One Identity Manager 8.2.1

HTML5-Entwicklungshandbuch

Copyright 2022 One Identity LLC.

ALLE RECHTE VORBEHALTEN.

Diese Anleitung enthält urheberrechtlich geschützte Informationen. Die in dieser Anleitung beschriebene Software wird unter einer Softwarelizenz oder einer Geheimhaltungsvereinbarung bereitgestellt. Diese Software darf nur in Übereinstimmung mit den Bestimmungen der geltenden Vereinbarung verwendet oder kopiert werden. Kein Teil dieser Anleitung darf ohne die schriftliche Erlaubnis von One Identity LLC in irgendeiner Form oder mit irgendwelchen Mitteln, elektronisch oder mechanisch reproduziert oder übertragen werden, einschließlich Fotokopien und Aufzeichnungen für irgendeinen anderen Zweck als den persönlichen Gebrauch des Erwerbers.

Die Informationen in diesem Dokument werden in Verbindung mit One Identity Produkten bereitgestellt. Durch dieses Dokument oder im Zusammenhang mit dem Verkauf von One Identity LLC Produkten wird keine Lizenz, weder ausdrücklich oder stillschweigend, noch durch Duldung oder anderweitig, an jeglichem geistigen Eigentumsrecht eingeräumt. MIT AUSNAHME DER IN DER LIZENZVEREINBARUNG FÜR DIESES PRODUKT GENANNTEN BEDINGUNGEN ÜBERNIMMT ONE IDENTITY KEINERLEI HAFTUNG UND SCHLIESST JEGLICHE AUSDRÜCKLICHE, IMPLIZIERTE ODER GESETZLICHE GEWÄHRLEISTUNG ODER GARANTIE IN BEZUG AUF IHRE PRODUKTE AUS, EINSCHLIESSLICH, ABER NICHT BESCHRÄNKT AUF DIE IMPLIZITE GEWÄHRLEISTUNG DER ALLGEMEINEN GEBRAUCHSTAUGLICHKEIT, EIGNUNG FÜR EINEN BESTIMMTEN ZWECK ODER NICHTVERLETZUNG VON RECHTEN. IN KEINEM FALL HAFTET ONE IDENTITY FÜR JEGLICHE DIREKTE, INDIREKTE, FOLGE-, STÖRUNGS-, SPEZIELLE ODER ZUFÄLLIGE SCHÄDEN (EINSCHLIESSLICH, OHNE EINSCHRÄNKUNG, SCHÄDEN FÜR VERLUST VON GEWINNEN, GESCHÄFTSUNTERBRECHUNGEN ODER VERLUST VON INFORMATIONEN), DIE AUS DER NUTZUNG ODER UNMÖGLICHKEIT DER NUTZUNG DIESES DOKUMENTS RESULTIEREN, SELBST WENN ONE IDENTITY AUF DIE MÖGLICHKEIT SOLCHER SCHÄDEN HINGEWIESEN HAT. One Identity übernimmt keinerlei Zusicherungen oder Garantien hinsichtlich der Richtigkeit und Vollständigkeit des Inhalts dieses Dokuments und behält sich das Recht vor, Änderungen an Spezifikationen und Produktbeschreibungen jederzeit ohne vorherige Ankündigung vorzunehmen. One Identity verpflichtet sich nicht, die in diesem Dokument enthaltenen Informationen zu aktualisieren.

Wenn Sie Fragen zu Ihrer potenziellen Nutzung dieses Materials haben, wenden Sie sich bitte an:

One Identity LLC.
Attn: LEGAL Dept
4 Polaris Way
Aliso Viejo, CA 92656

Besuchen Sie unsere Website (http://www.OneIdentity.com) für regionale und internationale Büro-Adressen.

Patente

One Identity ist stolz auf seine fortschrittliche Technologie. Für dieses Produkt können Patente und anhängige Patente gelten. Für die aktuellsten Informationen über die geltenden Patente für dieses Produkt besuchen Sie bitte unsere Website unter http://www.OneIdentity.com/legal/patents.aspx.

Marken

One Identity und das One Identity Logo sind Marken und eingetragene Marken von One Identity LLC. in den USA und anderen Ländern. Für eine vollständige Liste der One Identity Marken besuchen Sie bitte unsere Website unter www.OneIdentity.com/legal. Alle anderen Marken sind Eigentum der jeweiligen Besitzer.

Legende

WARNUNG: Das Symbol WARNUNG weist auf ein potenzielles Risiko von Körperverletzungen oder Sachschäden hin, für das Sicherheitsvorkehrungen nach Industriestandard empfohlen werden. Dieses Symbol ist oft verbunden mit elektrischen Gefahren bezüglich Hardware.

VORSICHT: Das Symbol VORSICHT weist auf eine mögliche Beschädigung von Hardware oder den möglichen Verlust von Daten hin, wenn die Anweisungen nicht befolgt werden.

One Identity Manager HTML5-Entwicklungshandbuch
Aktualisiert - 27. April 2022, 00:48 Uhr
Version - 8.2.1

Inhalt

Über dieses Handbuch ... **5**

Architektur der One Identity Manager-HTML-Anwendungen ... **6**

HTML-Anwendungsentwicklung mit dem One Identity-GitHub-Repository ... **7**

Architektur des Angular-Workspace ... 7

Arbeitsumgebung für Nutzung des One Identity-GitHub-Repositorys einrichten ... 9

Zugriff auf die API ... 9

Eigene HTML-Anwendungen erstellen und bearbeiten ... 11

Bibliotheken anpassen ... 11

Plugins hinzufügen ... 11

Fehlende Übersetzungen prüfen ... 12

HTML-Anwendungen registrieren ... 13

Angular-Projekt kompilieren und bereitstellen ... 13

Debugging ... 14

API Server lokal hosten ... 15

Debugging mit Plugins ... 15

HTML-Anwendungsentwicklung mit der Visual Studio Code-Erweiterung ... **17**

One Identity Visual Studio Code-Erweiterung installieren ... 18

Arbeitsumgebung mit der Visual Studio Code-Erweiterung einrichten ... 18

Verbindung zur One Identity Manager-Datenbank herstellen ... 18

Entwicklungsordner einrichten ... 19

One Identity Visual Studio Code-Erweiterung verwenden ... 20

Kompilierungszweige ... 20

Kompilierungszweige auswählen ... 20

Kompilierungszweige erstellen ... 21

Kompilierungszweige löschen ... 21

API Client aktualisieren ... 22

HTML-Anwendungen automatisch einrichten ... 22

HTML-Anwendungen kompilieren und in Datenbank speichern ... 22

Übersetzungen prüfen ... 23

Anwendung erstellen ... 23

Über uns **27**
Kontaktieren Sie uns 27
Technische Supportressourcen 27

1

Über dieses Handbuch

Dieses Handbuch zeigt Webentwicklern, wie sie die HTML-Anwendungen des One Identity Manager im Quelltext ansehen und deren interne Funktionsweise nachvollziehen können.

Dafür stehen Ihnen folgende Möglichkeiten zur Verfügung:

- Sie können bestehende HTML-Anwendungen aus dem One Identity-GitHub-Repository als Vorlage verwenden (siehe HTML-Anwendungsentwicklung mit dem One Identity-GitHub-Repository auf Seite 7).
- Sie verwenden eine Erweiterung für Visual Studio Code (siehe HTML-Anwendungsentwicklung mit der Visual Studio Code-Erweiterung auf Seite 17).

Verfügbare Dokumentation

Die Online Version der One Identity Manager Dokumentation finden Sie im Support-Portal unter Online-Dokumentation. Videos mit zusätzlichen Informationen finden Sie unter www.YouTube.com/OneIdentity.

2

Architektur der One Identity Manager-HTML-Anwendungen

Der One Identity Manager verwaltet eine Ordnerstruktur, in der die Quelldateien für alle HTML-Anwendungen liegen. Diese Ordnerstruktur ist im Datenmodell als Teil der automatischen Software-Aktualisierung abgelegt und dort der Maschinenrolle **HTML Development** zugewiesen.

Sie können diese Ordnerstruktur lokal in einem Entwicklungsordner ablegen. Dort liegen auf oberster Ebene die Ordner für Datenbankmodule. Darunter liegt für jede HTML-Anwendung ein eigener Ordner. Wie Sie einen solchen Entwicklungsordner einrichten, erfahren Sie im Kapitel Entwicklungsordner einrichten auf Seite 19.

Die HTML-Anwendungen sind als nodeJS-Anwendungen aufgebaut, die das Framework **Angular** einsetzen. Grundsätzlich werden alle HTML-Anwendungen unterstützt, die sich als nodeJS-Anwendungen kompilieren lassen.

Die HTML-Anwendungen nutzen für die Kommunikation mit der One Identity Manager-API den API-Client. Der API-Client ist eine npm-Bibliothek, die automatisch bei der Kompilierung der API erzeugt und in der Datenbank abgelegt wird. Der API-Client regelt den kompletten Netzwerkzugriff auf den API Server.

Ein Großteil der Logik der HTML-Anwendungen ist in Form von Plugins realisiert, die unabhängig von einer konkreten HTML-Anwendung genutzt werden können. Damit können Sie diese Logik auch für Ihre eigenen HTML-Anwendungen nutzen.

3

HTML-Anwendungsentwicklung mit dem One Identity-GitHub-Repository

Sie können eigene HTML-Anwendungen entwickeln, indem Sie die Quelltexte der Standard-HTML-Anwendungen als Vorlage verwenden.

Die Quelltexte der Standard-HTML-Anwendungen stehen Ihnen unter folgender URL in einem GitHub-Repository zur Verfügung:
https://github.com/OneIdentity/IdentityManager.Imx

Architektur des Angular-Workspace

Das One Identity-GitHub-Repository enthält den Quellcode für die in One Identity Manager enthaltenen HTML-Anwendungen.

Es ist ein Monorepo, das den Angular-Workspace enthält, der aus Anwendungen und Bibliotheken besteht.

Jede Angular-Bibliothek und -Anwendung gehört zu einem Ordner im Verzeichnis `projects`. Der Angular-Workspace wird in der Datei `angular.json` definiert.

Tabelle 1: Angular-Bibliotheken

Name	Typ	Abhängigkeiten innerhalb des Workspace
qbm	Angular-Bibliothek	keine
qer	Angular-Bibliothek	qer
tsb	Angular-Plugin-Bibliothek	qbm, qer

Name	Typ	Abhängigkeiten innerhalb des Workspace
att	Angular-Plugin-Bibliothek	qbm, qer
rms	Angular-Plugin-Bibliothek	qbm, qer
aad	Angular-Plugin-Bibliothek	qbm, qer, tsb
aob	Angular-Plugin-Bibliothek	qbm, qer
uci	Angular-Plugin-Bibliothek	qbm, qer
cpl	Angular-Plugin-Bibliothek	qbm, qer
dpr	Angular-Plugin-Bibliothek	qbm
o3t	Angular-Plugin-Bibliothek	qbm, qer, tsb
pol	Angular-Plugin-Bibliothek	qbm, qer

Jede Angular-Bibliothek gehört zum gleichnamigen One Identity Manager-Modul.

Eine Angular-Bibliothek verhält sich wie eine reguläre Kompilierzeitabhängigkeit. Eine Plugin-Bibliothek wird zur Laufzeit dynamisch geladen. Das ist in den `imx-plugin-config.json`-Dateien der Plugins festgelegt.

Tabelle 2: Angular-Anwendungen

Name	Beschreibung	Projekttyp	Statische Abhängigkeiten
qbm-app-landingpage	API Server-Landing-Page und Server-Verwaltung	Angular-Anwendung	qbm
qer-app-portal	Web Portal	Angular-Anwendung	qbm, qer
qer-app-operationssupport	Web Portal für Betriebsunterstützung	Angular-Anwendung	qbm, qer
qer-app-pwdportal	Kennwortrücksetzungsportal	Angular-Anwendung	qbm, qer
arc-app-certaccess	CertAccessWeb Portal	Angular-App	verschiedene

Arbeitsumgebung für Nutzung des One Identity-GitHub-Repositorys einrichten

In diesem Kapitel erfahren Sie, wie Sie Ihre Arbeitsumgebung für die Nutzung des One Identity-GitHub-Repositorys einrichten, sodass Sie anschließend eigene Webanwendungen entwickeln können.

Voraussetzungen:

- Sie besitzen ein gültiges GitHub-Konto (siehe https://github.com/).

Um Ihre Arbeitsumgebung einzurichten

1. Beantragen Sie Zugriff auf das One Identity-GitHub-Repository. Das One Identity-GitHub-Repository steht Ihnen unter folgender URL zur Verfügung: https://github.com/OneIdentity/IdentityManager.Imx
2. Erstellen Sie einen Fork des One Identity-GitHub-Repositorys (siehe https://docs.github.com/en/get-started/quickstart/fork-a-repo).
3. Installieren Sie npm (siehe https://docs.npmjs.com/downloading-and-installing-node-js-and-npm).
4. Führen Sie folgenden Befehl in einem Kommandozeilenprogramm aus:

   ```
   npm install -g @angular/cli
   ```

Zugriff auf die API

Für den Zugriff auf die API wird der Typescript-API-Client verwendet.

Für jedes One Identity Manager-Modul, das API-Dienste bereitstellt, existiert eine Angular-Bibliothek `imx-api-<Modulname>.tgz` als NPM-Paket im Ordner `imx-modules` des Repositorys.

HINWEIS: Die API-Client-Bibliotheken sind nicht von Angular abhängig und können somit aus allen Javascript-Programmen verwendet werden.

Die TypeScript-API-Clients bestehen aus den folgenden Teilen:

- **Endpoint-basierte Methoden**:

  Die Klasse **V2Client** des API-Clients enthält für jeden API-Endpoint eine Methode. Der Name der Methode wird nach dem folgenden Schema generiert:

  ```
  <URL-Pfad des API-Endpoints>_<HTTP-Methode>
  ```

Beispiel

Die Methode **GET portal/serviceitems** wird zur Typescript-Methode **portal_serviceitems_get**.

- **Entity-basierte Methoden**:

  Die Klasse **TypedClient** des API-Clients enthält für jede entity-basierte API-Methode eine Wrapper-Klasse, die es ermöglicht, Entities zu laden und zu speichern.

  Beispiel für Methoden für interaktive Entities

  Für die Methode **portal/serviceitems/interactive** gibt es die Eigenschaft **TypedClient.PortalServiceItemsInteractive_byid** vom Typ der Wrapper-Klasse **PortalServiceItemsInteractive_byidWrapper**.

  Je nach Umfang der unterstützten Operationen einer API-Methode gibt es die folgenden Methoden an der Wrapper-Klasse:

  - Die Methode **createEntity** wird zum Erstellen einer neuen Entity verwendet.
  - Die Methode **Get_byid** wird zum Laden einer interaktiven Entity vom API Server verwendet. Der API Server unterstützt pro Anfrage nur das Laden eines einzelnen Objekts aus der Datenbank als interaktive Entity. Dabei müssen die Primärschlüsselwerte des Objekts als Methodenparameter angegeben werden.
  - Die Methoden **Put** und **Post** werden zum Speichern von Entities mit der PUT-Operation verwendet. Diese Methoden dürfen nicht direkt aufgerufen werden, sondern werden über die Methode **commit()** des Interfaces IEntity angesteuert.

Kapselung als Service

Der Zugriff auf die API ist pro Angular-Bibliothek in einem eigenen Angular-Service gekapselt, der sich in eigene Klassen importieren lässt:

- In der Angular-Bibliothek **qbm** heißt der Service **imx_SessionService**.
- In der Angular-Bibliothek **qer** heißt der Service **QerApiService**.
- In allen anderen Angular-Bibliotheken heißt der Service **ApiService**.

Eigene HTML-Anwendungen erstellen und bearbeiten

Um eigene HTML-Anwendungen zu erstellen und zu bearbeiten, können Sie Angular-Bibliotheken ändern und Plugins zum API Server hinzufügen.

Detaillierte Informationen zum Thema

- Bibliotheken anpassen auf Seite 11
- Plugins hinzufügen auf Seite 11

Bibliotheken anpassen

Wenn Sie den Code einer Angular-Bibliothek ändern, müssen Sie eigene Versionen aller Angular-Anwendungen, die die geänderte Angular-Bibliothek verwenden sollen, erstellen und bereitstellen.

> Wenn Sie beispielsweise die Angular-Bibliothek **qer** ändern, müssen Sie auch die Angular-Anwendungen **qer-app-portal**, **qer-app-operationssupport** und **qer-app-pwdportal** kompilieren, da all diese Anwendungen die Angular-Bibliothek **qer** enthalten.

Wenn Sie den Code einer Angular-Plugin-Bibliothek ändern, müssen Sie eine eigene Version der Angular-Plugin-Bibliothek selbst und aller von ihr abhängigen Angular-Plugin-Bibliothek erstellen und bereitstellen.

> Wenn Sie beispielsweise die Angular-Plugin-Bibliothek **tsb** ändern, müssen Sie auch die Angular-Plugin-Bibliotheken **aad** und **o3t** kompilieren, da diese Angular-Plugin-Bibliotheken die Angular-Plugin-Bibliothek **tsb** enthalten.

Plugins hinzufügen

Plugins sind Angular-Bibliotheken, die zur Laufzeit dynamisch geladen werden. Die Plugins werden vom API Server verwaltet. Plugins werden automatisch vom API Server erkannt, indem im Programmverzeichnis nach Dateien mit dem Namen `imx-plugin-config.json` gesucht wird.

Die folgende Beispieldatei legt fest, dass die Angular-Plugin-Bibliothek `ccc` in die **qer-app-portal**-Anwendung geladen werden soll. Der Name des Angular-Moduls, das instanziiert werden soll, ist **CustomConfigModule**.

```
{
   "qer-app-portal":[
      {
         "Container":"ccc",
         "Name":"CustomConfigModule"
      }
   ]
}
```

Um ein Plugin hinzuzufügen

1. Erstellen Sie auf dem API Server die Datei `imxweb\<Name der Angular-Plugin-Bibliothek>\imx-plugin-config.json` mit folgendem Inhalt:

   ```
   {
      "<Name der HTML-Anwendung>":[
         {
            "Container":"<Name der Angular-Plugin-Bibliothek>",
            "Name":"<Name des Angular-Moduls>"
         }
      ]
   }
   ```

2. Importieren Sie die Datei mithilfe des Software Loaders in Ihre One Identity Manager-Datenbank und weisen Sie sie der Maschinenrolle **API Server** zu. Weitere Informationen zum Importieren von Dateien mit dem Software Loader finden Sie im *One Identity Manager Administrationshandbuch für betriebsunterstützende Aufgaben*.
3. (Optional) Um zu prüfen, ob die HTML-Anwendung das Plugin richtig lädt, rufen Sie die URL **<URL des API Servers>/imx/applications** auf und prüfen Sie, dass an der HTML-Anwendung das entsprechende Plugin in der Liste erscheint.

Fehlende Übersetzungen prüfen

Sie können HTML-Anwendungen mithilfe des ImxClient-Kommandozeilenprogramms auf fehlende Übersetzungen überprüfen. Weitere Informationen zum ImxClient-Kommandozeilenprogramm finden Sie im *One Identity Manager API-Entwicklungshandbuch*.

Um eine HTML-Anwendung auf fehlende Übersetzungen zu überprüfen

1. Starten Sie das ImxClient-Kommandozeilenprogramm.
2. Führen Sie im Ordner, den Sie auf fehlende Übersetzungen prüfen möchten, den Befehl **check-translations** aus.

   Ein Bericht wird erstellt. Dieser Bericht zeigt Ihnen alle Dateien, in denen Texte gefunden wurden, die noch nicht oder nur teilweise übersetzt wurden.
3. (Optional) Um die Übersetzungsschlüssel und Übersetzungen anzulegen, verwenden Sie das Programm Designer. Weitere Informationen zu Übersetzungen finden Sie im *One Identity Manager Konfigurationshandbuch*.

HTML-Anwendungen registrieren

Um neue HTML-Anwendungen für die Nutzung zur Verfügung zu stellen und somit auf der Startseite des API Servers anzuzeigen, müssen Sie die HTML-Anwendungen in der Datenbank anlegen.

Um eine HTML-Anwendung in der Datenbank anzulegen

1. Starten Sie das Programm Designer.
2. Verbinden Sie sich mit der entsprechenden Datenbank.
3. Klicken Sie in der Navigation die Kategorie **Basisdaten** > **Sicherheitseinstellungen** > **HTML-Anwendungen**.
4. In der Menüleiste klicken Sie  (**Ein neues Objekt erstellen**).
5. In der Liste klicken Sie den neuen Eintrag.
6. Im Bereich **Eigenschaften** geben Sie in den entsprechenden Feldern die Daten der HTML-Anwendung an. Geben Sie mindestens die folgenden Informationen an:
   - **Anzeigename**: Geben Sie einen Namen für die HTML-Anwendung ein.
   - **HTML-Anwendung**: Geben Sie den Pfad `CCC/<Name Ihrer HTML-Anwendung>` ein.
   - **Vorkompiliert**: Setzen Sie den Wert auf **True**.

Angular-Projekt kompilieren und bereitstellen

Um ein Angular-Projekt über den API Server zur Verfügung zu stellen, müssen Sie das Angular-Projekt kompilieren und das Paket als ZIP-Datei verfügbar machen.

Um ein Angular-Projekt zu kompilieren und bereitzustellen

1. Starten Sie ein Kommandozeilenprogramm.
2. Wechseln Sie in das Verzeichnis des Angular-Workspace.
3. Führen Sie den folgenden Befehl aus:

   ```
   ng build <Projektname>
   ```

4. Komprimieren Sie den Inhalt des Verzeichnisses mit dem Kompilat (üblicherweise `dist/<Projektname>`) als ZIP-Datei mit dem Namen `Html_<Projektname>.zip`.
5. Kopieren Sie die ZIP-Datei in den Unterordner `imxweb\custom` Ihrer Arbeitsumgebung.
6. Importieren Sie die ZIP-Datei mithilfe des Software Loaders in Ihre One Identity Manager-Datenbank und weisen Sie sie der Maschinenrolle **API Server** zu. Weitere Informationen zum Importieren von Dateien mit dem Software Loader finden Sie im *One Identity Manager Administrationshandbuch für betriebsunterstützende Aufgaben*.

Debugging

Das Ausführen und Debuggen von HTML-Anwendungen ist mit den Standardwerkzeugen der Angular-CLI-Toolchain möglich.

> Sie können beispielsweise den Befehl `ng serve qer-app-portal` verwenden, um die Web Portal-HTML-Anwendung zu debuggen.

Um eine HTML-Anwendung zu debuggen

1. Hosten Sie den API Server lokal (siehe API Server lokal hosten auf Seite 15).
2. Starten Sie ein Kommandozeilenprogramm.
3. Wechseln Sie in das Verzeichnis des Angular-Workspace.
4. Führen Sie den folgenden Befehl aus:

   ```
   npm run start <Name der HTML-Anwendung>
   ```

   Ein Webserver, der standardmäßig unter der URL `http://localhost:4200` erreichbar ist und die HTML-Anwendung hostet, wird gestartet.
5. Starten Sie das Debugging in einer entsprechenden Entwicklungsumgebung (beispielsweise Visual Studio Code).

API Server lokal hosten

Um HTML-Anwendungen zu debuggen und zu entwickeln, benötigen Sie eine API Server-Instanz, mit der sich die HTML-Anwendungen verbinden. Für diese Zwecke können Sie einen API Server lokal hosten.

HINWEIS: Die HTML-Anwendungen verbinden sich mit dem API Server über die URL, die in der Datei `environment.ts` der HTML-Anwendungen definiert ist. Die Standard-URL, unter der ein lokal gehosteter API Server ausgeführt wird, ist `http://localhost:8182`.

Um einen API Server lokal zu hosten

1. Starten Sie ein Kommandozeilenprogramm.
2. Wechseln Sie in das Verzeichnis des Angular-Workspace.
3. Führen Sie den folgenden Befehl aus:

```
imxclient.exe run-apiserver -B
```

Debugging mit Plugins

Sie können das Debugging auch mit Plugins durchführen. Das Debugging mit Plugins funktioniert nur, wenn der lokale API Server das Plugin auch finden kann.

Um eine statische Angular-Bibliothek zu debuggen

1. Hosten Sie den API Server lokal (siehe API Server lokal hosten auf Seite 15).
2. Starten Sie ein Kommandozeilenprogramm.
3. Wechseln Sie in das Verzeichnis des Angular-Workspace.
4. Führen Sie den folgenden Befehl aus:

```
npm run build:watch <Name der Angular-Bibliothek>
```

5. Starten Sie ein weiteres Kommandozeilenprogramm.
6. Wechseln Sie in das Verzeichnis des Angular-Workspace.
7. Führen Sie den folgenden Befehl aus:

```
npm run start <Name der HTML-Anwendung>
```

   Ein Webserver, der standardmäßig unter der URL `http://localhost:4200` erreichbar ist und die HTML-Anwendung hostet, wird gestartet.
8. Starten Sie das Debugging in einer entsprechenden Entwicklungsumgebung (beispielsweise Visual Studio Code).

Um eine Angular-Plugin-Bibliothek zu debuggen

1. Hosten Sie den API Server lokal (siehe API Server lokal hosten auf Seite 15).
2. Starten Sie ein Kommandozeilenprogramm.
3. Wechseln Sie in das Verzeichnis des Angular-Workspace.
4. Führen Sie den folgenden Befehl aus:

   ```
   npm run build:watch:dynamic <Angular-Plugin-Bibliothek>
   ```

5. Starten Sie ein weiteres Kommandozeilenprogramm.
6. Wechseln Sie in das Verzeichnis des Angular-Workspace.
7. Führen Sie den folgenden Befehl aus:

   ```
   npm run start <Name der HTML-Anwendung>
   ```

   Ein Webserver, der standardmäßig unter der URL `http://localhost:4200` erreichbar ist und die HTML-Anwendung hostet, wird gestartet.
8. Starten Sie das Debugging in einer entsprechenden Entwicklungsumgebung (beispielsweise Visual Studio Code).

4

HTML-Anwendungsentwicklung mit der Visual Studio Code-Erweiterung

In diesem Kapitel finden Sie Informationen über den Umgang mit der One Identity Visual Studio Code-Erweiterung.

Mit der Visual Studio Code-Erweiterung können Sie die Kommandos des ImxClient-Kommandozeilenprogramms ohne tiefere Kenntnis der Syntax direkt in Visual Studio Code aufrufen:

- Verbinden zur Datenbank
- Einrichten des Entwicklungsordners
- Verwalten und verwenden von Kompilierungszweigen
- Aktualisieren der HTML-Anwendungen mit gleichzeitiger Einrichtung
- Kompilieren einer HTML-Anwendung
- Aktualisieren der Basisbibliotheken (API-Client)
- Prüfen auf fehlende Übersetzungen

Die Visual Studio Code-Erweiterung liefert Ihnen zudem folgende hilfreiche Informationen:

- Status der aktuellen Datenbankverbindung
- Informationen über den verwendeten ImxClient
- Informationen über den verwendeten API-Client
- Zustand des Entwicklungsordners
- Anzeige des Kompilierungszweiges
- Informationen zur letzten Übersetzungsprüfung

Verwandte Themen

- One Identity Visual Studio Code-Erweiterung installieren auf Seite 18
- Arbeitsumgebung mit der Visual Studio Code-Erweiterung einrichten auf Seite 18
- One Identity Visual Studio Code-Erweiterung verwenden auf Seite 20
- Anwendung erstellen auf Seite 23

One Identity Visual Studio Code-Erweiterung installieren

Um die One Identity Visual Studio Code-Erweiterung zu installieren

1. Starten Sie das Programm Visual Studio Code.
2. In Visual Studio Code klicken Sie in der linken Navigation auf **Erweiterungen**.
3. Im Bereich **Erweiterungen** klicken Sie auf **...** | **Aus VSIX installieren**.
4. Im Datei-Browser wählen Sie im One Identity Manager-Installationsverzeichnis die Datei `vcode-extension.vsix` und klicken Sie **Installieren**.
5. Starten Sie Visual Studio Code neu.
6. In Visual Studio Code klicken Sie in der linken Navigation auf **Explorer**.
7. Im Bereich **Explorer** navigieren Sie zu **One Identity** | **Configuration**.
8. Neben **Imx client** klicken Sie **One Identity Manager: edit imx client path**.
9. Im Datei-Browser wählen Sie im One Identity Manager-Installationsverzeichnis die Datei `ImxClient.exe` und klicken Sie **Öffnen**.

Arbeitsumgebung mit der Visual Studio Code-Erweiterung einrichten

In diesem Kapitel erfahren Sie, wie Sie mithilfe der One Identity Visual Studio Code-Erweiterung Ihre Arbeitsumgebung einrichten, sodass Sie anschließend problemlos die Visual Studio Code-Erweiterung und ihre Funktionen verwenden können.

Verwandte Themen

- Verbindung zur One Identity Manager-Datenbank herstellen auf Seite 18
- Entwicklungsordner einrichten auf Seite 19

Verbindung zur One Identity Manager-Datenbank herstellen

Viele der Funktionen, die Ihnen die Visual Studio Code-Erweiterung bietet, benötigen eine Verbindung zur One Identity Manager-Datenbank.

Um eine Verbindung zur One Identity Manager-Datenbank herzustellen

1. In Visual Studio Code klicken Sie in der linken Navigation auf **Explorer**.
2. Im Bereich **Explorer** navigieren Sie zu **One Identity** | **Workspace**.
3. Neben **Database** klicken Sie **One Identity Manager: connect to database**.

   Das Dialogfenster zum Auswählen der Datenbankverbindung öffnet sich.
4. Im Dialogfenster nehmen Sie eine der folgenden Aktionen vor:
   - Um eine bestehende Verbindung zur One Identity Manager-Datenbank zu verwenden, wählen Sie aus der Auswahlliste **Datenbankverbindung auswählen** die entsprechende Verbindung aus.

     - ODER -
   - Um eine neue Verbindung zur One Identity Manager-Datenbank zu verwenden, klicken Sie **Neue Verbindung erstellen** und geben Sie eine neue Verbindung an.
5. Unter **Authentifizierungsverfahren** geben Sie das Verfahren und die Anmeldedaten an, mit denen Sie sich an der Datenbank anmelden möchten.
6. Klicken Sie **Anmelden**.

Entwicklungsordner einrichten

Wenn Sie einen Entwicklungsordner einrichten werden die folgenden Aktionen durchgeführt:

- Die Ordner für HTML-Anwendungen werden aus der Datenbank heruntergeladen und als Unterordner des Entwicklungsordners gespeichert.
- Die Bibliotheken werden aus der Datenbank heruntergeladen und im Unterordner `imx-modules` des Entwicklungsordners gespeichert.
- Die neuesten Bibliotheken werden in den Unterordner `imx-modules` des Entwicklungsordners heruntergeladen. Wenn keine Datenbankverbindung besteht, wird dieser Schritt übersprungen (siehe auch Verbindung zur One Identity Manager-Datenbank herstellen auf Seite 18).

Um einen One Identity Manager-Entwicklungsordner einzurichten

1. Erstellen Sie auf Ihrer Festplatte einen Ordner, den Sie als Entwicklungsordner verwenden möchten.

   HINWEIS: Der Pfad zu diesem Ordner sowie der Ordner selbst dürfen nur UTF-8-Zeichen enthalten.
2. In Visual Studio Code klicken Sie in der Menüleiste **Datei** | **Ordner öffnen**.
3. Im Datei-Browser wählen Sie den zuvor erstellten Ordner, den Sie als Entwicklungsordner verwenden möchten.
4. In Visual Studio Code klicken Sie in der linken Navigation auf **Explorer**.

5. Im Bereich **Explorer** navigieren Sie zu **One Identity** | **Workspace**.
6. Neben **Development folder** klicken Sie **One Identity Manager: set up development folder**.
7. Bestätigen Sie die Abfrage mit **Continue setup**.

One Identity Visual Studio Code-Erweiterung verwenden

In diesem Kapitel erfahren Sie, wie Sie die One Identity Visual Studio Code-Erweiterung verwenden.

Verwandte Themen

- Kompilierungszweige auf Seite 20
- API Client aktualisieren auf Seite 22
- HTML-Anwendungen automatisch einrichten auf Seite 22
- HTML-Anwendungen kompilieren und in Datenbank speichern auf Seite 22
- Übersetzungen prüfen auf Seite 23

Kompilierungszweige

Um verschiedene Versionen Ihres kompilierten Projektes zu verwalten und in der Datenbank zu hinterlegen, verwenden Sie Kompilierungszweige. Sie können den zu verwendenden Kompilierungszweig auswählen, die vorhandenen Kompilierungszweige anzeigen, Kompilierungszweige erstellen und Kompilierungszweige löschen.

Verwandte Themen

- Kompilierungszweige auswählen auf Seite 20
- Kompilierungszweige erstellen auf Seite 21
- Kompilierungszweige löschen auf Seite 21

Kompilierungszweige auswählen

Um einen Kompilierungszweig auszuwählen und zu verwenden

1. In Visual Studio Code klicken Sie in der linken Navigation auf **Explorer**.
2. Im Bereich **Explorer** navigieren Sie zu **One Identity** | **Configuration**.

3. Neben **Branch ID**, klicken Sie **One Identity Manager: edit branch id**.

   TIPP: Falls noch keine Verbindung zur One Identity Manager-Datenbank besteht, müssen Sie diese jetzt herstellen (siehe Verbindung zur One Identity Manager-Datenbank herstellen auf Seite 18).

   Eine Auswahlliste mit den verfügbaren Kompilierungszweigen öffnet sich.
4. In der Auswahlliste klicken Sie den Kompilierungszweig, den Sie verwenden möchten.

Kompilierungszweige erstellen

Um einen Kompilierungszweig zu erstellen

1. In Visual Studio Code klicken Sie in der linken Navigation auf **Explorer**.
2. Im Bereich **Explorer** navigieren Sie zu **One Identity** | **Configuration**.
3. Neben **Branch ID**, klicken Sie **Add branch**.

   Ein Eingabefeld öffnet sich.
4. Im Eingabefeld geben Sie einen Namen für den Kompilierungszweig ein und drücken Sie **Enter**.

   TIPP: Falls noch keine Verbindung zur One Identity Manager-Datenbank besteht, müssen Sie diese jetzt herstellen (siehe Verbindung zur One Identity Manager-Datenbank herstellen auf Seite 18).

Kompilierungszweige löschen

Um einen Kompilierungszweig zu löschen

1. In Visual Studio Code klicken Sie in der linken Navigation auf **Explorer**.
2. Im Bereich **Explorer** navigieren Sie zu **One Identity** | **Configuration**.
3. Neben **Branch ID**, klicken Sie **One Identity Manager: edit branch id**.

   TIPP: Falls noch keine Verbindung zur One Identity Manager-Datenbank besteht, müssen Sie diese jetzt herstellen (siehe Verbindung zur One Identity Manager-Datenbank herstellen auf Seite 18).

   Eine Auswahlliste mit den verfügbaren Kompilierungszweigen öffnet sich.
4. In der Auswahlliste klicken Sie den Kompilierungszweig, den Sie löschen möchten.
5. Im Bereich **Explorer** neben **Branch ID**, klicken Sie **Delete branch**.
6. Bestätigen Sie die Abfrage mit **OK**.

API Client aktualisieren

Sie können die Basisbibliotheken und damit den API Client jederzeit aktualisieren.

Um den API-Client zu aktualisieren

1. In Visual Studio Code klicken Sie in der linken Navigation auf **Explorer**.
2. Im Bereich **Explorer** navigieren Sie zu **One Identity** | **Workspace**.
3. Neben **Api client**, klicken Sie **One Identity Manager: update api client**.

   TIPP: Falls noch keine Verbindung zur One Identity Manager-Datenbank besteht, müssen Sie diese jetzt herstellen (siehe Verbindung zur One Identity Manager-Datenbank herstellen auf Seite 18).

HTML-Anwendungen automatisch einrichten

Um HTML-Anwendungen automatisch zu erkennen und einzurichten, führen Sie eine Überprüfung der vorhandenen Ordner in Ihrem Entwicklungsordner durch. Dabei werden folgende Schritte durchgeführt:

- Der Unterordner `assets` des Arbeitsverzeichnisses wird mit dem Unterordner `src/assets` der Anwendung verknüpft.
- Die entsprechenden Plugins werden verknüpft und integriert.

HINWEIS: HTML-Anwendungen werden ebenso vor jeder Kompilierung automatisch eingerichtet.

Um HTML-Anwendungen automatisch einzurichten

1. In Visual Studio Code klicken Sie in der linken Navigation auf **Explorer**.
2. Im Bereich **Explorer** navigieren Sie zu **One Identity** | **Workspace** | **Development folder**.
3. Neben **Apps** klicken Sie **reload existing apps**.

   TIPP: Falls noch keine Verbindung zur One Identity Manager-Datenbank besteht, müssen Sie diese jetzt herstellen (siehe Verbindung zur One Identity Manager-Datenbank herstellen auf Seite 18).

HTML-Anwendungen kompilieren und in Datenbank speichern

Sie können Ihre HTML-Anwendungen kompilieren und das Ergebnis automatisch in der One Identity Manager-Datenbank speichern.

Um die HTML-Anwendungen zu kompilieren und in der Datenbank zu speichern

1. In Visual Studio Code klicken Sie in der linken Navigation auf **Explorer**.
2. Im Bereich **Explorer** navigieren Sie zu **One Identity** | **Workspace** | **Development folder** | **Apps**.
3. Neben der HTML-Anwendung, die Sie kompilieren und in der Datenbank speichern möchten, klicken Sie **Compile app**.

   TIPP: Falls noch keine Verbindung zur One Identity Manager-Datenbank besteht, müssen Sie diese jetzt herstellen (siehe Verbindung zur One Identity Manager-Datenbank herstellen auf Seite 18).

   Die Debug-Kompilierung wird durchgeführt und die Änderungen werden im aktiven Kompilierungszweig gespeichert.

Übersetzungen prüfen

Sie können den Entwicklungsordner auf fehlende Übersetzungen in Ihrer HTML-Anwendung überprüfen.

Um den Entwicklungsordner auf fehlende Übersetzungen zu überprüfen

1. In Visual Studio Code klicken Sie in der linken Navigation auf **Explorer**.
2. Im Bereich **Explorer** navigieren Sie zu **One Identity** | **Workspace** | **Development folder**.
3. Neben **Translations** klicken Sie **Check translations**.

   Ein Bericht öffnet sich. Dieser Bericht zeigt Ihnen alle Dateien, in denen Texte gefunden wurden, die noch nicht oder nur teilweise übersetzt wurden.

Anwendung erstellen

Sie können mithilfe der One Identity Visual Studio Code-Erweiterung Ihre eigene HTML-Anwendung erstellen und in die bestehende Umgebung einbinden.

Das Erstellen einer neuen HTML-Anwendung führen Sie in drei Schritten durch:

1. Erstellen Sie die neue HTML-Anwendung.
2. Importieren Sie die Quelldateien in die Datenbank.
3. Legen Sie die HTML-Anwendung in der Datenbank an.

Um eine neue HTML-Anwendung zu erstellen

1. Installieren Sie die One Identity Visual Studio Code-Erweiterung (siehe One Identity Visual Studio Code-Erweiterung installieren auf Seite 18).

2. Richten Sie mithilfe der One Identity Visual Studio Code-Erweiterung die Arbeitsumgebung ein (siehe Arbeitsumgebung mit der Visual Studio Code-Erweiterung einrichten auf Seite 18).
3. In Ihrer Arbeitsumgebung legen Sie einen neuen Ordner `CCC` an.
4. Stellen Sie sicher, dass Sie die korrekte Angular-Version verwenden (9.0):
    a. Öffnen Sie ein Kommandozeilenprogramm.
    b. Im Ordner `CCC` führen Sie den Befehl **ng version** aus.
    c. Überprüfen Sie die ausgegebene Versionsnummer.

    > TIPP: Wenn keine Versionsnummer ausgegeben wird, dann ist Angular weder lokal noch global installiert.
    >
    > Um Angular global zu installieren, führen Sie den Befehl **npm install -g @angular/cli** aus.
    >
    > Um Angular lokal zu installieren, führen Sie den Befehl **npm install @angular/cli** aus.
5. Im Ordner `CCC` führen Sie den Befehl **ng new <Name Ihrer HTML-Anwendung>** aus.
6. Im Ordner `QBM/OpsWeb` kopieren Sie die Datei `imx-plugin-config.json` und fügen Sie sie in den Ordner `CCC/<Name Ihrer HTML-Anwendung>` ein.
7. Im Ordner `CCC/<Name Ihrer HTML-Anwendung>` bearbeiten Sie die (gerade kopierte) Datei `imx-plugin-config.json`:
    - Entfernen Sie alle Kategorien außer **common** und **shared**.
8. Im Ordner `CCC/<Name Ihrer HTML-Anwendung>` bearbeiten Sie die Datei `package.json`:
    a. Unter `scripts` fügen Sie `"build:debug": "ng build --prod --source-map"` hinzu.
    b. Unter `dependencies` fügen Sie die Abhängigkeiten zu `imx-api`, `imx-qbm-components` und `imx-qbm-dbts` hinzu.
    c. Unter `dependencies | rxjs` ändern Sie die Version auf `^6.3.3`.
9. Im Ordner `CCC/<Name Ihrer HTML-Anwendung>` bearbeiten Sie die Datei `angular.json`:
    - Im Bereich `projects | <Name Ihrer HTML-Anwendung> | architect | build | options` fügen Sie `"outputPath": "dist"` hinzu.
    - Im Bereich `projects | <Name Ihrer HTML-Anwendung> | architect | build | options` fügen Sie `"preserveSymlinks": true` hinzu.
10. Im Ordner `CCC/<Name Ihrer HTML-Anwendung>/src` bearbeiten Sie die Datei `index.html`:
    - Im Bereich `head` fügen Sie `<base href=".">` hinzu.
11. Im Ordner `CCC/<Name Ihrer HTML-Anwendung>/src/environments` bearbeiten Sie die Dateien `environment.<Name Ihrer Domäne>.ts` und `environment.ts`:
    - Fügen Sie jeweils einen Eintrag für die Client-URL hinzu:

      Für `environment.<Name Ihrer Domäne>.ts`: `clientUrl: ''`

      Für `environment.ts`: `clientUrl: 'http://localhost:8182'`

12. Im Ordner `CCC/<Name Ihrer HTML-Anwendung>` bearbeiten Sie die Datei `tsconfig.json`:
    - Im Bereich `compilerOptions | paths` fügen Sie `"@shared/*": [ "src/imx-plugins/QBM/shared/*"]` hinzu.
13. Stellen Sie mithilfe der One Identity Visual Studio Code-Erweiterung sicher, dass Sie keinen Kompilierungszweig verwenden (siehe Kompilierungszweige auswählen auf Seite 20).
14. Kompilieren Sie die HTML-Anwendung mithilfe der One Identity Visual Studio Code-Erweiterung (siehe HTML-Anwendungen kompilieren und in Datenbank speichern auf Seite 22).
15. Rufen Sie die fertig kompilierte HTML-Anwendung über die URL: **<URL des API Servers>/html/<Name Ihrer HTML-Anwendung>/** auf.

    HINWEIS: Verwenden Sie hier den Namen der HTML-Anwendung, so wie er in der Datei `package.json` hinterlegt ist.

Um die Quelldateien in die Datenbank zu importieren

1. Wechseln Sie in Ihre Arbeitsumgebung.
2. Starten Sie das Programm Software Loader.
3. Klicken Sie auf **In Datenbank importieren**.
4. Klicken Sie **Weiter**.
5. Auf der Seite **Verbindung zur Datenbank herstellen** wählen Sie die gewünschte Datenbank aus und geben Sie die Benutzerdaten an.
6. Klicken Sie **Weiter**.
7. Wählen Sie Ihre Arbeitsumgebung aus.
8. Markieren Sie alle Dateien unterhalb des Ordners `CCC/<Name Ihrer HTML-Anwendung>`. Schließen Sie dabei die Dateien der folgenden Ordner aus:
    - `node_modules`
    - `dist`
    - `.git`
    - `src/assets`
    - `src/imx-modules`
9. Klicken Sie **Weiter**.
10. Bestätigen Sie die Abfrage mit **Ja**.
11. Auf der Seite **Maschinenrollen zuordnen** markieren Sie alle Dateien und aktivieren Sie die Maschinenrolle **HTML Development**.
12. Klicken Sie **Weiter**
13. Auf der Seite **Änderungskennzeichen wählen** führen Sie eine der folgenden Aktionen aus:

- Um kein Änderungskennzeichen zu verwenden, klicken Sie **Die Dateien werden keinem Änderungskennzeichen zugeordnet**.
- Um ein Änderungskennzeichen zu verwenden, klicken Sie **Die Dateien sollen folgendem Änderungskennzeichen zugeordnet werden**. Klicken Sie anschließend **...** und wählen Sie das gewünschte Änderungskennzeichen.

14. Klicken Sie **Weiter**.
15. Nachdem die Dateien auf der Seite **Übertragen der Dateien** erfolgreich in die Datenbank übertragen wurden, klicken Sie **Weiter**.
16. Auf der Seite **Beenden des Assistenten** klicken Sie **Fertig**.

Um die HTML-Anwendung in der Datenbank anzulegen

1. Starten Sie das Programm Designer.
2. Navigieren Sie zu **Basisdaten** | **Sicherheitseinstellungen** | **HTML Anwendungen**.
3. Legen Sie einen neuen Eintrag an, indem Sie in der Menüleiste **Objekt** | **Neu** klicken.
4. Vergeben Sie einen Anzeigennamen und geben Sie den Pfad **CCC/<Name Ihrer HTML-Anwendung>** an.

Verwandte Themen

- One Identity Visual Studio Code-Erweiterung installieren auf Seite 18
- Kompilierungszweige auswählen auf Seite 20
- HTML-Anwendungen kompilieren und in Datenbank speichern auf Seite 22

Über uns

One Identity Lösungen eliminieren die Komplexität und die zeitaufwendigen Prozesse, die häufig bei der Identity Governance, der Verwaltung privilegierter Konten und dem Zugriffsmanagement aufkommen. Unsere Lösungen fördern die Geschäftsagilität und bieten durch lokale, hybride und Cloud-Umgebungen eine Möglichkeit zur Bewältigung Ihrer Herausforderungen beim Identitäts- und Zugriffsmanagement.

Kontaktieren Sie uns

Bei Fragen zum Kauf oder anderen Anfragen, wie Lizenzierungen, Support oder Support-Erneuerungen, besuchen Sie https://www.oneidentity.com/company/contact-us.aspx.

Technische Supportressourcen

Technische Unterstützung steht für One Identity Kunden mit einem gültigen Wartungsvertrag und Kunden mit Testversionen zur Verfügung. Sie können auf das Support Portal unter https://support.oneidentity.com/ zugreifen.

Das Support Portal bietet Selbsthilfe-Tools, die Sie verwenden können, um Probleme schnell und unabhängig zu lösen, 24 Stunden am Tag, 365 Tage im Jahr. Das Support Portal ermöglicht Ihnen:

- Senden und Verwalten von Serviceanfragen
- Anzeigen von Knowledge Base Artikeln
- Anmeldung für Produktbenachrichtigungen
- Herunterladen von Software und technischer Dokumentation
- Anzeigen von Videos unter www.YouTube.com/OneIdentity
- Engagement in der One Identity Community
- Chat mit Support-Ingenieuren
- Anzeigen von Diensten, die Sie bei Ihrem Produkt unterstützen