Technik, Dateien und Übersetzungen

Diese Seite richtet sich an alle, die den Editor warten oder ausliefern.

Vier Dateien, die zusammengehören

Datei	Inhalt
`TableServiceDefinitionEditor.html`	die Hülle – kein sichtbarer Text, nur Struktur
`TableServiceDefinitionEditor.css`	Gestaltung
`TableServiceDefinitionEditor.js`	Verhalten
`TableServiceTranslations.json`	die Texte (de/en) und die Vorlage für `lang_properties`

Dazu das Backend: das Portalskript `DEXPRO__TableServiceDefinitionEditor` mit seinen Teilen für DDL und Schreibzugriffe.

Stil und Skript sind mit Absicht nicht inline

Der DOCUMENTS-Client liefert eine Content-Security-Policy der Form `style-src 'self' 'nonce-…'`. Sie blockiert ein inline `<style>`, ein inline `<script>` und **jedes** `style="…"`-Attribut, auch die zur Laufzeit erzeugten. Als eigene Dateien von derselben Herkunft greift `'self'`.

> Ein inline `style="…"` irgendwo im JavaScript zerlegt das Layout im Client, ohne eine zweite Fehlermeldung. Deshalb schaltet der Editor Sichtbarkeit über CSS-Klassen.

`?v=` hochzählen

`TableServiceDefinitionEditor.css?v=<n>` und `…js?v=<n>` tragen den Stand der beiden Dateien. Der DOCUMENTS-Client behält Stil und Skript im Cache, und ein Neuladen der Seite holt sie nicht

wieder. **Wer `.css` oder `.js` ändert, zählt die Zahl in beiden Zeilen hoch** – sonst läuft die Oberfläche beim nächsten Benutzer noch auf dem alten Stand.

Die `id="cssDocuments"` am Stylesheet wird im Skript gesucht: dort hängt der Editor die Signalfarbe des Mandanten als Regel an. Bitte nicht umbenennen.

Übersetzungen

Jeder sichtbare Text trägt im HTML nur seinen technischen Namen (`data-i18n`, `data-i18n-title`, `data-i18n-placeholder`). Eingesetzt wird er beim Start aus zwei Quellen, in dieser Reihenfolge:

1. **Die Übersetzungstabelle des Clients.** Gelesen wie überall in DEX über `DEX.translate(name, datei)`; gefüllt aus `lang_properties` (Application `Documents`, PropType `PropertiesGentable`, PropTypeSubCategory `TableService`) durch `DEXPRO__TableServiceExportTranslations`. **Was dort steht, gewinnt** – eine Übersetzung lässt sich also ändern, ohne eine Datei anzufassen.
2. **`TableServiceTranslations.json`** neben der Seite, als Rückfallebene und zugleich als Vorlage, aus der das Exportskript die Zeilen schreibt.

Fällt beides aus, bleibt der technische Name stehen – unschön, aber die Oberfläche bleibt bedienbar. Woher die Texte tatsächlich kommen, steht im Tooltip des Abzeichens rechts oben; das ist die erste Frage, wenn alles in einer Sprache erscheint.

Ein **neuer Text** gehört deshalb nie in die HTML-Datei, sondern in den Katalog – und von dort mit `DEXPRO__TableServiceExportTranslations` in die Tabelle.

Die Properties-Datei heißt `Gentable`, weil `getMessages()` nur Dateien im Tomcat-Kontext findet: `DEXPRO\Gentable` hängt in der `documents.xml` unter `/WEB-INF/classes`, und dort liegt `Gentable_<lang>.properties` mit unseren Texten unter `# SubCategory: TableService`.

Definitionsdateien bleiben unangetastet

Die Dateien unter `TableService/data/table-definitions` werden vom Editor **nicht** angefasst. Sie bleiben die Quelle der Auslieferung, und ein erneuter Lauf von `DEXPRO__CreateCustomPropertiesFromAllTableServices` überschreibt alles, was hier geändert wurde.

Für Dauerhaftes also **Exportieren** (siehe Definitionen importieren und exportieren) und das Ergebnis in die Auslieferung geben.

Revision #2
Created 2026-09-17 11:04:00 UTC by Helge Czerwinski
Updated 2026-09-17 11:04:04 UTC by Helge Czerwinski