hinzugekommen und der Beschreibungstext „Angaben für ein volljähriges Kind“ wurde geändert.

Hingegen sind die Feldkennungen „0501310“, „0501509“, „0501609“ und „0501702“ ab VZ 2017 nicht mehr möglich.

Abbildung 3-25 Ausschnitt aus Vordruck „2017AnlKind021.png“

15	Der Wohnsitz oder gewöhnliche Aufenthalt des anderen Elternteiles ist nicht zu ermitteln oder der Vater des Kindes ist amtlich nicht feststellbar	0501513 05 NEU! 1 = Ja
Angaben für ein volljähriges Kind		gelöscht: 0501310, 0501509, 0501609, 0501702
<p>Das Kind</p> <ul style="list-style-type: none"> – befand sich in einer Schul-, Hochschul- oder Berufsausbildung, – befand sich in einer Übergangszeit von höchstens vier Monaten (z. B. zwischen zwei Ausbildungsabschnitten), – konnte eine Berufsausbildung mangels Ausbildungsplatzes nicht beginnen oder fortsetzen und / oder – hat ein freiwilliges soziales oder ökologisches Jahr (Jugendfreiwilligendienstgesetz), einen europäischen / entwicklungspolitischen Freiwilligendienst, einen Freiwilligendienst aller Generationen (§ 2 Abs. 1a SGB VII), einen Internationalen Jugendfreiwilligendienst, Bundesfreiwilligendienst oder einen Anderen Dienst im Ausland (§ 5 Bundesfreiwilligendienstgesetz) geleistet. <p>(Folgt diese Abschnitte unmittelbar aufeinander, sind sie zu einem Zeitraum zusammenzufassen.)</p> <p>1. Zeitraum 2. Zeitraum</p> <p>vom bis vom bis</p>		
16	80 0501610 (MZI=12) NEU! TTMMJJJJJ	81 TTMMJJJJJ

⁴¹ Vordrucke\archive\2017\Erklärungssteuern\ESt\Grafiken_und_Erweiterungen_UFA10\

Die neue Feldkennung „0501610“ soll in der Steuersoftware aufgenommen werden. Hierzu müssen Sie das Feldformat kennen und wissen, ob es Abhängigkeiten zu anderen Feldern und Regeln gibt. Das erfahren Sie aus dem Dokument [Jahresdokumentation_10_2017.xml](#). bzw. [Deltadokumentation_10_2017.xml](#). Die „10“ im Dateinamen ist die Unterfallart (UFA). Diese kann dem EHB, Kap. „Unterstützte Fachverfahren und Daten- / Steuerarten“ entnommen werden.

Öffnen Sie jetzt das Dokument [Jahresdokumentation_10_2017.xml](#) mit Excel oder einer kompatiblen Anwendung.

Abbildung 3-26 Beispiel: Jahresdokumentation*.xml

Angaben_fuer_ein_volljaehrigen_Kind	0501408	Bezeichnung der Schul-, Hochschul- oder Berufsausbildung	12	String
Angaben_fuer_ein_volljaehrigen_Kind	0501408 (alt)	Bezeichnung der Schul-, Hochschul- oder Berufsausbildung	6	String
Angaben_fuer_ein_volljaehrigen_Kind	0501802	vom - bis	6	DatumBereich TT.MM.JJJJ-TT.MM.JJJJ mit Zusatzprüfung
Angaben_fuer_ein_volljaehrigen_Kind	0501905	vom - bis	2	DatumBereich TT.MM.JJJJ-TT.MM.JJJJ mit Zusatzprüfung
Angaben_fuer_ein_volljaehrigen_Kind	0502009	vom - bis	2	DatumBereich TT.MM.JJJJ-TT.MM.JJJJ mit Zusatzprüfung
Angaben_fuer_ein_volljaehrigen_Kind	0501610	Zeitraum vom - bis	12	DatumBereich TT.MM.JJJJ-TT.MM.JJJJ mit Zusatzprüfung
Angaben_fuer_ein_volljaehrigen_Kind	0501316	Ausbildungsabschnitt vom - bis	6	DatumBereich TT.MM.JJJJ-TT.MM.JJJJ mit Zusatzprüfung
Angaben_fuer_ein_volljaehrigen_Kind	0501508	vom - bis	6	DatumBereich TT.MM.JJJJ-TT.MM.JJJJ mit Zusatzprüfung
Angaben_fuer_ein_volljaehrigen_Kind	0501608	vom - bis	6	DatumBereich TT.MM.JJJJ-TT.MM.JJJJ mit Zusatzprüfung
Angaben_fuer_ein_volljaehrigen_Kind	0501702	vom - bis	6	DatumBereich TT.MM.JJJJ-TT.MM.JJJJ mit Zusatzprüfung
Angaben_zur_Erwerbstaeftigkeit_eines_volljaehrigen_Kindes	0502204	Das Kind hat bereits eine erstmalige Berufsausbildung oder ein Erststudium abgeschlossen (1 = Ja / 2 = Nein)	1	JaNein (1 2)
Angaben_zur_Erwerbstaeftigkeit_eines_volljaehrigen_Kindes	0502401	Falls Zeile 21 mit Ja beantwortet wurde: Das Kind war erwerbstätig (kein Ausbildungsdienstverhältnis) (1 = Ja / 2 = Nein)	1	JaNein (1 2)
Angaben_zur_Erwerbstaeftigkeit_eines_volljaehrigen_Kindes	0502401 (alt)	Falls Zeile 23 mit Ja beantwortet wurde: Das Kind war erwerbstätig (kein Ausbildungsdienstverhältnis) (1 = Ja / 2 = Nein)	1	JaNein (1 2)
Angaben_zur_Erwerbstaeftigkeit_eines_volljaehrigen_Kindes	0502501	Das Kind übte eine / mehrere geringfügige Beschäftigung(en) im Sinne der §§ 8, 8a SGB IV (sogenannter Minijob) aus (1 = Ja / 2 = Nein)	1	JaNein (1 2)
Angaben_zur_Erwerbstaeftigkeit_eines_volljaehrigen_Kindes	0502502	Beschäftigungszeitraum vom - bis	12	DatumBereich TT.MM-TT.MM mit Zusatzprüfung
Angaben_zur_Erwerbstaeftigkeit_eines_volljaehrigen_Kindes	0502616	Das Kind übte andere Erwerbstätigkeiten aus (1 = Ja / 2 = Nein)	1	JaNein (1 2)
Angaben_zur_Erwerbstaeftigkeit_eines_volljaehrigen_Kindes	0502617	Erwerbszeitraum vom - bis	12	DatumBereich TT.MM-TT.MM mit Zusatzprüfung

Suchen Sie nach der Feldkennung „0501610“ im Arbeitsblatt „Kind – Felder“. Neue Zeilen sind rot markiert, gelbe Zeilen signalisieren eine Änderung, graue und weiße Zeilen enthalten keinerlei Änderungen. Der gefundenen Zeile entnehmen Sie das Feldformat, die Beschreibung und stellen fest, dass es sich um ein optionales Feld handelt. Mit diesen Informationen können Sie das neue Feld in Ihre Steuersoftware, wie bereits in den vorigen Kapiteln beschrieben, aufnehmen und die XML-Daten erweitern.

Gehen Sie nun analog für alle weiteren geänderten Felder vor, achten Sie hierbei auf Abhängigkeiten zu anderen Feldern und Regeln. Validieren Sie das aktualisierte XML mit „ericdemo“, um diese Aufgabe abzuschließen.

3.4 Vierte Aufgabe: Den TransferHeader mit EricCreateTH() erstellen

Neben dem XML-Datenteil muss das Steuerdaten-XML einen TransferHeader enthalten. Zur Erstellung des TransferHeaders ist die Verwendung der API-Funktion *EricCreateTH()* empfohlen, da sie folgende Vorteile bietet:

- Die manuelle, fehlerträchtige Erstellung des TransferHeaders entfällt.
- Pflichtelemente und -werte werden automatisch erzeugt und korrekt befüllt, z. B. das Element `<Datei>` wird mit den korrekten Werten versorgt.
- Die richtige Reihenfolge der Elemente laut Schemadefinition wird gewährleistet.
- Der TransferHeader im XML bleibt schemakonform unabhängig vom ERiC Release.
- Es kann das gesamte Steuerdaten-XML oder nur der Datenteil im Parameter xml übergeben werden.

Die Verwendung von *EricCreateTH()* im C++ Beispielprogramm „ericdemo“ wird nachfolgend demonstriert.

3.4.1 Schritt 1 - XML-Eingabedatei erstellen

Entnehmen Sie aus der Datei `ESt_2020.xml` den `<DatenTeil>`-Block und speichern diesen in einer neuen Datei ab, z. B. in `ESt_2020_Datenteil.xml`.

Abbildung 3-27 Schematischer Aufbau der Datei `ESt_2020_Datenteil.xml`

```
<DatenTeil>
  <Nutzdatenblock>
    <NutzdatenHeader version="11">
      ...
    </NutzdatenHeader>
    <Nutzdaten>
      ...
    </Nutzdaten>
  </Nutzdatenblock>
</DatenTeil>
```

3.4.2 Schritt 2 - EricCreateTH() in ericdemo aufrufen

Der typedef, die Funktionszeiger, der Wrapper und die Implementierung der Proxymethode für **EricCreateTH()** sind bereits in ericdemo enthalten und können ohne Anpassung verwendet werden, siehe [eric.h](#) und [eric.cpp](#).

1. Fügen Sie die folgende neue Methode „erzeugeTransferHeader“ in [ericvorgang.cpp](#) am Ende ein. Der API-Aufruf von **EricCreateTH()** verwendet in diesem Beispiel feste Übergabewerte:

Abbildung 3-28 Ausschnitt aus ericvorgang.cpp

```
int EricVorgang::erzeugeTransferHeader(std::string& ergebnis)
{
    EricPuffer ergebnisPuffer(ericAdapter);

    int rc = ericAdapter.EricCreateTH(
        xmlDaten.c_str(),           // zuvor eingelesene Steuerdaten
        "ElsterErklaerung",        // Verfahren
        "EST",                      // Datenart
        "send-Auth",               // Vorgang
        "700000004",               // Testmarker
        "74931",                   // Verwenden Sie nur Ihre Hersteller-ID!
        "Softwaretester ERiC",     // Datenlieferant
        "MeineSteuerSoftware 2.4", // versionClient
        nullptr,                   // publicKey
        ergebnisPuffer.handle()); // Rückgabepufferhandle

    ergebnis = ergebnisPuffer.inhalt();

    if (rc == 0)
        xmlDaten = ergebnis;

    return rc;
}
```

2. Zur neuen Methode "EricVorgang::erzeugeTransferHeader" muss in [ericvorgang.h](#) hinter der Deklaration leseDatensatz() auch eine passende Methodendeklaration eingefügt werden:

Abbildung 3-29 Ausschnitt aus ericvorgang.h

```
/** @brief Reichere ein XML für eine Est_2020 mit einem Elster-Transferheader an
 */
int erzeugeTransferHeader(std::string& ergebnis);
```

3. Nach dem Einlesen des Datensatzes `vorgang leseDatensatz(argParser.getDatensatzDatei());` den folgenden Code mit dem neuen Methodenaufruf „createTH“ in [ericdemo.cpp](#) einfügen:

Abbildung 3-30 Ausschnitt aus ericdemo.cpp

```
// TransferHeader mit EricCreateTH() erstellen
System::titelZeile("Den TransferHeader mit EricCreateTH() erstellen.");
std::string ergebnisXML;

fehlerkode = vorgang.erzeugeTransferHeader(ergebnisXML);

if (fehlerkode == ERIC_OK)
    std::cout << ergebnisXML << std::endl;
```

4. Übersetzen und linken Sie ericdemo, um es als ausführbares Programm für die Konsole zu erstellen.
5. Führen Sie auf der Konsole folgenden Befehl aus:

```
starte-ericdemo.bat /v ESt_2020 /x ESt_2020_Datenteil.xml
```

und verfolgen Sie die Ausgaben von ericdemo.

Der Rückgabepuffer von **EricCreateTH()** enthält das mit dem **<TransferHeader>** angereicherte XML:

Abbildung 3-31 Konsolenausgabe: Beispiel für das Element <TransferHeader>

```
*** Lese die Datensatzdatei "ESt_2020_Datenteil.xml" mit Datenartversion "ESt_2020" ein ***
*** Sende und drucke den Datensatz ***
Sende mit Zertifikat: test-softidnr-pse.pfx
*** Den TransferHeader mit EricCreateTH() erstellen. ***

<?xml version="1.0" encoding="UTF-8"?>
<Elster xmlns="http://www.elster.de/elsterxml/schema/v11">
  <TransferHeader version="11">
    <Verfahren>ElsterErklaerung</Verfahren>
    <DatenArt>ESt</DatenArt>
    <Vorgang>send-Auth</Vorgang>
    <Testmerker>700000004</Testmerker>
    <SigUser>
      <Sig/>
    </SigUser>
    <Empfaenger id="L">
      <Ziel>CS</Ziel>
    </Empfaenger>
    <HerstellerID>74931</HerstellerID>
    <DatenLieferant>Softwaretester ERiC</DatenLieferant>
    <Datei>
      <Verschluesselung>CMSEncryptedData</Verschluesselung>
      <Kompression>GZIP</Kompression>
      <Erstellung>
        <Eric>
          <Version></Version>
        </Eric>
      </Erstellung>
    </Datei>
    <VersionClient>MeineSteuerSoftware 2.4</VersionClient>
  </TransferHeader>
  <DatenTeil>
    <Nutzdatenblock>
      <NutzdatenHeader version="11">
```

6. Kontrollieren Sie in der Konsole den erstellten **<TransferHeader>**-Block.

Der Aufruf von *EricCreateTH()* war erfolgreich: Die Übergabewerte sind zusammen mit den erstellten Pflichtelementen und -werten im *<TransferHeader>* zu finden.

4 ERiC optimal in die Steuersoftware am Beispiel ESt_2020 integrieren

Nach dem Abschluss der Entwicklungs- und Testphase der Steuersoftware wird der ERiC in die Steuersoftware integriert. Dieser Abschnitt zeigt anhand des Beispiels **ESt_2020**, welche ERiC Bibliotheken mindestens in die an den Endkunden auszuliefernde Steuersoftware zu integrieren sind, und welche Vorteile sich daraus ergeben.

Lesen Sie zuerst die Grundlagen im EHB, Kap. „Beste Strategie für die Integration einer ERiC Auslieferung in die Steuersoftware“.

Voraussetzungen für das Beispiel ESt_2020 für Windows, 64-Bit:

- Sie haben das ERiC Softwarepaket [ERiC-<version>-Windows-x86_64.jar](#)⁴² installiert. Für Informationen zur Auswahl des benötigten ERiC Softwarepakets siehe EHB, Kap. „Das ERiC Softwarepaket: ERiC-<version>-<os>“.
- Sie haben das Dokumentations-Paket [ERiC-<version>-Dokumentation.zip](#) installiert.

So ermitteln Sie die notwendigen ERiC Plugin-Bibliotheken:

1. Öffnen Sie die Datei `~~\Dokumentation\Datenartversionmatrix.xml` und ermitteln Sie für die `datenartVersion`⁴³ **ESt_2020** den ERiC Plugin-Bibliotheksnamen.

Abbildung 4-1 Zusammenhang zwischen Datenartversionmatrix.xml und plugins-Ordner

	A	B	C	D	E
1	Verfahren	Datenart(en)	Jahr / Version	datenartVersion-Parameter	Plugin-Bibliothek
15	ElsterErklärung	ESt	<Jahr>: ab 2012	ESt_<Jahr>	checkESt_<Jahr>

[A] Es ist die ERiC Plugin-Bibliothek [checkESt_2020.dll](#) zu verwenden.

[B] Die ERiC Plugin-Bibliothek [commonData.dll](#) wird grundsätzlich immer benötigt.⁴⁴

2. Alle anderen ERiC Plugin-Bibliotheken können Sie im Verzeichnis `plugins` löschen.⁴⁴

⁴² siehe EHB, Kap. „Inhalt des ERiC“

⁴³ siehe EHB, Kap. „datenartVersion – Definition und Verwendung“

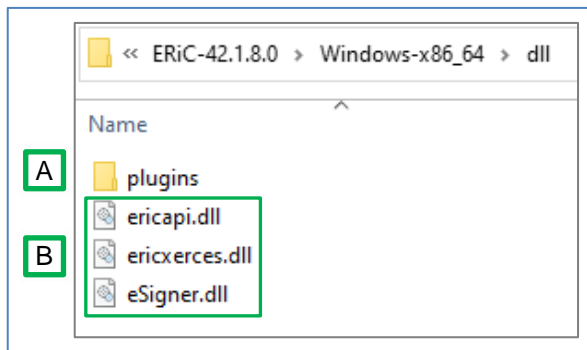
⁴⁴ siehe EHB, Kap. „Dynamische Programmbibliotheken“ → „ERiC Plugins im Verzeichnis „plugins““



HINWEIS:

Im Gegensatz zu den ERiC Plugin-Bibliotheken müssen die ERiC Basisbibliotheken komplett in die Steuersoftware übernommen werden. Siehe [Abbildung 4-2](#).

Abbildung 4-2 ERiC Basisbibliotheken



[A] ERiC Plugin-Bibliotheken im Verzeichnis plugins.

[B] ERiC Basisbibliotheken im Verzeichnis dll.

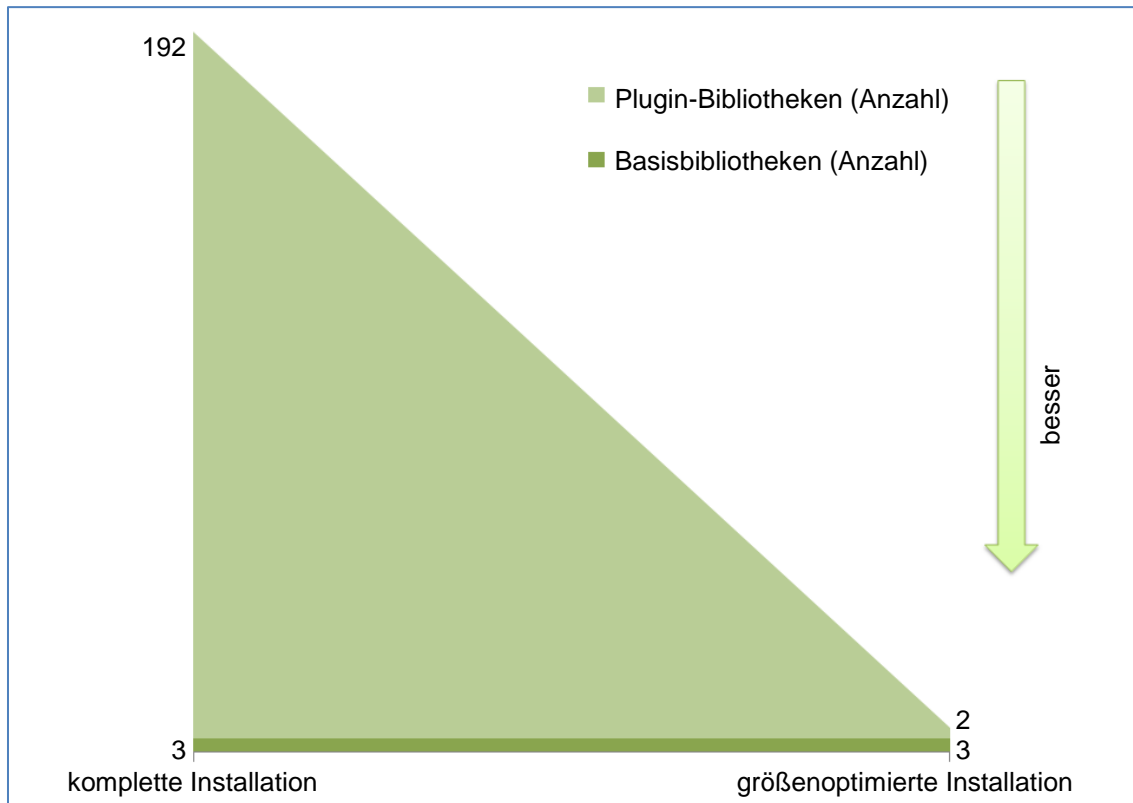
Mit den vorangegangenen Schritten wurden die ERiC Plugin-Bibliotheken für das Beispiel **ES_t_2020** auf die benötigte minimale Anzahl reduziert. Die daraus resultierenden Vorteile sind in der [Tabelle 4-1](#) und der [Abbildung 4-3](#) zusammengefasst.

Tabelle 4-1 Beispiel ES_t_2020: Komplette versus größenoptimierter Installation mit ERiC 37.2.6

	Komplette Installation Dateien / Speicherplatz	Gelöschte Dateien / Speicherplatz	Größenoptimierte Installation Dateien / Speicherplatz
Basisbiblio- theken	3 / 15,6 MiB ⁴⁵	0 / 0 MiB	3 / 15,6 MiB
Verzeichnis plugins	192 / 1,32 GiB	190 / 1,30 GiB	2 / 21,1 MiB
Summe	195 / 1,34 GiB	190 / 1,30 GiB	5 / 36,7 MiB

⁴⁵ 1 MiB = 1024 x 1024 Byte = 1.048.576 Byte, 1 MB = 1.000.000 Byte, siehe auch <https://de.wikipedia.org/wiki/Bin%C3%A4rpr%C3%A4fix> und <https://de.wikipedia.org/wiki/Byte#Vergleich>

Abbildung 4-3 Anzahl ERiC Plugin-Bibliotheken und Basisbibliotheken der kompletten gegenüber der größenoptimierten ERiC Installation



Es wird empfohlen, die Vorteile der größenoptimierten ERiC Installation zu nutzen:

- Der benötigte Speicherplatz wird reduziert.
- Der ERiC Ladevorgang wird beschleunigt, ebenso das Entladen.
- Die ERiC Funktionalität wird auf den gewünschten Umfang beschränkt.

5 Fortschrittcallbacks am Beispiel ericdemo implementieren

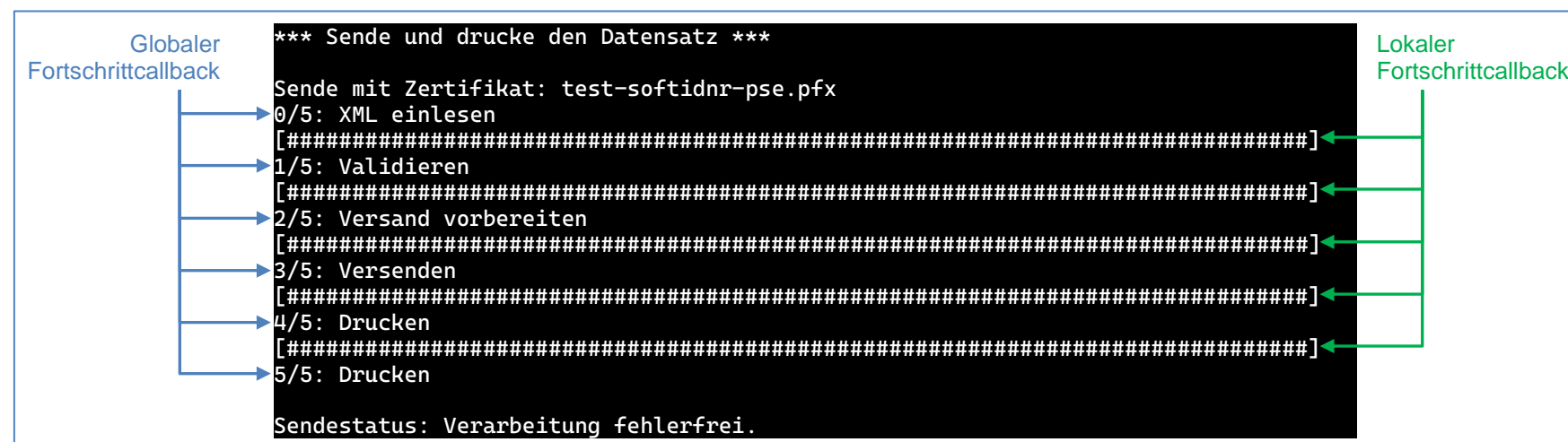
Die Verarbeitungsschritte von [EricBearbeiteVorgang\(\)](#) und deren Fortschritte können dem Endanwender durch Callbackfunktionen angezeigt werden.

Lesen Sie hierzu die Grundlagen im EHB, Kap. „Funktionen für Fortschrittcallbacks“, und informieren Sie sich über die zugehörigen Funktionssignaturen und Typen in der API-Referenz ([ericapi.h](#) und [eric_types.h](#)).

Die folgende Abbildung zeigt das gewünschte Ergebnis in der Beispielanwendung ericdemo:

- Der globale Fortschrittcallback soll die Variable *pos* und *max* mit einem von der *id* abhängigen Text ausgeben.
- Der lokale Fortschrittcallback soll eine einfach textbasierte Fortschrittanzeige implementieren.

Abbildung 5-1 Die Ausgaben der Fortschrittcallbacks in ericdemo



Sehen Sie sich im Folgenden die entsprechenden Codeausschnitte der Implementation in ericdemo an.

Der Fortschritt wird von der Klasse *CallbackHandler* auf der Konsole visualisiert. Die Funktion *mapId()* ordnet jeder *id* eines Verarbeitungsschritts von *EricBearbeiteVorgang()* einen passenden Text zu.

Abbildung 5-2 Fortschrittcallbacks am Beispiel der „callbackhandler.cpp“

```
std::string mapId(uint32_t id) {
    switch (id) {
        case ERIC_FORTSCHRITTCALLBACK_ID_EINLESEN:
            return "XML einlesen";
        case ERIC_FORTSCHRITTCALLBACK_ID_VORBEREITEN:
            return "Versand vorbereiten";
        case ERIC_FORTSCHRITTCALLBACK_ID_VALIDIEREN:
            return "Validieren";
        case ERIC_FORTSCHRITTCALLBACK_ID_SENDEN:
            return "Versenden";
        case ERIC_FORTSCHRITTCALLBACK_ID_DRUCKEN:
            return "Drucken";
        default:
            throw std::runtime_error("Unbekannte ID");
    }
}

CallbackHandler::CallbackHandler(const Eric& myEric) { ... }

CallbackHandler::~~CallbackHandler() { ... }

[A] void CallbackHandler::globalerFortschritt(uint32_t id, uint32_t pos, uint32_t max) const {
    (System::out << pos << "/" << max << ": " << mapId(id) << std::endl)(*stdout);
}

[B] void CallbackHandler::fortschritt(uint32_t id, uint32_t pos, uint32_t max) {
    if (letzteId != 0 && letzteId != id) {
        (System::out << std::string(78 - letzteSpalte, '#') << ']' << std::endl)(*stdout);
        letzteId = 0;
        letzteSpalte = 0;
        fortschritt(id, pos, max);
    } else {
        if (pos == 0) {
            (System::out << '[')(*stdout);
            letzteId = id;
            letzteSpalte = 0;
        } else if (pos == max) {
            (System::out << std::string(78 - letzteSpalte, '#') << ']' << std::endl)(*stdout);
            letzteId = 0;
            letzteSpalte = 0;
        } else {
            unsigned int spalte = (pos * 78) / max;
            (System::out << std::string(spalte - letzteSpalte, '#'))(*stdout);
            letzteSpalte = spalte;
        }
    }
}
```

[A] Die Methode *globalerFortschritt()* fügt den Inhalt der Variablen *pos*, *max* und den Text aus *mapId()* zu einer Ausgabezeile zusammen, beispielsweise „0/5: XML Einlesen“.

[B] Die Methode *fortschritt()* erstellt die Fortschrittsanzeige innerhalb eines Verarbeitungsschrittes, beispielsweise „[#####]“.

Der Aufruf der Methoden `globalerFortschritt()` und `fortschritt()` aus [Abbildung 5-2](#) oben erfolgt in der `callbackhandler.cpp` jeweils über die Callbackfunktionen `globalerFortschrittAdapter()` und `fortschrittAdapter()`.

Diese Callbackfunktionen sind erforderlich, weil die ERiC-Schnittstelle nicht objektorientiert sein darf. Daher können die Methoden nicht direkt am `CallbackHandler` aufgerufen werden.

Die beiden Callbackfunktionen erwarten im zweiten Parameter `userData` einen Zeiger auf den `CallbackHandler`, an dem die Methoden `globalerFortschritt()` und `fortschritt()` aufgerufen werden:

Abbildung 5-3 Die Callbackfunktionen `globalerFortschrittAdapter()` und `fortschrittAdapter()`

```
static void STDCALL globalerFortschrittAdapter(uint32_t id, uint32_t pos, uint32_t max, void *userData) {
    const CallbackHandler *const handler = reinterpret_cast<const CallbackHandler *>(userData);
    handler->globalerFortschritt(id, pos, max);
}

static void STDCALL fortschrittAdapter(uint32_t id, uint32_t pos, uint32_t max, void *userData) {
    CallbackHandler *const handler = reinterpret_cast<CallbackHandler *>(userData);
    handler->fortschritt(id, pos, max);
}
```

Damit [`EricBearbeiteVorgang\(\)`](#) die beiden Callbackfunktionen aufruft und ihnen dabei den `CallbackHandler` übergibt, sind die Funktionen (erster Parameter) und der Handler (zweiter Parameter) mit den API-Funktionen [`EricRegistriereGlobalenFortschrittCallback\(\)`](#) bzw. [`EricRegistriereFortschrittCallback\(\)`](#) zu registrieren:

Abbildung 5-4 Callbacks anmelden

```
// Callbacks fuer Fortschrittsbalken und Nachrichten anmelden
eric.EricRegistriereGlobalenFortschrittCallback(globalerFortschrittAdapter, this);
eric.EricRegistriereFortschrittCallback(fortschrittAdapter, this);
```

Abbildung 5-5 Das Abmelden der beiden Callbackfunktionen im Destructor nicht vergessen!

```
// Callbacks Abmelden
eric.EricRegistriereGlobalenFortschrittCallback(nullptr, nullptr);
eric.EricRegistriereFortschrittCallback(nullptr, nullptr);
```

Damit sind die wichtigsten Codestellen in `ericdemo` für die Fortschrittcallbacks betrachtet. Die Ausgabe erfolgt wie in [Abbildung 5-1](#) oben zu sehen.

6 Datenabholung: Allgemeiner Abhol-Prozess (nicht für VaSt-Belege)

Die wichtigsten Verarbeitungsschritte für die meisten Datenarten sind bereits im Kap. 3 anschaulich beschrieben. Die Datenabholung mit Anhängen und die Auswertung der Antwort erfolgt in mehreren Schritten:

- XML-Anfrage erstellen, ob Daten abgeholt werden können, siehe Kap. [6.1](#)
- Serverantwort auswerten, siehe Kap. [6.2](#)
- Herunterladen der bereitgestellten Daten vom OTTER-Server, siehe Kap. [6.3](#)
- Bestätigung der abgeholten Daten, siehe Kap. [6.4](#)

Diese Schritte veranschaulichen wir mithilfe der C++ Beispielprogramme „ericdemo“ und „ottodemo“.



HINWEIS:

Während Sie die vorherigen Kapitel durchgearbeitet haben, haben Sie bestimmte Änderungen an der ericdemo vorgenommen. Um die folgenden Kapitel auszuprobieren, benötigen Sie jedoch eine unveränderte ericdemo.

Alternativ müssen Sie mindestens die Änderungen aus Kap. [3.4.2](#) rückgängig machen oder den Code an die neue Datenart anpassen.

Bevor Sie mit dem nachfolgenden, praktischen Teil beginnen:

1. Machen Sie sich mit den Grundlagen der Datenabholung und den Anhängen vertraut:
 - Siehe EHB, Kap. „ElsterDatenabholung“.
 - Siehe EHB, Kap. „Anhänge im Elster-XML“ mit Unterkap. „Rückmeldung zu den Anhängen“.
 - Siehe EHB, Kap. „Otto und OTTER“ mit Unterkap. „Datenabholung mit Otto“.
2. Öffnen Sie die folgenden Dateien im Ordner `~~\Dokumentation\Tutorial\Beispiele\`
 - Die Beispieldatei [PostfachAnfrage.xml](#) und
 - Die Beispieldatei [PostfachBestaetigung.xml](#)
3. Ersetzen Sie in den *.xml-Beispieldateien im Element `<HerstellerID>` den Inhalt durch die Ihnen zugeteilte Hersteller-ID.
4. Öffnen Sie als Referenz zum Nachschlagen folgende Dateien aus den Unterordnern von `~~\Dokumentation\Schnittstellenbeschreibungen\Sonstige\ElsterDatenabholung_31\...`
 - Die Schnittstellenbeschreibung [ElsterDatenabholung-V<version>.pdf](#)
 - Die Schemadateien *.xsd
5. Für die Übermittlung der Beispieldaten mit dem Vorgang „send-Auth“ verwenden Sie, wenn möglich, Ihr eigenes Steueridentifikationsnummer (IdNr) Zertifikat.

- Nachfolgend werden Platzhalter für das Zertifikat und die zugehörige PIN verwendet, und zwar `<mein_idnr_zertifikat>` und `<meine_pin>`.
- Ersatzweise können Sie das Testzertifikat [test-softidnr-pse.pfx](#) verwenden, das für das Tutorial bereitgestellt wird. Da dieses Testzertifikat aber von mehreren Personen gleichzeitig verwendet werden kann, sind Seiteneffekte, insbesondere bei der Datenabholung nicht auszuschließen.

6.1 XML-Anfrage erstellen, ob Daten abgeholt werden können

Im EHB, Kap. „ElsterDatenabholung“ inklusive der Unterkapitel haben Sie erfahren, dass das Verfahren **ElsterDatenabholung** und die Datenart **PostfachAnfrage** in der XML-Anfrage zu verwenden sind.

XML-Anfrage erstellen und versenden:

1. Speichern Sie die Beispieldatei [PostfachAnfrage.xml](#) jetzt in diesem Ordner:
`~~\Windows-x86_64\Beispiel\ericdemo-cpp\`
2. Um abholbare Daten anzufragen, definieren Sie in der `*.xml`-Beispieldatei den Block der `<Nutzdaten>` zum Beispiel wie folgt:

Abbildung 6-1 XML-Beispiel: Elemente `<Nutzdaten>` und `<Datenabholung>`

```
...
<Nutzdaten>
  <Datenabholung xmlns="http://finkonsens.de/elster/elsterdatenabholung/v3"
    version="31">
    [A] <PostfachAnfrage einschraenkung="alle" max="1000">
      <DatenartBereitstellung name="DivaBescheidEst"/>
    </PostfachAnfrage>
  </Datenabholung>
</Nutzdaten>
...
```

- [A] **Für die Datenabholung gilt folgende Regel:** Als Daten-Abholer müssen Sie für Ihre **PostfachAnfrage** genau die Datenart verwenden, mit welcher der Daten-Ersteller die Bereitstellung durchgeführt hat.

Eine Liste dieser Bereitstellungsdatenarten, auf die via ERiC im Verfahren **Elster-Datenabholung** eine **PostfachAnfrage** durchgeführt werden kann, finden Sie im EHB, Kap. „Bereitstellungsdatenarten“.

- [B] **Konkrete Umsetzung im XML:** Im Element `<DatenartBereitstellung>` ist im Attribut `name = "..."` die Bereitstellungsdatenart einzutragen.

3. Um die XML-Anfrage zu versenden, führen Sie an der Kommandozeile aus:

```
starte-ericdemo.bat /v PostfachAnfrage_31 /x PostfachAnfrage.xml ^
/c <mein_idnr_zertifikat>.pfx /p <meine_pin> /t 0
```

- /t 0** Beachten Sie, dass im Rahmen der **ElsterDatenabholung** beim ersten Aufruf von **EricBearbeiteVorgang()** das Transferhandle **/t** mit **0** zu übergeben ist.
- Der Rückgabewert des Transferhandles wird für alle weiteren Aufrufe von **EricBearbeiteVorgang()** benötigt.
- /<option>** Siehe die `~~\Windows-x86_64\Beispiel\ericdemo-cpp\Liesmich.txt` für eine Beschreibung der Optionen.
- ^** Zeilenfortsetzungszeichen in Batch-Kommandos. Achten Sie darauf, dass nach dem **^** KEIN Leerzeichen folgt.

Ergebnis:

Die Beispieldatei **PostfachAnfrage.xml** wird versendet und an der Konsole wird folgende Ausgabe angezeigt:

Abbildung 6-2 Konsolenausgabe: Transferhandle und Serverantwort

```
Sende mit Zertifikat: test-softidnr-pse.pfx
0/3: XML einlesen
[#####]
1/3: Validieren
[#####]
2/3: Versenden
[#####]
3/3: Versenden

Sendestatus: Verarbeitung fehlerfrei.
[A] Transferhandle: 977163844

Rückgabe:
<?xml version="1.0" encoding="UTF-8"?>
<EricBearbeiteVorgang xmlns="http://www.elster.de/EricXML/1.1/EricBearbeiteVorgang">
  <Erfolg/>
  <Transfers>
    <Transfer>
      <TransferTicket>eh2914jvs125zq32v001tb16makzxf8d</TransferTicket>
    </Transfer>
  </Transfers>
</EricBearbeiteVorgang>
[B] Serverantwort:
```

[A] Rückgabewert des Transferhandles **/t**

[B] Die Serverantwort aus der obigen Konsolenausgabe wird nachfolgend zwecks besserer Lesbarkeit mit XML-Syntax-Highlighting dargestellt. Siehe Kap. [Serverantwort auswerten](#).

6.2 Serverantwort auswerten

1. Notieren Sie sich aus der oben gezeigten Konsolenausgabe den Rückgabewert des Transferhandles */t* (siehe [A] in [Abbildung 6-2](#)).
2. Notieren Sie sich aus der unten gezeigten Serverantwort folgende IDs:
 - **Bereitstellungs-ID**: im Element `<Bereitstellung id = "...">`, siehe [C] unten.
 - **Objekt-ID** (des Anhangs): im Element `<DateiReferenzId>`, siehe [D] unten.
In diesen Elementen sind die Objekt-IDs zu finden, unter denen Sie **Anhänge** abholen können.

Abbildung 6-3 Serverantwort mit XML-Syntax-Highlighting

```

...
<PostfachAnfrageAntwort einschraenkung="alle" max="1000">
  <DatenartBereitstellung name="DivaBescheidEst" anzahltreffer="1">
    <Bereitstellungen>
      <Bereitstellung id="be2156xz75hi9y32ekgb2kpewrvxemma" ← [C]
        bereitstellungsticket="pu2151byebuejp4iuh4r8kq8dxmu4f2y"
        groesse="63320">
        <MetaInformationen> ...</MetaInformationen>
        <Anhaenge>
          <Anhang>
            <Dateibezeichnung>9198030000065.BES.000004.0000001.PDF</Dateibezeichnung>
            <Dateityp>application/pdf</Dateityp>
            [D] → <DateiReferenzId>6e4cb8e7-8339-4081-b0f3-7871ef0ee7dd</DateiReferenzId>
            <DateiGroesse>63320</DateiGroesse>
          </Anhang>
        </Anhaenge>
      </Bereitstellungen>
    </DatenartBereitstellung>
  </PostfachAnfrageAntwort>
...

```

- Für die Abholung von **Datenpaketen** hingegen benötigen Sie die Objekt-IDs, die im Element `<ReferenzId>` zu finden sind.

Ergebnis:

Wie die Konsolenausgabe oben gezeigt hat, wurde unsere **PostfachAnfrage** im Verfahren **ElsterDatenabholung** fehlerfrei verarbeitet und an den Server übermittelt.

Der Serverantwort können wir entnehmen, dass ein (1) Anhang zur Abholung bereitsteht, siehe Attribut `anzahltreffer = "1"` im Element `<DatenartBereitstellung>`.

Das Konsolenfenster bzw. die Serverantwort halten folgende wichtige Rückgabewerte bereit:

- Das Transferhandle: `977163844`
- Die Bereitstellungs-ID: `be2156xz75hi9y32ekgb2kpewrvxemma`
- Die Objekt-ID: `6e4cb8e7-8339-4081-b0f3-7871ef0ee7dd`

Diesen Rückgabewerten werden wir im weiteren Verlauf der Datenabholung immer wieder begegnen. Denn nur mit ihnen kann die Abholung und die zwingend erforderliche Bestätigung der Abholung durchgeführt werden.

6.3 Bereitgestellte Daten vom OTTER-Server herunterladen

Im EHB, Kap. „Datenabholung mit Otto“ haben Sie erfahren, dass die abzuholenden Daten direkt vom **OTTER-Server** herunterzuladen sind. Des Weiteren haben Sie im EHB erfahren, dass **für den Download** die **otto**-Bibliothek von ERiC zum Einsatz kommt.

Um den gewünschten Anhang herunterzuladen zu können, benötigen Sie die Objekt-ID. In unserem Beispiel die Objekt-ID `6e4cb8e7-8339-4081-b0f3-7871ef0ee7dd`

Den bereitgestellten Anhang vom OTTER-Server herunterladen:

1. Wechseln Sie jetzt in den Ordner `~~\Windows-x86_64\Beispiel\ottodemo-cpp\`
2. Um die **otto**-Bibliothek aufzurufen und den Anhang vom OTTER-Server herunterzuladen, führen Sie an der Kommandozeile aus:

```
starte-ottodemo.bat /i <hersteller_id> /c <mein_idnr_zertifikat>.pfx ^
                    /p <meine_pin> /o <objekt_id> /e empfang.pdf
```

- /o <objekt_id>**
- Für den Download eines Anhangs die Objekt-ID aus dem Element **<DateiReferenzId>** angeben.
 - Für den Download eines Datenpakets die Objekt-ID aus dem Element **<ReferenzId>** angeben.

/<option> Siehe die `~~\Windows-x86_64\Beispiel\ottodemo-cpp\Liesmich.txt` für eine Beschreibung der Optionen.

^ Zeilenfortsetzungszeichen in Batch-Dateien (*.bat). Achten Sie darauf, dass nach dem **^** KEIN Leerzeichen folgt.

Ergebnis:

Die ottodemo wird gestartet und unter der angegebenen Objekt-ID wird der Anhang vom OTTER-Server heruntergeladen. An der Konsole wird folgende Ausgabe angezeigt:

Abbildung 6-4 Konsolenausgabe

```
*** Hole Datei von OTTER ***
Objekt-ID: 6e4cb8e7-8339-4081-b0f3-7871ef0ee7dd
Speichere Daten in: empfang.pdf
Empfange Daten ..
81397 Byte erfolgreich abgeholt.
Bitte druecken Sie die Eingabetaste
```

6.4 XML-Bestätigung senden, dass Daten erfolgreich abgeholt wurden

Im EHB, Kap. „ElsterDatenabholung“ inklusive der Unterkapitel haben Sie erfahren, dass eine erfolgreiche Datenabholung bestätigt werden muss. Dabei kommt wieder das Verfahren **ElsterDatenabholung** zum Einsatz, aber diesmal mit der Datenart **PostfachBestaetigung**.

Wichtiger Hinweis:

Nach erfolgreicher Abholung muss zwingend eine Bestätigung innerhalb von 24 Stunden erfolgen.

Softwarehersteller, die dies nicht beachten, müssen mit einer Sperrung rechnen!

Um die erfolgreiche Abholung gegenüber dem Server zu bestätigen, müssen Sie in der als Nächstes zu versendenden *.xml-Datei die Bereitstellungs-ID angeben. In unserem Beispiel:

- Die Bereitstellungs-ID: *be2156xz75hi9y32ekgb2kpewrvxemma*

Für die Übermittlung dieser *.xml-Datei an den Server benötigen Sie das Transferhandle. In unserem Beispiel:

- Das Transferhandle: *977163844*

Die erfolgreiche Datenabholung bestätigen:

1. Speichern Sie die Beispieldatei **PostfachBestaetigung.xml** jetzt in diesen Ordner:
~~\Windows-x86_64\Beispiel\ericdemo-cpp\
2. Definieren Sie den Block **<PostfachBestaetigung>** wie folgt:

Abbildung 6-5 XML-Beispiel: Element <PostfachBestaetigung>

```

...
<Nutzdaten>
  <Datenabholung xmlns="http://finkonsens.de/elster/elsterdatenabholung/v3"
    version="31">
    [A] <PostfachBestaetigung>
      <Bereitstellungen>
        <Bereitstellung id="be2156xz75hi9y32ekgb2kpewrvxemma"/> [B]
      </Bereitstellungen>
    </PostfachBestaetigung>
  </Datenabholung>
</Nutzdaten>
...

```

[A] Für die Bestätigung der Datenabholung gilt folgende Regel:

In der **PostfachBestaetigung** verwenden Sie wieder dieselbe Bereitstellungs-ID, die Sie in der Serverantwort auf Ihre vorherige **PostfachAnfrage** erhalten haben.

[B] Konkrete Umsetzung im XML: Geben Sie die Bereitstellungs-ID im Element **<Bereitstellung id = "...">** ein.

3. Um die XML-Bestätigung an den Server zu versenden, führen Sie an der Kommandozeile aus:

```
starte-ericdemo.bat /v PostfachBestaetigung_31 /x PostfachBestaetigung.xml ^
/c <mein_idnr_zertifikat>.pfx /p <meine_pin> /t 977163844
```

- /t <wert>** Der Rückgabewert des Transferhandles in diesem Beispiel.
- /<option>** Siehe die `~~\Windows-x86_64\Beispiel\ericdemo-cpp\Liesmich.txt` für eine Beschreibung der Optionen.
- ^** Zeilenfortsetzungszeichen in Batch-Dateien (*.bat). Achten Sie darauf, dass nach dem ^ KEIN Leerzeichen folgt.

Ergebnis:

Sie können der Serverantwort **[A]** unten entnehmen, dass unser Versand der zwingend erforderlichen XML-Bestätigung erfolgreich war. Alle Schritte der Datenabholung sind somit ausgeführt worden.

Abbildung 6-6 Konsolenausgabe

The image shows a console window with the following output:

```
Sendestatus: Verarbeitung fehlerfrei.
Transferhandle: 977163844
Rückgabe:
<?xml version="1.0" encoding="UTF-8"?>
<EricBearbeiteVorgang xmlns="http://www.elster.de/EricXML/1.1/EricBearbeiteVorgang">
  <Erfolg/>
  <Transfers>
    <Transfer>
      <TransferTicket>eh2845ochmjrg04q0j30x6dzuuh0oia</TransferTicket>
    </Transfer>
  </Transfers>
</EricBearbeiteVorgang>
```

A green box labeled **A** highlights the text "Serverantwort:" in the console output.

A green arrow points from box **A** to a second window showing the XML data with syntax highlighting:

```
...
<PostfachBestaetigungAntwort>
  <Bereitstellungen>
    <Bereitstellung id="be2156xz75hi9y32ekgb2kpewrvxemma">
      <Rueckgabe>
        <Code>0</Code>
        <Text>Der Empfang der Daten wurde erfolgreich bestaetigt.</Text>
      </Rueckgabe>
    </Bereitstellung>
  </Bereitstellungen>
</PostfachBestaetigungAntwort>
</Datenabholung></Nutzdaten></Nutzdatenblock></DatenTeil></Elster>
```

Box **B** is located next to the first line of the highlighted XML snippet.

- [B]** Die Serverantwort aus der obigen Konsolenausgabe wird zwecks besserer Lesbarkeit mit XML-Syntax-Highlighting dargestellt.

7 Zusammenfassung

In diesem Tutorial lernten Sie mit dem Beispielprogramm „ericdemo“ und dem **ES**t-Beispiel ein Steuerprogramm zu erstellen, die Steuerdaten mit ERiC zu verarbeiten und die Änderungen, wie z. B. einen VZ-Wechsel oder eine Datenabholung mit den Beispielprogrammen „ericdemo“ und „ottodemo“, in Ihre Software zu integrieren.

Das **ES**t-Beispiel des Tutorials ist minimalistisch, vermittelt aber das Handwerkszeug, um selbständig alle weiteren Anlagen, die Sie in Ihrer Steuersoftware anbieten wollen, zu implementieren.

Für die Verarbeitung und Einbindung anderer Datenarten in Ihre Software können Sie ähnlich vorgehen. Das EHB unterstützt Sie hierbei und verweist auf weitere, notwendige Dokumente sowie das Herstellerforum im Internet.

Mit der API-Referenz erhalten Sie einen Überblick über die ERiC API-Funktionen und Datenstrukturen, die Sie in Ihre Steuersoftware integrieren können.

8 Tabellenverzeichnis

Tabelle 1-1	Typographische Konventionen	4
Tabelle 2-1	Dateien und Dokumente des ERiC-Tutorials	13
Tabelle 3-1	Beispieldaten für 2x „Spenden und Mitgliedsbeiträge“	34
Tabelle 4-1	Beispiel ESt_2020: Komplette versus größenoptimierter Installation mit ERiC 37.2.6	57

9 Abbildungsverzeichnis

Abbildung 2-1	Initiale Registrierung	7
Abbildung 2-2	Anmeldung (1 von 2).....	8
Abbildung 2-3	Anmeldung (2 von 2).....	8
Abbildung 2-4	Antragsformular für eine Hersteller-ID aufrufen	9
Abbildung 2-5	Downloads (Teil 1 von 2).....	10
Abbildung 2-6	Downloads (Teil 2 von 2).....	11
Abbildung 2-7	Nach dem Entpacken.....	12
Abbildung 2-8	Hersteller-ID eingeben zum Ausführen der ottodemo.....	15
Abbildung 2-9	Die Kernfunktionen des ERiC.....	16
Abbildung 2-10	Auszug aus der Datei „Est_2020.xml“	17
Abbildung 2-11	Ausschnitt aus ericvorgang.cpp: API-Funktionsaufruf	18
Abbildung 2-12	Ausschnitt aus ericvorgang.cpp: ERiC Druckeinstellungen	19
Abbildung 2-13	ERiC API-Referenz (html): Beispiel einer Funktionssignatur	20
Abbildung 2-14	ELSTERFORUM für Entwickler: Startseite nach dem Login.....	21
Abbildung 3-1	Erstellung der ESt XML-Eingangsdaten	23
Abbildung 3-2	Ausschnitt aus „2020Est1A011.png“	24
Abbildung 3-3	XML-Beispiel: Struktur der ELSTER eXML-Daten für die ESt	26
Abbildung 3-4	XML-Beispiel: Nutzdatenstruktur für eXML-Daten	27
Abbildung 3-5	Ausschnitt aus E10-2020.html: Schema für Est_2020	28
Abbildung 3-6	Ausschnitt aus der ESt Jahresdokumentation 2020	30
Abbildung 3-7	Hauptvordruck ESt.....	32
Abbildung 3-8	Vordruck	35
Abbildung 3-9	Anlage Sonderausgaben.....	35
Abbildung 3-10	Datenverarbeitung mit ERiC.....	37
Abbildung 3-11	Ausschnitt aus eric.cpp	38
Abbildung 3-12	Ausschnitt aus eric.cpp	38
Abbildung 3-13	Ausschnitt aus ericvorgang.cpp: Aufruf von EricBearbeiteVorgang()....	39
Abbildung 3-14	Ausschnitt aus ericvorgang.cpp: Definition der ERiC Druckfunktionalität.....	40
Abbildung 3-15	Nach Login im Entwicklerbereich	41
Abbildung 3-16	Download der Test-Zertifikate	41
Abbildung 3-17	API-Referenz (html): ERiC Fehlercodes.....	42
Abbildung 3-18	XML-Beispiel: Plausibilitätsfehler bei Geburtsdatum	43
Abbildung 3-19	Konsolenausgabe	43
Abbildung 3-20	eric.log: Fehlertext und Returncode	44

Abbildung 3-21	API-Referenz (html): Korrespondierender Fehlertext und Returncode..	44
Abbildung 3-22	Konsolenausgabe	46
Abbildung 3-23	Serverantwort im XML-Format	47
Abbildung 3-24	Beispiel: ericprint.pdf	48
Abbildung 3-25	Ausschnitt aus Vordruck „2017AnlKind021.png“	50
Abbildung 3-26	Beispiel: Jahresdokumentation*.xml.....	51
Abbildung 3-27	Schematischer Aufbau der Datei ESt_2020_Datenteil.xml	52
Abbildung 3-28	Ausschnitt aus ericvorgang.cpp.....	53
Abbildung 3-29	Ausschnitt aus ericvorgang.h	53
Abbildung 3-30	Ausschnitt aus ericdemo.cpp	54
Abbildung 3-31	Konsolenausgabe: Beispiel für das Element <TransferHeader>.....	54
Abbildung 4-1	Zusammenhang zwischen Datenartversionmatrix.xml und plugins- Ordner.....	56
Abbildung 4-2	ERiC Basisbibliotheken.....	57
Abbildung 4-3	Anzahl ERiC Plugin-Bibliotheken und Basisbibliotheken der kompletten gegenüber der größenoptimierten ERiC Installation	58
Abbildung 5-1	Die Ausgaben der Fortschrittcallbacks in ericdemo	59
Abbildung 5-2	Fortschrittcallbacks am Beispiel der „callbackhandler.cpp“	60
Abbildung 5-3	Die Callbackfunktionen globalerFortschrittAdapter() und fortschrittAdapter()	61
Abbildung 5-4	Callbacks anmelden.....	61
Abbildung 5-5	Das Abmelden der beiden Callbackfunktionen im Destructor nicht vergessen!	61
Abbildung 6-1	XML-Beispiel: Elemente <Nutzdaten> und <Datenabholung>.....	63
Abbildung 6-2	Konsolenausgabe: Transferhandle und Serverantwort.....	64
Abbildung 6-3	Serverantwort mit XML-Syntax-Highlighting	65
Abbildung 6-4	Konsolenausgabe	66
Abbildung 6-5	XML-Beispiel: Element <PostfachBestaetigung>	67
Abbildung 6-6	Konsolenausgabe	68