Schnittstellen-Spezifikation zur Übertragung von Rechtstexten der IT-Recht Kanzlei München in Ihr System

Daten werden per POST im XML-Format an Ihre Schnittstelle gepusht. Das Dokument ist UTF-8 kodiert. Der POST-Parameter lautet standardmäßig “xml” (`$_POST['xml']`), andere Vereinbarungen sind möglich.

1 Implementierung und Verwendung des PHP-SDKs

Für die Nutzer unserer IT-Recht Kanzlei Schnittstelle mit PHP Backend bieten wir ein SDK an. Das SDK ist übersichtlich gestaltet und leicht zu verwenden. Zunächst müssen die zwei abstrakten Methoden der Klasse **LTIHandler** und eine der beiden Authentifizierungsmethoden überschrieben werden.

1.1 Erstellung eines eigenen LTIHandlers

1.1.1 Authentifizierungsmethoden

Damit Sie sicherstellen können, dass die Übertragung der Rechtstexte von System der IT-Recht Kanzlei stammen, können Sie eine der beiden folgenden Methoden implementieren.

isTokenValid(string $token): bool

Hier muss die Korrektheit des übermittelten Tokens geprüft werden, wenn das System mit einem Token arbeiten soll. Der Token wird von Ihnen manuell bzw. Ihrem Shopsystem automatisiert generiert und während Konfiguration der Schnittstelle durch den Nutzer der Schnittstelle im Mandantenportal der IT-Recht Kanzlei hinterlegt. Dies ist die bevorzugte Authentifikationsmethode. Um ein zufälliges Token zu erzeugen können Sie die Methode `LTI::generateToken()` verwenden.

validateUserPass(string $username, string $password): bool

Hier muss die Korrektheit von einem übermittelten Benutzernamen und Passwort geprüft werden. Den Benutzernamen und das Passwort können die Benutzer in Ihrem Plugin selbst vergeben oder es kann von Ihrem Plugin erstellt werden. Benutzername und Passwort müssen anschließend bei der Einrichtung der Schnittstelle im Mandantenportal hinterlegt werden.

1.1.2 handleActionGetAccountList(): LTIAccountListResult

Diese Funktion soll als Ergebnis eine Liste aller Verkaufskanäle (Sales Channels) Ihres Systems zurück liefern. Für jeden Verkaufskanal ist eine ID (`accountid`) und der Name (`accountname`) anzugeben. Zusätzlich kann zu jedem Verkaufskanal eine Liste der verfügbaren Zielsprachen und -länder übergeben werden.

Auch wenn es sich bei Ihrem System nicht um ein Multishop-System handelt, wird empfohlen diese Funktion zu implementieren, um die unterstützten Zielsprachen und -länder Ihres Systems bekannt zu machen. Für diesen Anwendungsfall muss nur ein Verkaufskanal angegeben werden bei dem die ID mit `0` angeben wird. Der Accountname kann ebenfalls leer bleiben.

Wichtig: Bitte achten Sie bei den Namen der Verkaufskanäle (`accountname`) darauf, dass diese 5 XML-Sonderzeichen durch entsprechende Entities ersetzt werden:

```
& => &amp;
< => &lt;
> => &gt;
" => &quot;
' => &apos;
```

1.1.3 handleActionPush(LTIPushData $data): LTIPushResult

Nachdem der Request am Plugin angekommen ist, muss das Dokument in die entsprechende Zielseite integriert werden. Die Verarbeitung des Dokuments wird hier vorgenommen. Das Objekt `LTIPushData` verfügt über Getter-Methoden für alle Eigenschaften des Dokuments, die abgefragt und anschließend weiterverarbeitet werden können.

1.2 Verwendung der LTI Klasse und des von Ihnen implementierten LTI-Handlers.

Die Klasse `LTI` (`LTI.php`) übernimmt die Vorverarbeitung der eingehen XML-Daten, wertet die auszuführende Aktion aus und führt einen Großteil der Fehlerbehandlung durch. Um die Lauffähigkeit des SDKs zu gewährleisten, sollte diese Datei nicht geändert werden.

Der Konstruktor der LTI-Klasse benötigt drei Parameter.

1. Eine Instanz Ihrer überschriebenen LTIHandler-Klasse
2. Die Version des Systems, das mit ihrer Implementation angesprochen wird
3. Die Version Ihrer Implementation, in dem das SDK verwendet wird

Als letzter Schritt muss die `handleRequest()`-Methode des Objekts der LTI-Klasse aufgerufen werden. Hierzu kann `$_POST['xml']` direkt als Parameter genutzt werden.

Sowohl Fehlerbehandlung innerhalb des SDKs, als auch das Vorbereiten der Response für den IT-Recht Kanzlei Server werden vom SDK automatisch behandelt. Der Entwickler kann diese Aspekte ignorieren.

Eine Beispielimplementierung des SDKs ist in der mitgelieferten Datei example.php enthalten.

2 Allgemeines

2.1 Versionierung

Voraussetzungen für die Verarbeitung der Versionsnummern durch das Mandantenportal der IT-Recht Kanzlei:

- `meta_modulversion`: Vergeben Sie für Ihre Implementation bei Erstersterstellung und bei jedem folgenden Update hochzählende Versionsnummern
- `meta_shopversion`: Hier wenn möglich ebenfalls eine durch Vergleichsoperatoren vergleichbare Systemversionsnummer übermitteln (z.B. “2.0” statt “XY2” - andernfalls nach Absprache). Teilen Sie uns bitte die Struktur Ihrer Versionierung mit (z.B. “major.minor”), damit wir diese intern korrekt verarbeiten können.

2.2 Multishop-Systeme

Falls es sich bei Ihrem System um ein sogenanntes Multishop-System handelt, d.h. unter eine Administrationsoberfläche existieren mehrere Verkaufskanäle/Dienste, ist es erforderlich, dass dem Benutzer Ihrer Implementation im Mandantenportal der IT-Recht Kanzlei zunächst eine Auswahlmöglichkeit angeboten wird, für welchen Verkaufskanal/Dienst die Rechtstexte übertragen werden sollen.

Die Liste der Verkaufskanäle wird mit Hilfe der Funktion `handleActionGetAccountList()` abgerufen.

Beim anschließenden “push” wird bei Multishop-Systemen die vom Benutzer ausgewählte `accountid` im XML-Element `user_account_id` übermittelt.

2.3 Best Practices

- Vergeben Sie für Ihr Modul bei Erstersterstellung und bei jedem folgenden Update ordentliche numerische, hochzählende Versionsnummern (z.B. “1.0”, “1.1”, “1.2”, ...). Neben klassischen Versionsnummern kann hier auch numerisch das Veröffentlichungsdatum genutzt werden (z.B. “20230827”, Format YYYYMMDD). Nennen Sie die aktuelle Versionsnummer immer in der dem Modul beiliegenden Dokumentation / Installationsanleitung und mindestens in der Haupt-Programmdatei.
- Fügen Sie Ihrem Modul-Download oder auf der Download-Seite eine gut verständliche Installationsanleitung bei. Berücksichtigen Sie in Ihrer Beschreibung auch Sonderfälle (z.B. für ältere Shopversionen).
- Fügen Sie neuen Versionen Ihrer Implementation (Updates) eine Beschreibung für den Nutzer bei, wie die Aktualisierung auf die neueste Version durchzuführen ist (diese weicht oft von der klassischen Installationsanleitung ab), sofern das Update nicht automatisch abläuft (z.B. durch einen Klick auf “Update” im Modul-Store des Shops).
- Fügen Sie neuen Veröffentlichungen Ihrer Implementation (Updates) eine changelog.txt o. Ä. bei, um den Nutzer über die Neuerungen und Fehlerkorrekturen zu informieren.

3 Testen

Um Ihre Implementierung der Schnittstelle testen zu können, lesen Sie bitte die README.md im Verzeichnis **testSuite**.

Als weiteres Werkzeug zum Testen können Sie das LTI Test Tool verwenden. Dort können Sie die API-URL und die Daten zur Authentifikation für ihr Testsystem hinterlegen und Requests mit der Rechtstexteschnittstelle der IT-Recht Kanzlei ausführen.

4 Details der Implementierung innerhalb des SDK

Sollten Sie das SDK nicht verwenden können oder wollen, finden Sie nachfolgend noch einige Details, die für eine eigene Implementierung hilfreich sein könnten.

4.1 XML-Elemente, [Datentyp] und (mögliche Werte):

- api_version [string]

  Versionsnummer der Schnittstelle (z.B. “1.0”)

- user_auth_token [string]

  Wird von Ihrem System generiert und ist auch dort zu entnehmen. Dient der Authentifizierung.

- action [string] (`push` | `getaccountlist` | `version`)

  “`version`” gibt lediglich die Shop-Version, Modul-Version und auf dem System installierte PHP-Version zurück.

  “`getaccountlist`” listet alle Verkaufskanäle und deren unterstützte Sprachen auf.

  “`push`” ist der Befehl für die Übertragung eines Rechtstextes.

- user_account_id [string]

  wird nur für Multishop-Systeme gesetzt, wenn zuvor durch die Action `getaccountlist` eine Liste der Verkaufskanäle abgerufen und vom Mandanten eine Auswahl getroffen wurde.

- rechtstext_type [string] (impressum | agb | datenschutz | widerruf)

  Art des übertragenen Rechtstextes

- rechtstext_title [string]

  Titel des übertragenen Rechtstextes in Originalsprache

- rechtstext_text [text]

  Text-Variante des Rechtstextes

- rechtstext_pdf [text]

  PDF -Variante des Rechtstextes

- rechtstext_html [text]

  HTML-Variante des Rechtstextes

- rechtstext_country [string]

  ISO 3166-1-alpha-2, Land, z.B. “DE” für Deutschland, wird uppercase übermittelt

- rechtstext_language [string]

  ISO 639-1, Sprache d. Rechtstextes, z.B. “de” für Deutsch, wird lowercase übermittelt

- rechtstext_language_iso639_2b [string]

  ISO 639-2 bibliographic code (B code), Sprache des Rechtstextes, z.B. “ger” für Deutsch, lowercase

4.2 Statuscodes

- **success:** Es hat alles geklappt. Sie bestätigen,
  - dass bei der Abfrage der Version keine Fehler aufgetreten sind
  - dass bei der Abfrage der Accountliste keine Fehler aufgetreten sind
  - dass bei einem Push eines Rechtstextes dieser erfolgreich im Account/Shop des Users publiziert wurde
- **error:** Unabhängig vom Fehlercode wird der Status der Fehler-Response immer “error” sein

4.3 Errorcodes

Die Fehlercodes sind in der Datei LTIError.php dokumentiert. Bitte verwenden Sie, wenn möglich, nur die in der Datei definierten Fehlercodes. Vermeiden Sie die Verwendung des Fehlercodes 99 so gut es geht. Eigene Fehlercodes können mit dem Zahlenraum >= 100 definiert werden. Bitte teilen Sie den Fehlercode und dessen Bedeutung der IT-Recht Kanzlei mit. Fehlercodes für weitere generische Fehler können nach Absprache auch dem Zahlenraum < 100 hinzugefügt werden.

4.4 Beispiel XML-Response Ausgaben

4.4.1 Bei Erfolg “version”:

```
<?xml version="1.0" ?>
<response>
    <status>success</status>
    <meta_shopversion>1.0</meta_shopversion>
    <meta_modulversion>1.1.0</meta_modulversion>
    <meta_phpversion>7.4</meta_phpversion>
</response>
```

4.4.2 Bei Erfolg “getaccountlist”:

```
<?xml version="1.0" ?>
<response>
    <status>success</status>
    <meta_shopversion>1.0</meta_shopversion>
    <meta_modulversion>1.1.0</meta_modulversion>
    <meta_phpversion>7.4</meta_phpversion>
    <account>
        <accountid>12345</accountid>
        <accountname>Shop 1</accountname>
        <locales>
            <locale>de</locale>
            <locale>en</locale>
            <locale>fr</locale>
        </locales>
        <countries>
          <country>DE</country>
          <country>AT</country>
          <country>GB</country>
          <country>FR</country>
        </countries>
    </account>
    <account>
        <accountid>23456</accountid>
        <accountname>Shop 2</accountname>
        <locales>
            <locale>de</locale>
        </locales>
        <countries>
          <country>DE</country>
        </countries>
    </account>
</response>
```

4.4.3 Bei Erfolg “push”:

```
<?xml version="1.0" ?>
<response>
    <status>success</status>
    <meta_shopversion>1.0</meta_shopversion>
    <meta_modulversion>1.1.0</meta_modulversion>
    <meta_phpversion>7.4</meta_phpversion>
    <target_url>https://example.org/imprint</target_url>
</response>
```


4.4.4 Bei einem Fehler:

```
<?xml version="1.0" ?>
<response>
    <status>error</status>
    <meta_shopversion>1.0</meta_shopversion>
    <meta_modulversion>1.1.0</meta_modulversion>
    <meta_phpversion>7.4</meta_phpversion>
    <error>12</error>
    <error_message>Something was not found!</error_message>
</response>
```

5 Kontakt

Herr Max-Lion Keller, LL.M.
IT-Recht Kanzlei, Alter Messeplatz 2, 80339 München
Tel.: +49 89 1301433-0
Fax: +49 89 1301433-60
E-Mail: m.keller@it-recht-kanzlei.de

