

# Development

Marvin Rohrbach

Noah Pfeffel

Dr. Veit Jahns

Martin Kozianka

Sebastian Straub

Thomas Richter

# Development

## Inhalt

- DOCUMENTS Manager 2.0 (Marvin Rohrbach, Martin Kozianka)
- OAuth2.0 (Noah Pfeffel, Sebastian Straub)
- Scripting Libraries (Dr. Veit Jahns)
- VS Code Extension (Thomas Richter)

# DOCUMENTS Manager 2.0

## Neues Design

- Moderner
- Touchfreundlicher

## Neue Funktionen

- Diagramme
- Codeeditor
- Propertyeditor

The screenshot shows the DOCUMENTS Manager 2.0 application window. The left sidebar displays a navigation tree with categories like Administration, Benutzermanagement, Documents, Mappentypen, and Scripts. The main area is divided into two sections: a diagram editor and a property editor.

**Diagram Editor:** Shows a UML-like class diagram for the `crmProduct` entity. The `crmProduct` class has attributes: `crmId : String`, `crmName : String`, `crmDescription : Text`, `crmListPrice : Numerisch`, `crmUnit : String`, `crmCategory : Aufzählung`, `crmDiscontinued : Checkbox`, `crmPurchase : Numerisch`, and `crmPurchaseCode : String`. It has associations with `crmAccount` (multiplicity 0..1), `crmOpportunities` (multiplicity 0..1), and `crmOpportunity` (multiplicity 0..1).

**Property Editor:** Shows properties for the `crmProduct` entity. The `afterMailScript` property is set to `crmProduct_MailSe...`. The `FulitextSearch` property is set to `1`.

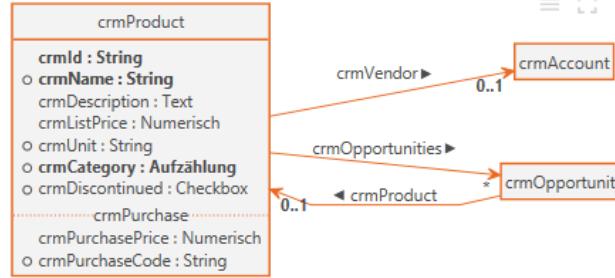
**Right Panel:** A list of properties for the `crmProduct` entity, each with a checkbox indicating its status or configuration.

Property	Value	Status
Letzter Bearbeiter	admin am 07.01.2020 11:48:00	
Ersteller	admin am 25.09.2006 10:59:54	
Name	crmProduct	
Freigegeben	✓	
Bezeichnung	de:Produkt;en:Product	
Automatischer Titel	%crmName% (%crmId%)	
Titel anzeigen		
Mappentypname als Mappendeckebezeichner verwenden	✓	
Eigentümer/Letzter Bearbeiter/Wiedervorlage unten darstellen	✓	
Titel in Suchmaschen		
Titel in Trefferlisten		
Eigentümer/Datum in Trefferliste		
Letzter Bearbeiter/Datum in Trefferliste		
Icon	crmProduct.gif	
Monitor anzeigen		
Mappen-Id	relations_fi2006000000005	
im Status protokollieren		
Mappentyp ändern		
im Status protokollieren		
Leitbeleg drucken		
im Status protokollieren		
PDF erstellen (drucken)		



- > Administration
- > Benutzermanagement
- Documents
  - > Outbars 3
  - > öffentliche Ordner 17
  - Mappentypen 19
    - > appFileConfig
    - > appListConfig
    - > appMainConfig
    - > crmAccount
    - > crmAppointment
    - > crmCampaign
    - > crmCase
    - > crmContact
    - > crmDistributionList
    - > crmKBArticle
    - > crmLead
    - > crmNote
    - > crmOpportunity
    - > crmProduct
    - > crmQuote
    - > crmServiceTeam
    - > crmTask
    - > crmUser
    - > otrTextTemplate
  - Mappentyp-Kategorien 0
  - Mappenverknüpfung 0
  - Suchkategorien 0
  - > Sendelisten 1
  - > Workflows 5
    - Aktenpläne 0
    - # Nummernkreise 6
  - Scripte 193
    - > appDocumentsGateway 100
    - > appScripts 18
    - > crmCase 10

## crmProduct



### Name ▾ Wert

afterMailScript crmProduct\_MailSe...

FulltextSearch 1

Property hin...

### Property Editor

Wählen Sie ein Property um zu beginnen. Drücken Sie Enter oder klicken Sie doppelt auf eine Zelle um diese zu bearbeiten oder verwenden Sie die letzte Zeile um ein neues Property hinzuzufügen.

**Automatisches Layout**  
Je nach verfügbarem Platz

Letzter Bearbeiter

**Menü**  
Aktiviert erweiterte Funktionen

Automatischer Titel

Titel anpassen

Mapper anpassen

Mapper anpassen

Eigentümer/Letzter  
Bearbeiter/Wiedervorlage unten darstellen

Titel in Suchmarken

Titel in Trefferlisten

Eigentümer/Datum in Trefferliste

Letzter Bearbeiter/Datum in Trefferliste

Icon

crmProduct.gif

Monitor anzeigen

Mappen-Id

relations\_fi20060000000005

im Status protokollieren

Mappentyp ändern

im Status protokollieren

Leitbeleg drucken

im Status protokollieren

PDF erstellen (drucken)

**Vollbild**  
Zeigt die Kachel im gesamten Bereich an

**Minimieren**  
Blendet die Kachel aus

# Property Editor

## Kategorien

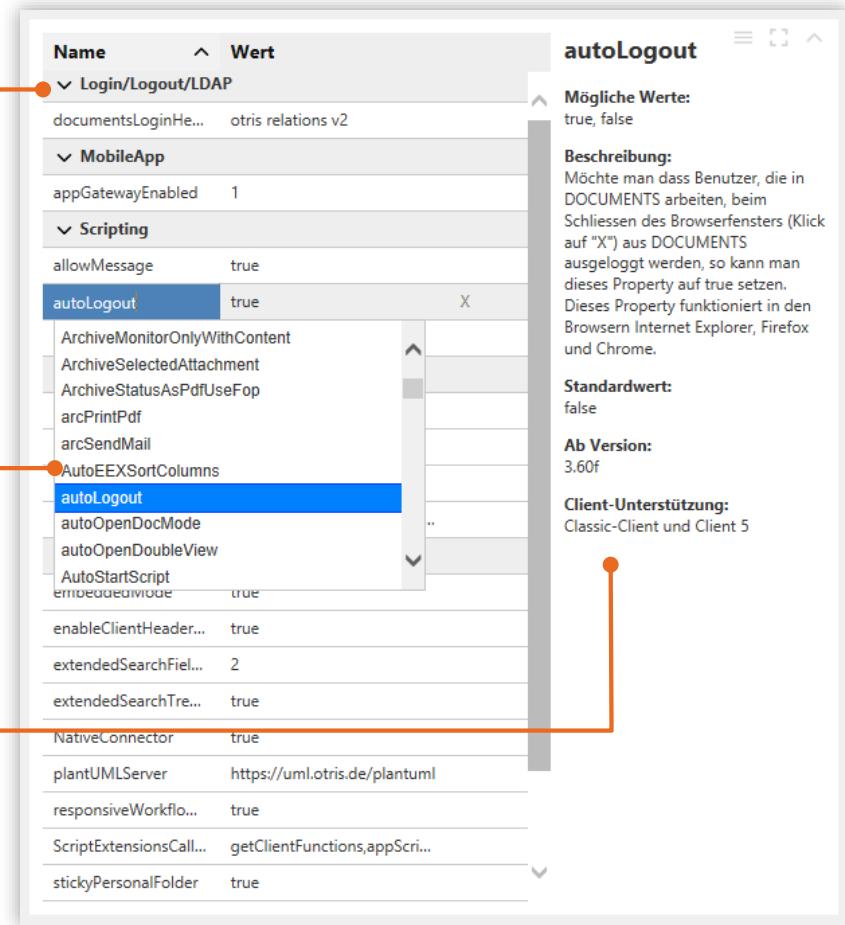
- Properties können nach Kategorien sortiert angezeigt werden
- Konfigurierbar über den Menüknopf oben rechts

## Autovervollständigung

- Properties und deren Werte werden automatisch vorgeschlagen

## Dokumentation

- Die Dokumentation der selektierten Eigenschaft wird rechts angezeigt



# Skript Editor

## Suchfunktion

- Skript kann im Manager durchsucht werden (Strg+F)
- Ersetzungen sind auch möglich (Strg+H)

## Syntaxhervorhebung

## Auto vervollständigung

- Bereits verwendete Elemente werden automatisch vervollständigt

## Speichern und Ausführen

The screenshot shows a script editor interface with the following features highlighted:

- Search function:** A search bar at the top right labeled "Search for" with the placeholder ".\* Aa \b S".
- Syntax highlighting:** The code uses color-coded syntax highlighting for various programming elements.
- Auto-completion:** A dropdown menu is open at the bottom of the code area, listing suggestions like "context", "local", "continue", etc.
- Save and Run buttons:** At the bottom right, there are two buttons: "Speichern" (Save) and "Ausführen" (Run).

```
1 // Delete connected files
2 // Albrecht 24.0
3 + 0 of 0
4 var docFile = context.local;
5 if (docfile)
6 {
7     var leadId = docFile.crmId;
8
9 // =====
10 // DELETE CONNECTED FILES
11 // =====
12 var docTypes = new Array("crmTask", "crmNote", "crmAppointment");
13 var filter = "crmLead|~" + leadId + "";
14 var sortOrder = "";
15
16 cont
17 context local
18 continue keyword
19 crmAppointment local
20 CONNECTED local
21 connected local
22 const keyword
23 _count_ keyword
24 decodeURIComponent keyword
25
26
27 myRS = null;
28 }
29
30 } else {
31     context.errorMessage = "Not in a file context!";
32     return -1;
33 }
```

# Detail Ansicht

## Darstellung der Objektattribute

### Zwei Ansichten

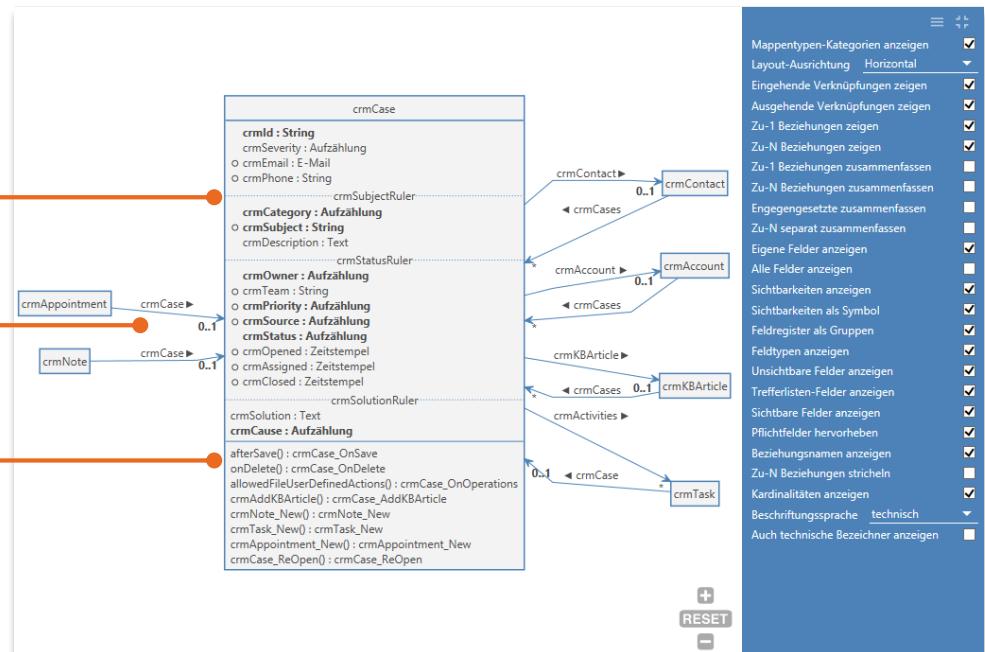
- Alle Attribute
- Attribute, welche verändert wurden
  - D.h. aktueller Wert != Standardwert
- Umgeschaltet wird mit dem Menüknopf oben rechts

Letzter Bearbeiter	admin am 07.01.2020 11:48:00
Ersteller	admin am 25.09.2006 10:59:54
Name	crmProduct
Freigegeben	✓
Bezeichnung	de:Produkt;en:Product
Automatischer Titel	%crmName% (%crmId%)
Titel anzeigen	
Mappentypname als Mappendeckelbezeichner verwenden	✓
Eigentümer/Letzter Bearbeiter/Wiedervorlage unten darstellen	✓
Titel in Suchmaske	
Titel in Trefferlisten	
Eigentümer/Datum in Trefferliste	
Letzter Bearbeiter/Datum in Trefferliste	
Icon	crmProduct.gif
Monitor anzeigen	
Mappen-Id	relations_fi20060000000005
im Status protokollieren	
Mappentyp ändern	
im Status protokollieren	
Leitbeleg drucken	
im Status protokollieren	
PDF erstellen (drucken)	
im Status protokollieren	
Mappe archivieren	
im Status protokollieren	
\$FulltextSearch	1
\$afterMailScript	crmProduct_MailSent#

# Mappentyp Diagramm

## Grafische Aufbereitung des Mappentypen

- Felder
- Beziehungen
- Aktionen



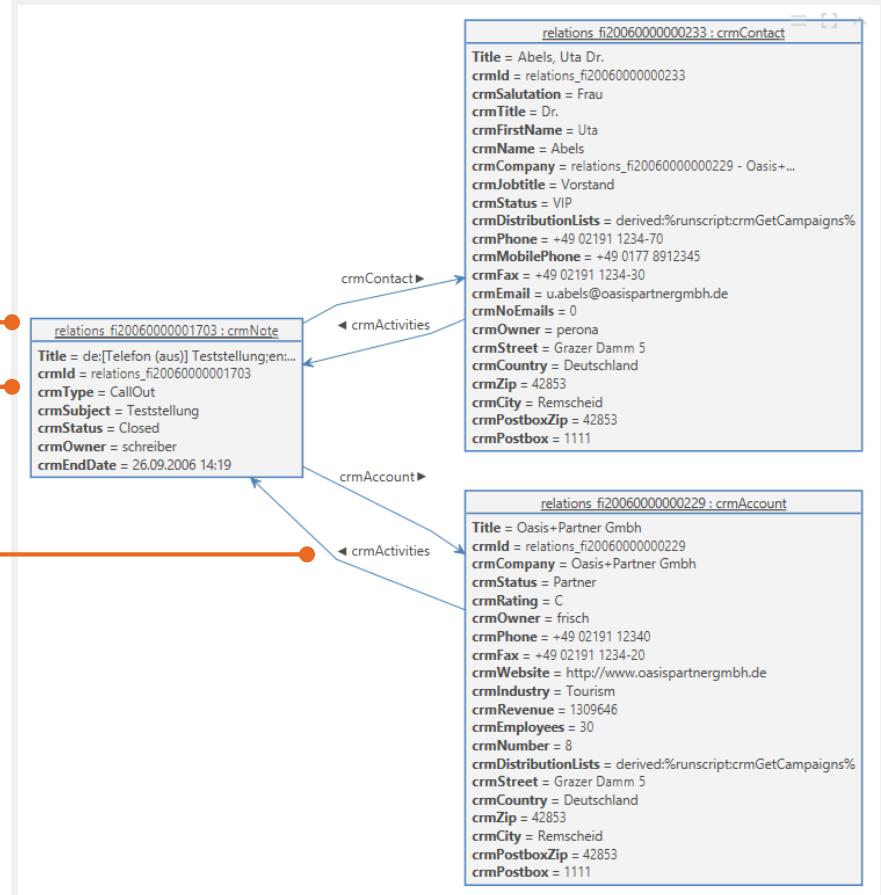
## Erweiterte Konfigurationen

- Erreichbar über den Menüknopf oben rechts

# Akten Diagramm

## Grafische Aufbereitung eines Vorgangs

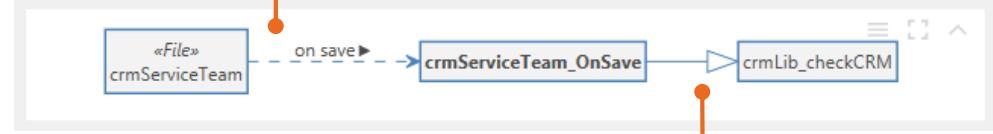
- Mappenid und Typ
- Felder und Feldwerte
- Beziehungen



# Skript Diagramm

## Einbindungspunkte des Skripts

- Hinweis: Nicht alle Einbindungspunkte werden untersucht, daher keine Garantie für Vollständigkeit



## Importierte Skripte

- Nur `#import` wird unterstützt!

# Exkurs: PlantUML

Um die Diagrammfunktionen zu Nutzen, muss ein PlantUML Server konfiguriert werden!

- Documents Einstellungen → Eigenschaften → plantUMLServer

Die otris stellt einen PlantUML Server bereit

- <https://uml.otris.de/plantuml>
- Kann auch direkt aufgerufen werden

Eigenes Hosting möglich

- Mehr Informationen: <https://plantuml.com/>

The screenshot shows the otris UML Playground interface. On the left, there is a code editor window containing PlantUML code. The code defines a class `Vehicle` with attributes `serialNo`, `plate`, `make`, and `purchased` (set to `CurrentDate`). It also defines a class `Document` with attributes `Title`, `Category` (set to `DocCategory`), and `Date`. An enum `DocCategory` has three values: `Category A`, `Category B`, and `other`. A constraint at the bottom states `Vehicle "1" - "*" Document: belongs to < * >|`. On the right, there are three UML class diagrams labeled `Vehicle`, `Document`, and `DocCategory`. The `Vehicle` class has attributes `serialNo`, `plate`, `make`, and `purchased`. The `Document` class has attributes `Title`, `Category` (set to `DocCategory`), and `Date`. The `DocCategory` class has three categories: `Category A`, `Category B`, and `other`. A relationship named `belongs to` connects `Vehicle` to `Document`. At the bottom of the interface, there are links to the otris Portal, Impressum, and Datenschutzerklärung.

# EAS-Manager

Darstellung des EAS-Managers  
innerhalb des DOCUMENTS Managers

## Folgende Voraussetzungen

- EAS Administratorkonto und Passwort müssen am Server konfiguriert sein
- Falls Manager extern gestartet wird: Portfreigabe des EAS-Servers

The screenshot shows the EAS Manager interface within a DOCUMENTS Manager application. The top navigation bar displays 'EAS | Manager'. The main content area is divided into several sections:

- Storestatus**: A section for managing time intervals (Zeit) with fields for Anfang, Ende, and Zeitspanne, each accompanied by a calendar icon.
- Zeit**: A button labeled 'Anzeigen' (Display) is located here.
- Registry**: A table showing the count of records and attachments. The table has columns: - (empty), Gesamtanzahl, and Indiziert.

-	Gesamtanzahl	Indiziert
records	0	0
attachments	0	0
- Index**: A table showing index status for documents and current status. The table has columns: Dokumente, Aktuell, and Besitzt gelöschte Mappen.

Dokumente	Aktuell	Besitzt gelöschte Mappen
0	ja	nein

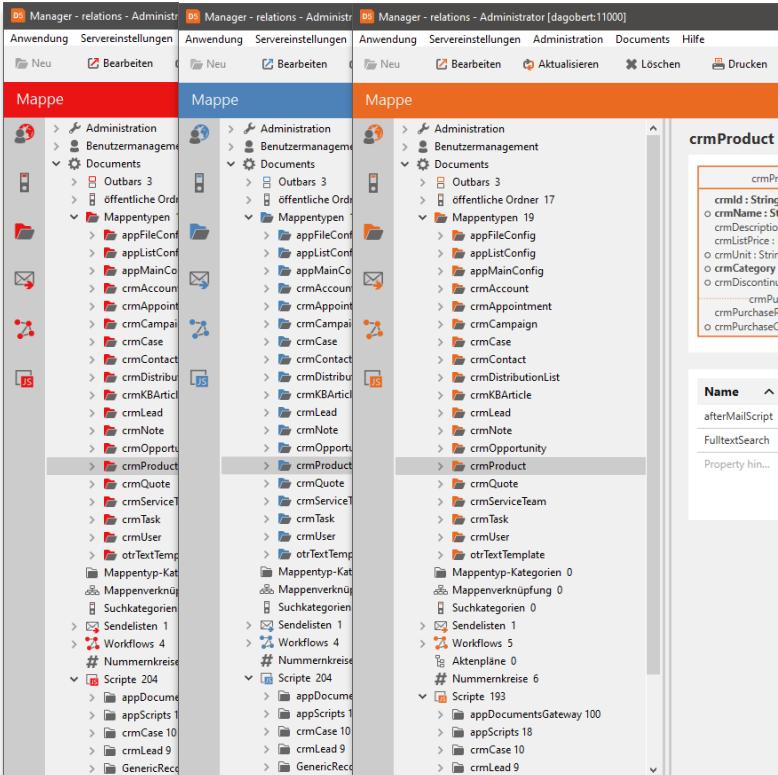
# Manager Farbschema anpassen

## Anpassung der Farbe des Managers

- Bsp: Rot als Warnhinweis für Produktivsystem

## Serverweite Einstellungen in der documents.ini

- Highlightfarbe → \$DMSkinColor1 234,20,20
- Hintergrundfarbe → \$DMSkinColor2 80,80,80
- Einstellung in [Rot],[Grün],[Blau]



# OAuth2.0

## Inhalt

- Grundlagen
- OneDrive Anbindung
- Signatur mit DocuSign

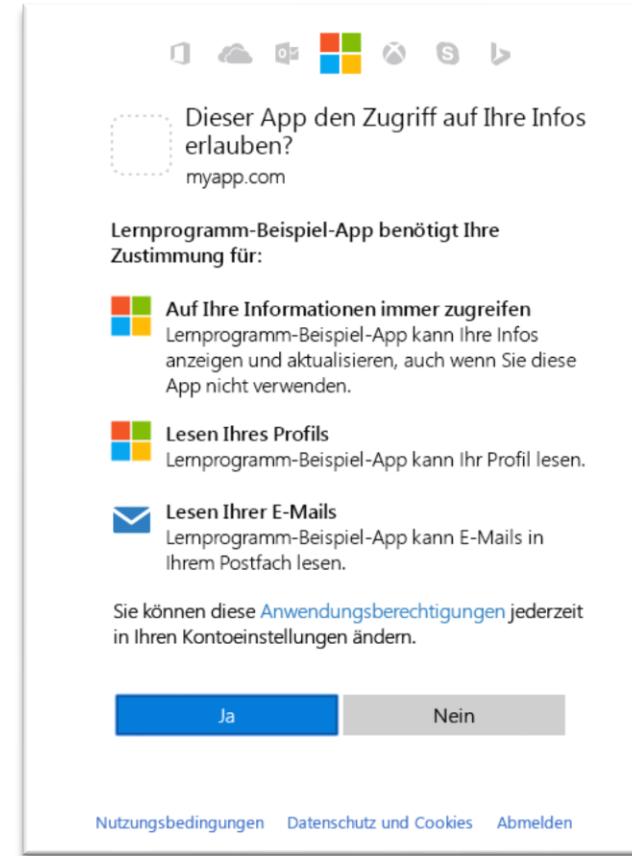
# Was ist OAuth2.0?

- Kurz für “Open Authorization” in Version 2.0
- Standardprotokoll für sichere API-Autorisierung
- Dient als Schnittstelle zum sicheren übermitteln von Daten zwischen Anwendungen



# Stichwort Autorisierung

- Kein Weitergeben der Login-Daten
- Nur beschränkter Zugriff
- Zugriff kann von beiden Seiten entzogen werden



# Wer unterstützt OAuth2.0?



Und viele mehr...

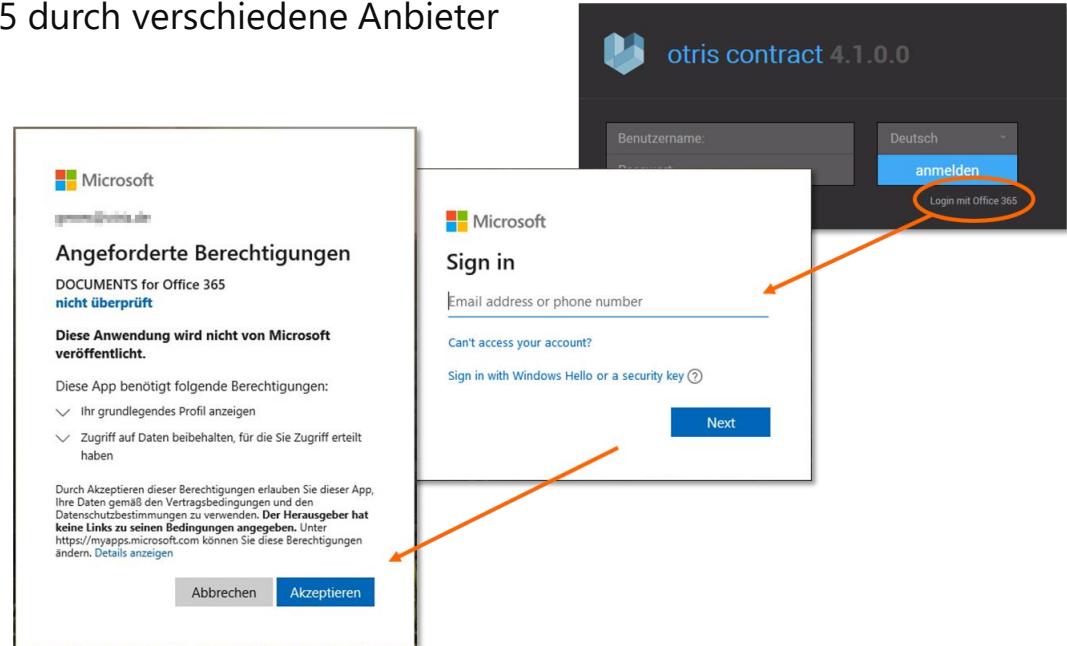
# Login

OpenID Connect als Aufsatz für OAuth2.0

Ermöglicht den Login in Documents5 durch verschiedene Anbieter

Features je nach Anbieter:

- SSO
- Massenimport der Benutzer
- Individueller Benutzerimport
- 2-Faktor Authentifizierung



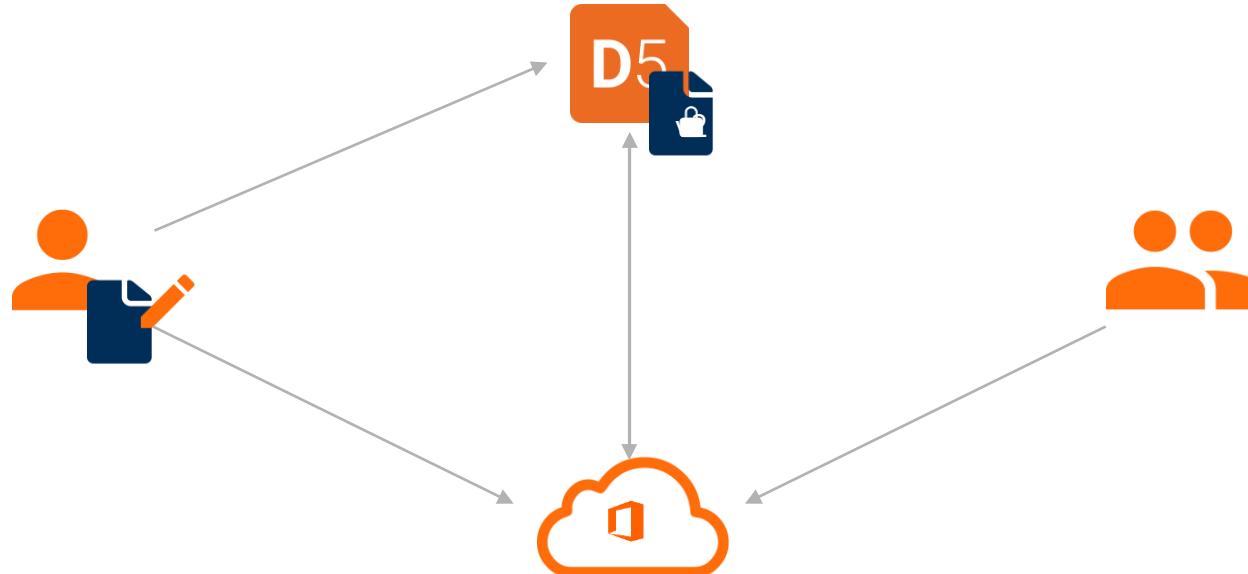
# Login mit Benutzerimport



# OneDrive

Kollaboratives Arbeiten

# Kollaboratives Arbeiten in DOCUMENTS

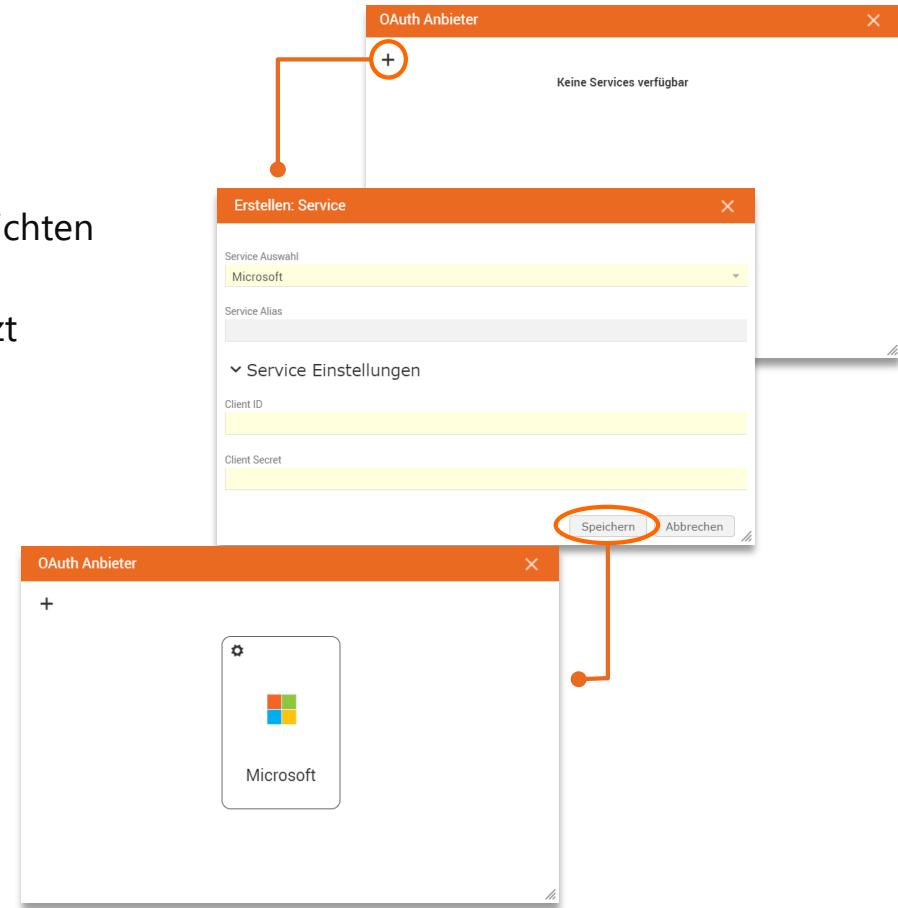


# Einrichtung

- Administratoren können Verbindungen einrichten
- Microsoft wird als erster Anbieter unterstützt

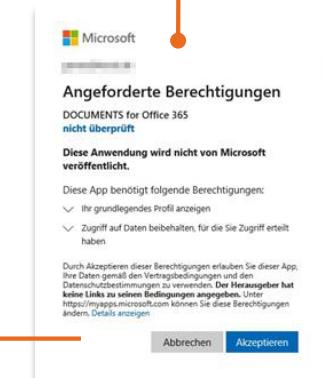
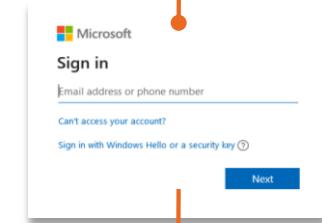
## Aktuell eingebaute Microsoft Features:

- Benutzer für Dokumente freigeben
- Dokumente hochladen
- Dokumente herunterladen
- Benutzer importieren



# Anmeldung

- Benutzer können sich bei eingerichteten Diensten anmelden
- Wenn SSO möglich ist und eine Session besteht, fallen die Microsoft-Schritte aus
- Der angemeldete Benutzer kann jetzt Microsoft-Funktionen in DOCUMENTS benutzen



# Dokumente Hochladen

- Voraussetzung: Benutzer ist bei einem Cloud-Service angemeldet
- Dokument auswählen und Hochladen-Button klicken



The screenshot shows the otris DOCUMENTS application interface. At the top, there's a header bar with a logo, the title "Individualanpassung DOCUMENTS | Gewonnen 50000,00€ ☆", and several buttons: "Bearbeiten", "Intern anzeigen", "Extern anzeigen", "Download", "Aktionen", and "Aktionen". Below the header is a table listing a single document:

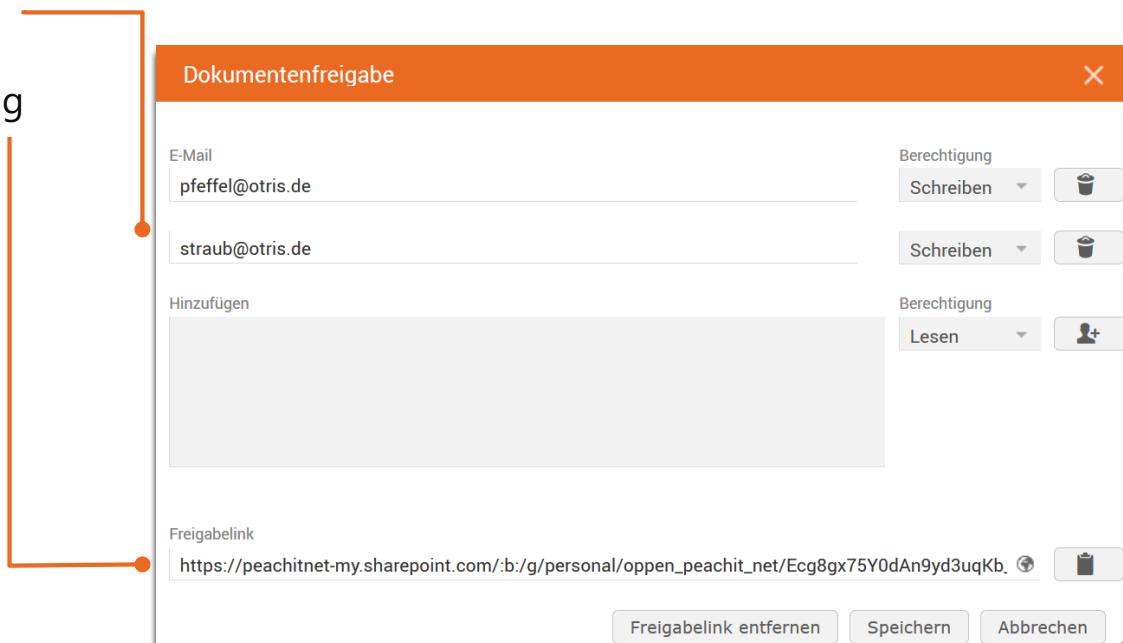
Dokumentenname	Größe	geändert am	um	Version
Vertragsdokument.pdf	34.8 KB	17.02.2020	16:36	1.0

To the right of the table, there's a sidebar with company information:

- Firma: Heckmaier Kunststoff GmbH
- Verkaufsprojekt
- Angebote: 0
- Anhänge: 1
- Status
- Kommentare: 0

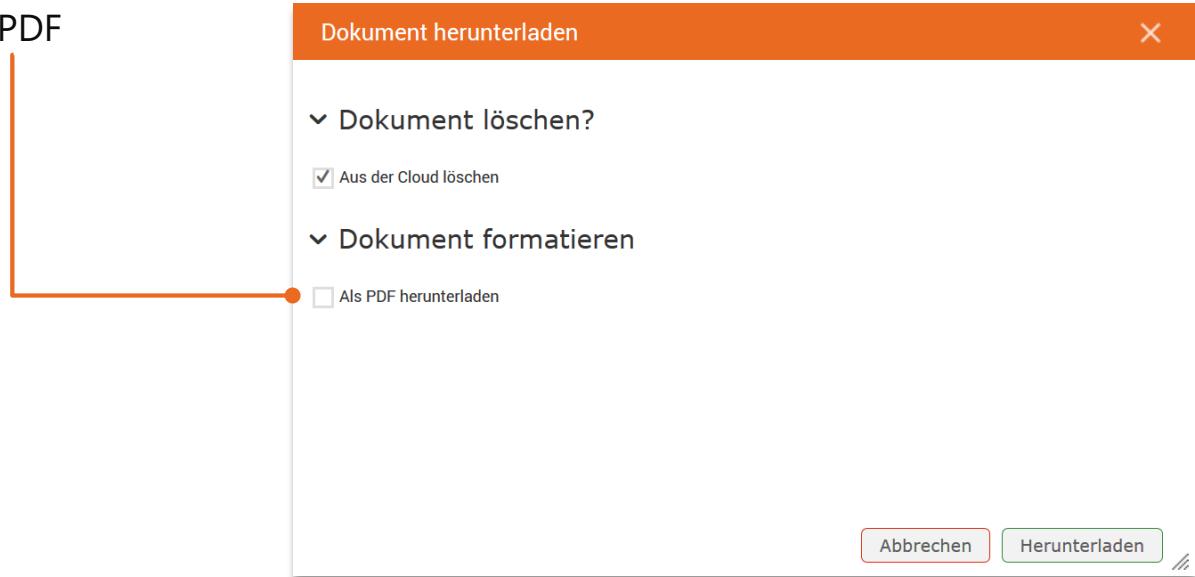
# Dokumente Teilen

- Voraussetzungen: Dokument ist in der Cloud & Benutzer ist berechtigt
- Personen individuell berechtigen
- Freigabelink mit Leseberechtigung



# Dokumente Herunterladen

- Voraussetzungen: Dokument ist in der Cloud & Benutzer ist berechtigt
- Bei Änderungen wird das alte Dokument versioniert
- Optional: Herunterladen als PDF



# DocuSign

Verträge elektronisch signieren

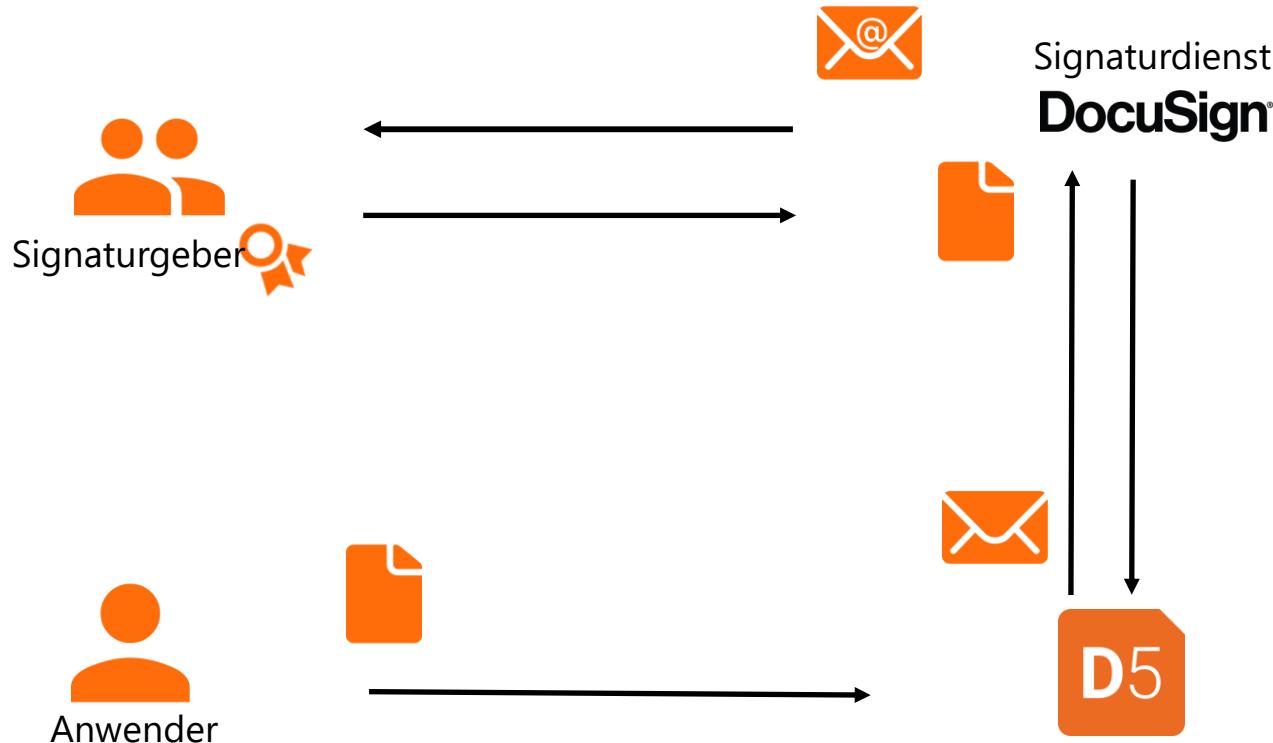
# Die elektronische Signatur

**Unter einer elektronischen Signatur versteht man**

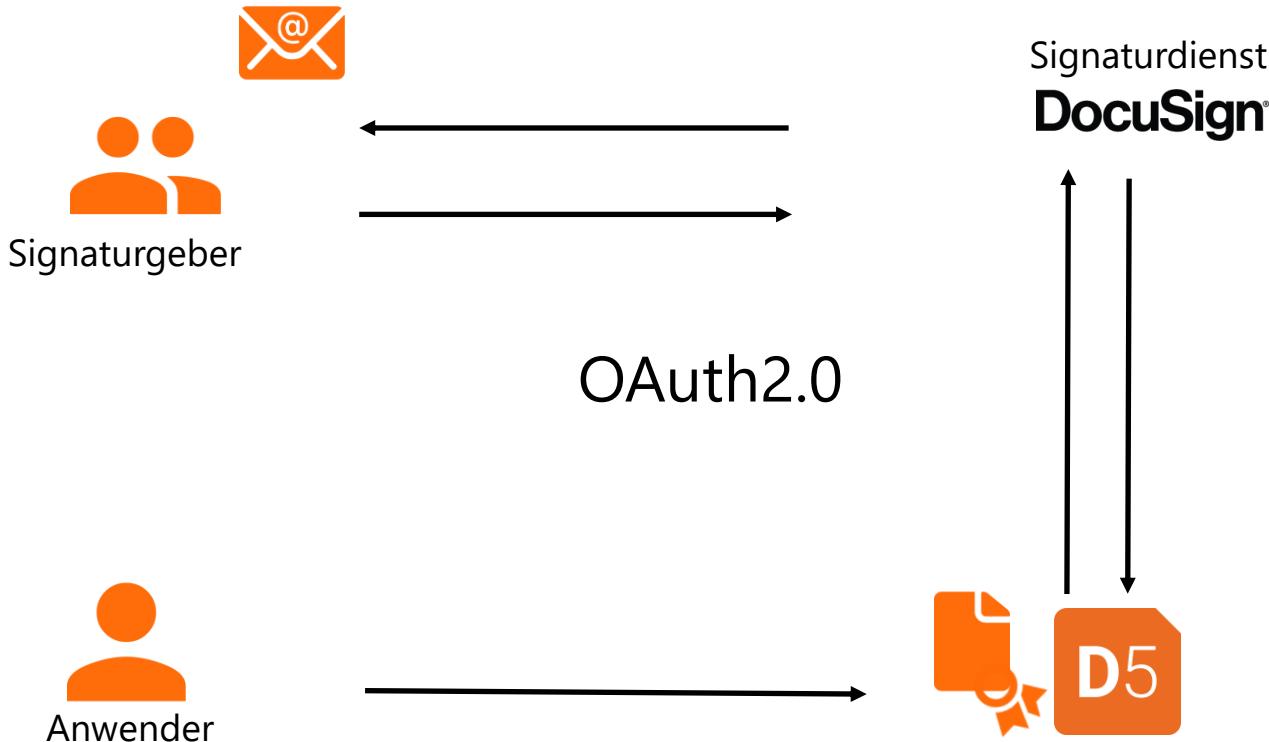
- Informationen in elektronischer Form, die
- logisch mit anderen Informationen (bspw. Dokumenten) verknüpft sind und
- zur Identifizierung des Signaturerstellers dienen.

**Der Unterzeichner fügt einem Dokument weitere Daten hinzu, die er als Unterschrift verwendet.**

# Ablauf einer Unterschrift



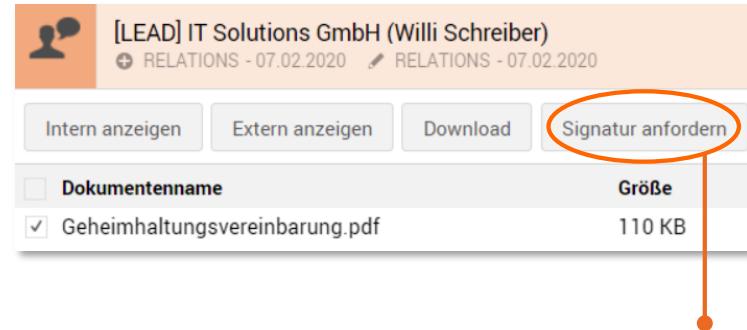
# Ablauf einer Unterschrift



# Unterschrift anfordern

## Einheitlicher Prozess für alle Versendungen

- Einfaches Anfordern von Unterschriften
- Ohne Medienbruch direkt aus dem Dokumentenregister
- Geführter Dialog mit Datenvorauswahl
- Identischer Eingang unterschriebener Dokumente



The screenshot shows the 'Signatur anfordern' dialog box. It has tabs for 'Allgemeines' and 'Unterzeichner'. Under 'Unterzeichner', there are two entries: 'Sebastian Straub' with email 'straub@otris.de' and 'Willi Schreiber' with email 'schreiber@otrisdemo.de'. Both entries have 'Entfernen' (Delete) buttons. Below these tabs is a 'Nachricht' section with a recipient field. At the bottom are buttons for 'Abbrechen' (Cancel), 'Zurück' (Back), and 'Weiter' (Next). There is also an 'Erweitern' (Expand) button in the signature area.

# Monitoring und Überwachung

## Erinnerungen verschicken

- Im Prozess ausstehende Signaturgeber benachrichtigen

## Prozessüberwachung im Ampelsystem

- Gesamtstatus des Signaturdokuments
- Status der einzelnen Signaturgeber

The screenshot shows a software interface for monitoring and managing a signature process. At the top, there are buttons for 'Bearbeiten' and 'Aktionen'. The 'Aktionen' button is highlighted with an orange circle. Below it, a document preview is shown with the title '[LEAD] IT Solutions GmbH (Willi Schreiber) - Geheimhaltungsvereinbarung'. The status of the document is listed as 'Abgeschlossen' (Completed), indicated by a green dot. To the right, it shows the service provider as 'DocuSign'. The 'Felder' (Fields) section on the right lists 'Auslösender Vorgang' (Triggering Action) as 'IT Solutions GmbH' and 'Anfragender' (Requester) as 'Administrator RELATIONS'. The 'Dokumente' section shows the document itself, and the 'Status' section shows the status of individual signers. A table at the bottom lists signers with columns for Nr (Number), Status (Status), Rolle (Role), Name (Name), and E-Mail (Email). The first signer (Nr 1) has a red dot in the status column, while the second signer (Nr 2) has a green dot.

Nr	Status	Rolle	Name	E-Mail
1	Red Dot		Sebastian Straub	straub@otris.de
2	Green Dot		Willi Schreiber	straub@otris.de

# OAuth2.0

## Zusammenfassung

- ✓ Ein standardisiertes Protokoll zur Autorisierung
- ✓ Wird von vielen Anbieter unterstützt
- ✓ Einrichtung und Anmeldung kann über DOCUMENTS erfolgen
- ✓ Anmeldung bei Documents über OpenID Connect

# Scripting Libraries

## Inhalt

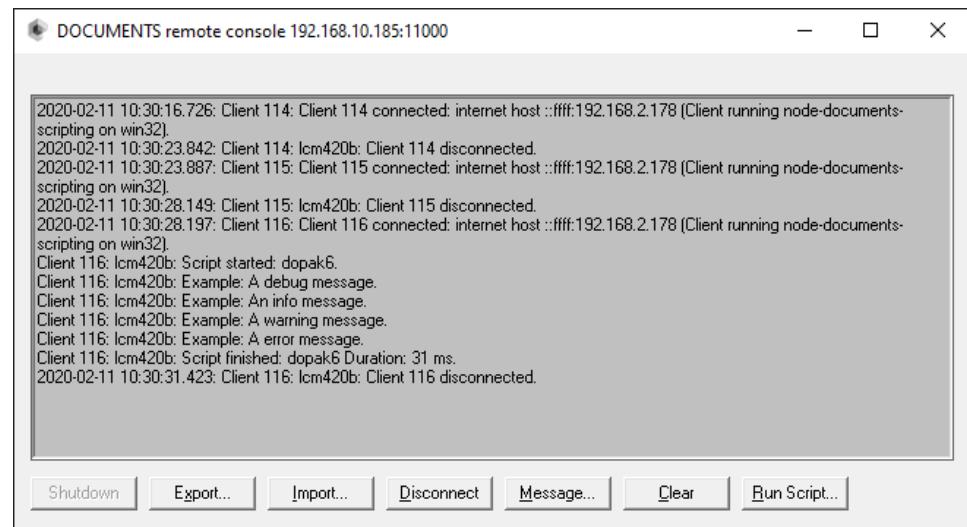
- otrLogger: Erweitertes Logging
- otrAssert: Erweiterte Fehlerbehandlung

# otrLogger: Erweitertes Logging

## Log-Ausgabe per util.out()

- Unstrukturiert
- Keine Filtermöglichkeit
- Erschwert Fehlersuche
- Unnötige Log-Ausgaben im laufenden Betrieb

```
util.out("Example: A debug message");
util.out("Example: An info message");
util.out("Example: A warning message");
util.out("Example: A error message");
```



## otrLogger

- Ausgabe pro Log-Level
- Filtern nach Log-Level
- Filtern nach Log-Kontext

# Ausgabe von Log-Meldungen

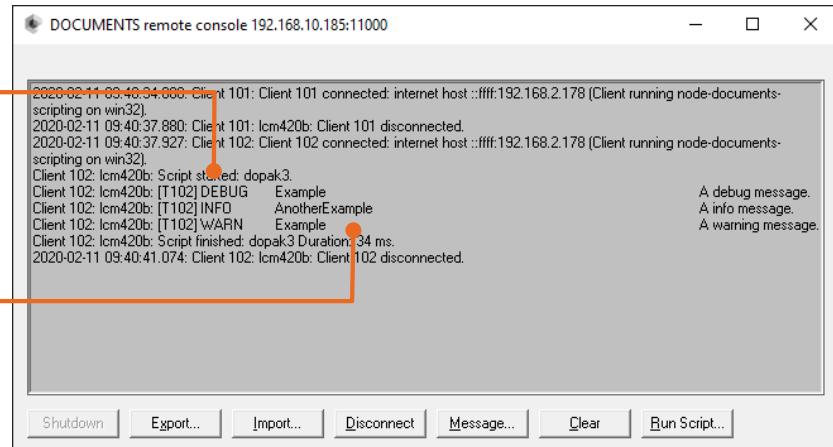
```
context.enableModules();
var otrLogger = require("otrLogger");

otrLogger.logDebug("Example", "A debug message");
otrLogger.logInfo("AnotherExample", "An info message");
otrLogger.logWarn("Example", "A warning message");
```

## Level einer Log-Meldung

- `logDebug()`, `logWarn()` ,`logInfo()`,  
`logError()`, `logFatal()`

## Kontext der Log-Meldung

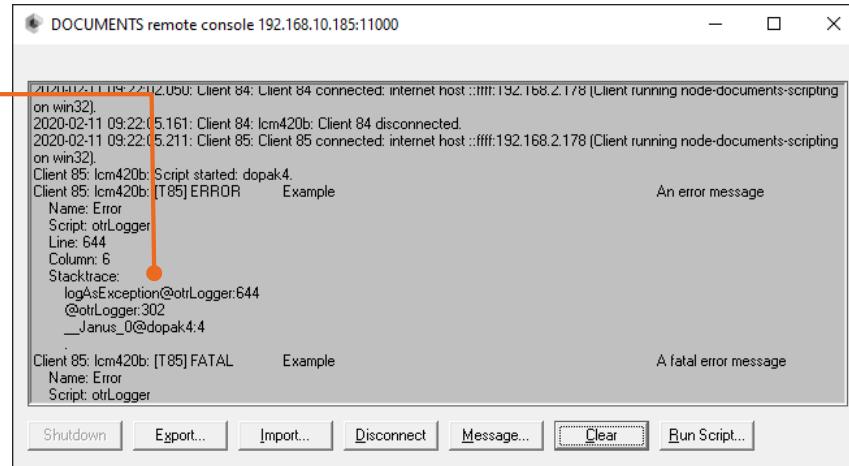


# Ausgabe von Fehlermeldungen

```
context.enableModules();
var otrLogger = require("otrLogger");

otrLogger.LogError("Example", "An error message");
otrLogger.logFatal("Example", "A fatal error message");
```

- Automatische Ergänzung des Stacktrace
- Exception als Log-Meldung
- Fatale Fehlermeldungen werden gecacht



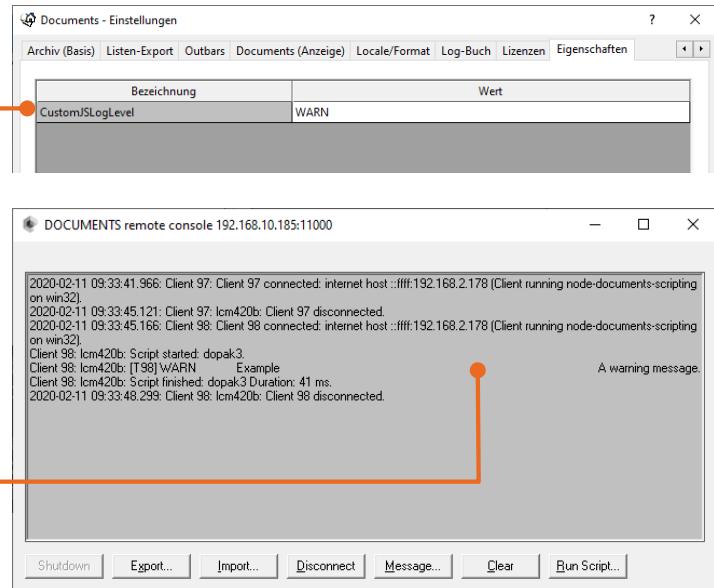
# Filtern von Log-Meldungen

```
context.enableModules();
var otrLogger = require("otrLogger");

otrLogger.logDebug("Example", "A debug message");
otrLogger.logInfo("AnotherExample", "An info message");
otrLogger.logWarn("Example", "A warning message");
```

## Filtern per Log-Level

- Per \$CustomJSLogLevel
- Mindest-Level  
DEBUG > INFO > WARN >
- ERROR > FATAL > NONE



# Filtern von Log-Meldungen

```
context.enableModules();
var otrLogger = require("otrLogger");

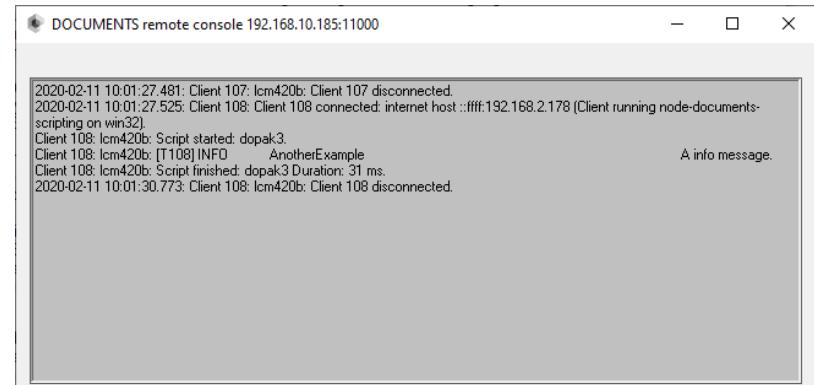
otrLogger.logDebug("Example", "A debug message");
otrLogger.logInfo("AnotherExample", "An info message");
otrLogger.logWarn("Example", "A warning message");
```

## Filtern per Log-Kontext

- Per \$CustomJSContextFilter
- Regulärer Ausdruck auf den Kontext einer Log-Meldung



Bezeichnung	Wert
CustomJSLogLevel	DEBUG
CustomJSContextFilter	^Another



```
2020-02-11 10:01:27.481: Client 107: lcn420b: Client 107 disconnected.
2020-02-11 10:01:27.525: Client 108: Client 108 connected: internet host :ffff:192.168.2.178 (Client running node-documents-scripting on win32).
Client 108: lcn420b: Script started: dopak3.
Client 108: lcn420b: [T108] INFO      AnotherExample
Client 108: lcn420b: Script finished: dopak3 Duration: 31 ms.
2020-02-11 10:01:30.773: Client 108: lcn420b: Client 108 disconnected.
```

Shutdown Export... Import... Disconnect Message... Clear Run Script...

# Stacktrace in Fehlermeldungen

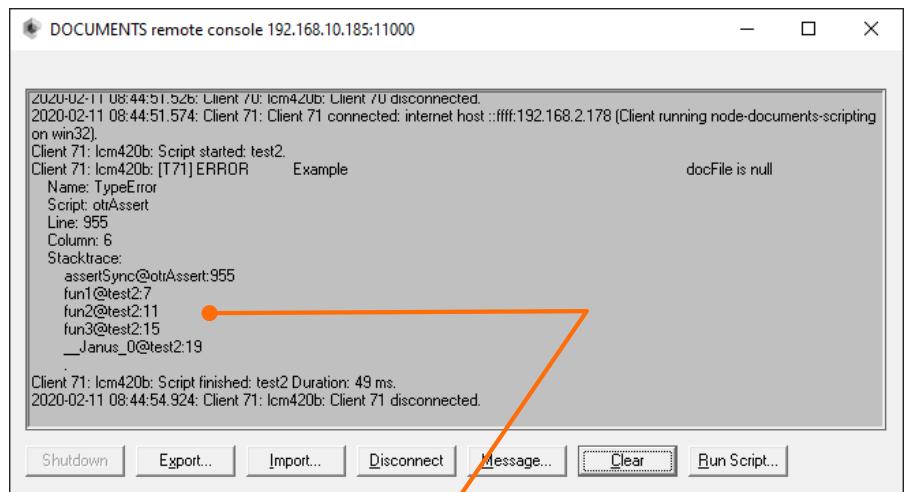
```
context.enableModules();
var otrLogger = require("otrLogger");

function fun1(file) {
    var otrAssert = require("otrAssert");
    // [...] some code [...]
    otrAssert.assertSync(file, "fun1");
}

function fun2(file) {
    fun1(file);
}

function fun3(file) {
    fun2(file);
}

try{
    fun3(context.file);
} catch (e) {
    otrLogger.logError("Example", e);
    return e.message;
}
```



## Stacktrace

Für alle Fehlermeldung und Exceptions als Log-Meldung

# otrAssert: Erweiterte Fehlerbehandlung

## Fehlerbehandlung im PortalScripting

- Auswertung der Fehlerflags
- Auswertung von getLastError()

## Nachteile

- Komplexe Kontrollpfade
- Code wird unübersichtlich
- Code wird wartungsanfällig

```
var file = context.file;

if (!file.startEdit()) {
    return file.getLastErrorMessage();
}

file.field = "New value";

// [...] more code [...]

if (!file.commit()) {
    var commitError = file.getLastErrorMessage();

    if (!file.abort()) {
        return commitError + "\n" +
            file.getLastErrorMessage();
    } else {
        return commitError;
    }
}
```

# PortalScripting-API „mit“ Exceptions

## Wrapper um Funktionen der API

- Automatische Auswertung der Fehlerflags
- Werfen von Exception

## Vorteile

- Vereinfachung der Fehlerbehandlung
- Fehlerbehandlung mit try-catch-finally
- Ausgabe des Stacktrace in Kombination mit otrLogger

```
context.enableModules();
var otrAssert = require("otrAssert");
var otrLogger = require("otrLogger");
var file = context.file;

try {
    otrAssert.assertStartEdit(file, "Example");

    file.field = "New value";

    // [...] more code [...]

    otrAssert.assertCommit(file, "Example");
} catch (e) {
    if (file.getOriginal()) {
        otrAssert.assertAbort(file, "Example", e);
    }
    otrLogger.logError("Example", e);
}
```

# PortalScripting-API „mit“ Exceptions

## Vorteile (Fortsetzung)

- Bereinigung mit try-finally
- Für Erfolgs- und Fehlerfall
- Nutzerwechsel, geöffnete Dateien, Datenbankverbindungen

```
// [...]
var oldUser = context.currentUser;
// no evaluation of the return flag for
// the sake of simplicity
context.changeScriptUser(technicalUser);

try {

    file.field = "New value";

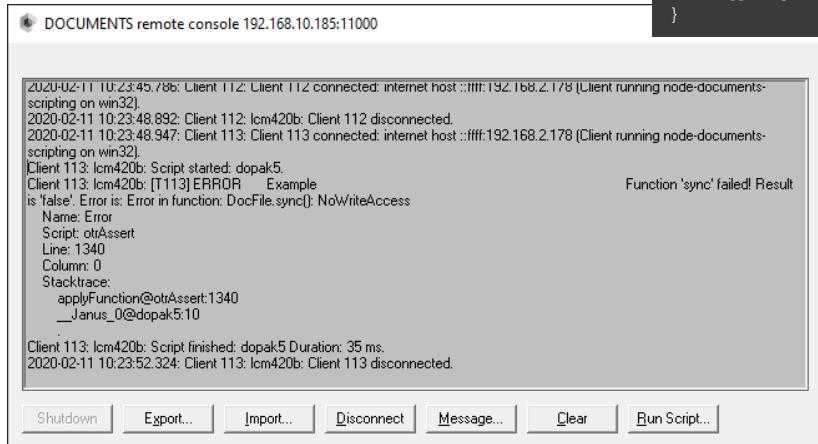
    // [...] more code [...]

    otrAssert.assertSync(file, "Example");
} finally {
    context.changeScriptUser(oldUser);
}
```

# PortalScripting-API „mit“ Exceptions

## Wrapper durch JS-Proxies

- Für beliebige Funktionen eines Objekts der PortalScripting-API
- Default-Listen für context und DocFile



## otrLogger und otrAssert

- ✓ Erleichtern Fehlersuche und Fehlerbehandlung
- ✓ Fester Bestandteil von DOCUMENTS
- ✓ Typings und Dokumentation

# VS Code

## Inhalt

- Multiselect Upload
- XML-Export / XML-Import
- Intellisense
- Intellisense und require()

# Neues im VS Code

## Script-Upload: Multi-Selection

- Upload Script: mehrere Scripte im Explorer auswählen
- Upload Scripts from Folder: mehrere Ordner im Explorer auswählen

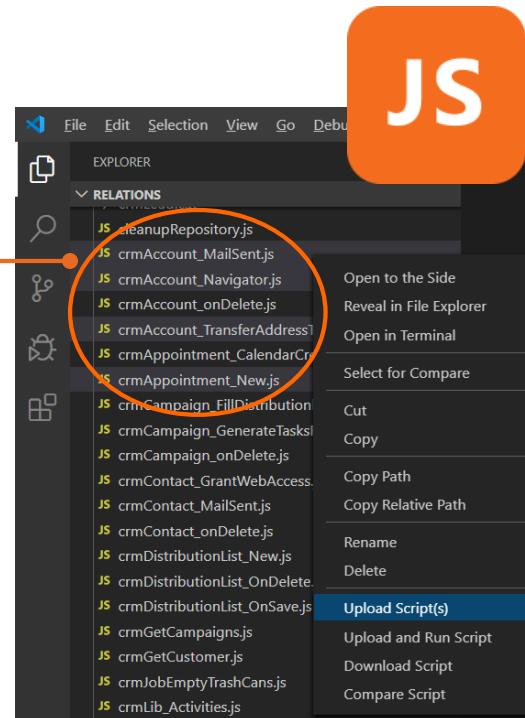
## Scriptlibs

- Bearbeitung von Scripten in ScriptLibs -> kein autom. clear cache
- F1 → Clear PortalScript Cache

## Wartungsoperationen

- Perform Maintenance Operation...

## XML Export / Import... (ff)



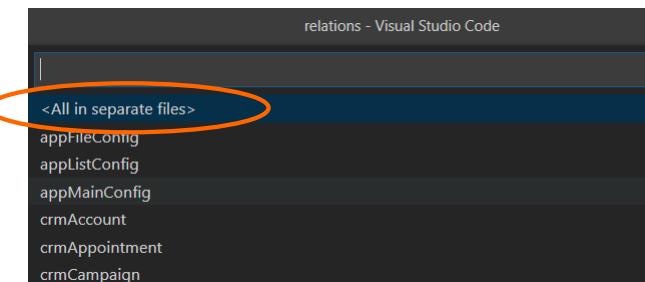
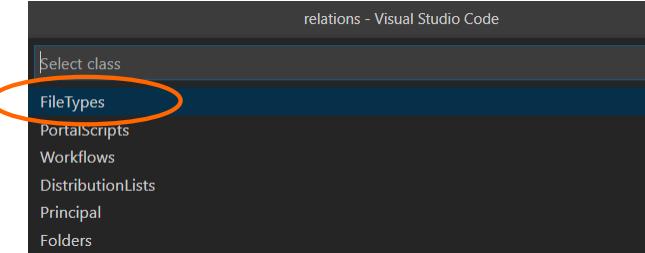
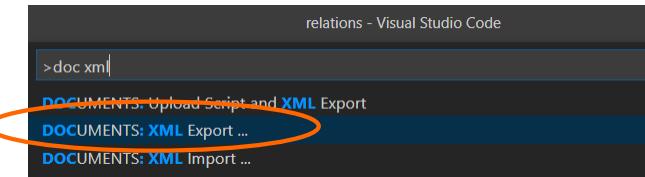
# XML Export FileTypes

## Wizard (Beispiel: Alle FileTypes)

- F1 → XML Export (type: doc xml)
- Select class: FileTypes
- Auswahlliste: <All in separate files>
- Output Pfad setzen: Default lassen

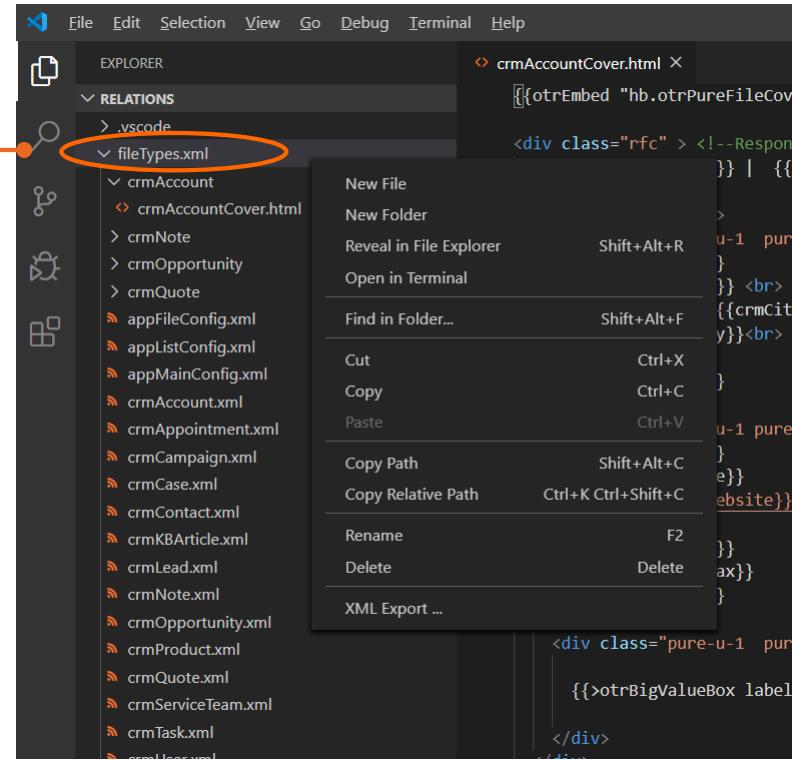
## XML Output

- Ordner mit Endung .xml → Kontextmenü... (ff)



# XML Export / Import

- Kontextmenü:
  - Ordner mit Endung .xml → XML Export
  - Datei mit Endung .xml → XML Import
- Motivation
  - Dokumentvorlagen im VS Code bearbeiten
  - Mappentyp und Dokumente in Git versionieren



# Struktur der Dokumentvorlagen

- Ordner-Name = FileType-Name (z.B. crmAccount)
  - (Anders als XML Export im Manager)
- Dokument-Name (crmAccountCover.html) wird beim XML-Import nicht geändert
  - (Anders als XML Import im Manager)

Empfehlung:  
Versionierung in Git  
(einfach über VS Code)

The screenshot shows the Visual Studio Code interface with the following details:

- EXPLORER View:** Shows a tree structure of files. The folder "RELATIONS" is expanded, containing ".vscode" and "fileTypes.xml". Inside "fileTypes.xml", the "crmAccount" entry is highlighted with an orange oval. Below it, "crmAccountCover.html" is also highlighted with an orange oval.
- Editor View:** Displays two files side-by-side.
  - crmAccount.xml:** An XML configuration file for the "DlcDocumentTemplate" class. It includes attributes like "label", "key", "keyscope", and "TemplateType". It also defines relations and object properties.
  - crmAccountCover.html:** A template file for the "DlcDocumentTemplate" class, defining attributes such as "Document", "DefaultDocumentName", "DefaultDocumentExtension", "AddOnFileCreation", "ParseTemplate", "ChangeNameAfterCreation", and "AutoOpen".

# XML Export PortalScripte

## Wizard (Beispiel: JSON Script Liste)

- JSON Datei im Editor: Liste mit Script-Namen
- F1 → XML Export
- Select Class: PortalScripts
- Auswahlliste: <Get names from JSON>
- JSON Datei: Default lassen
- Output Pfad setzen: Default lassen
- XML Output:
  - Alle Scripte aus der JSON Liste in einer XML Datei

The screenshot shows the Visual Studio Code interface with several windows and toolbars:

- Top Bar:** Shows the menu (File, Edit, Selection, View, Go, Debug, Terminal, Help) and the current file name "portalScripts.json - relations".
- Toolbar:** Includes icons for Selection, View, Go, Debug, Terminal, Help, and a search bar containing "> doc xml". Below the toolbar are three buttons: "DOCUMENTS: Upload Script and XML Export", "DOCUMENTS: XML Export ...", and "DOCUMENTS: XML Import ...". The "DOCUMENTS: XML Export ..." button is circled in orange.
- Editor Window:** Displays the JSON content of "portalScripts.json":

```
{
  "myScriptList": [
    "crmAccount_MailSent",
    "crmAccount_Navigator",
    "crmAccount_onDelete",
    "crmAccount_TransferAddressToContacts"
  ]
}
```
- relations - Visual Studio Code:** A dropdown menu titled "Select class" with "PortalScripts" selected (circled in orange). Other options include "FileTypes" and "Workflows".
- relations - Visual Studio Code:** A dropdown menu titled "File Types" with "<Get names from JSON>" selected (circled in orange). Other options include "<All in separate files>" and "<All in one file>".
- portalScripts.json - relations - Visual Studio Code:** A file selection dialog showing the path "c:\projekte\relations\portalScripts.json" (circled in orange).
- portalScripts.json - relations - Visual Studio Code:** A file selection dialog showing the path "c:\projekte\relations\portalScripts.xml" (circled in orange). Below it is the instruction "Enter folder (use .xml in folder name) (Press 'Enter' to confirm or 'Escape' to cancel)".

# Utilityfunktion für PortalScripte

- Script im Editor geöffnet → F1 → Upload and Export XML

The diagram illustrates a workflow for exporting a PortalScript script. It starts with a screenshot of a code editor showing a script named `crmAccount_onDelete.js`. A red box highlights the menu bar "Debug Terminal Help". A red arrow points from the "Help" menu to a second screenshot showing the results of the "Upload and Export XML" command. In the second screenshot, a red box highlights the status bar message "DOCUMENTS: Upload Script and XML Export". Another red arrow points from this message to a third screenshot showing the resulting XML file `crmAccount_onDelete.xml`, which is also highlighted by a red box.

Debug Terminal Help      crmAccount\_onDelete.js - relations - Visual S

JS crmAccount\_onDelete.js X      >doc up xml

// Delete referring connec  
// Albrecht 12.01.06

var docFile = context.file;

if (docFile)

{

  var accountId = docFile.crmId;

  // =====

  // DELETE CONNECTED FILES

  // =====

  // CONTACTS

  var docType = "crmContact";

  var filter = "crmCompany|~'" + accountId + "'";

  var sortOrder = "";

  // =====

  // DELETED

}

DOCUMENTS: Upload Script and XML Export

JS crmAccount\_onDelete.js      crmAccount\_onDelete.xml X

<?xml version="1.0" encoding="UTF-8"?>

<janus language="uk">

  <object class="PortalScript" label="259:96297" key="Name">

    <attribute name="Name">crmAccount\_onDelete</attribute>

    <attribute name="FromScriptlib"/>

    <attribute name="SourceCode"><![CDATA[// Delete referring connecte  
// Albrecht 12.01.06 14.10

  var docFile = context.file;

  if (docFile)

  {

    var accountId = docFile.crmId;

    // =====

    // DELETE CONNECTED FILES

    // =====

    // CONTACTS

    var docType = "crmContact";

    var filter = "crmCompany|~'" + accountId + "'";

    var sortOrder = "";

    // =====

    // DELETED

  }

# IntelliSense und JSDoc (Wdh.)

- Typ einer Variable/Funktion in JSDoc Kommentar angeben
  - → IntelliSense an der Variable/Parameter für den angegebenen Typ
- **Tipp:** über einer Funktion `/** Enter` eingeben → alle `@param` automatisch

Typ durch  
Definition klar



```
var myVar = "DOCUMENTS 5";
myVar.
```

IntelliSense suggestion list:

- localeCompare
- match
- normalize
- repeat
- replace
- search
- slice
- small
- split
- startsWith
- strike

Falls Return-Type der  
Funktion nicht klar  
→ Typ mit JSDoc angeben



```
/** @type {string} */
var myVar = anyModule.getString();
myVar.
```

IntelliSense suggestion list:

- localeCompare
- match
- normalize
- repeat
- replace
- search
- slice
- small
- split
- startsWith
- strike

Typen für  
Parameter mit  
JSDoc angeben

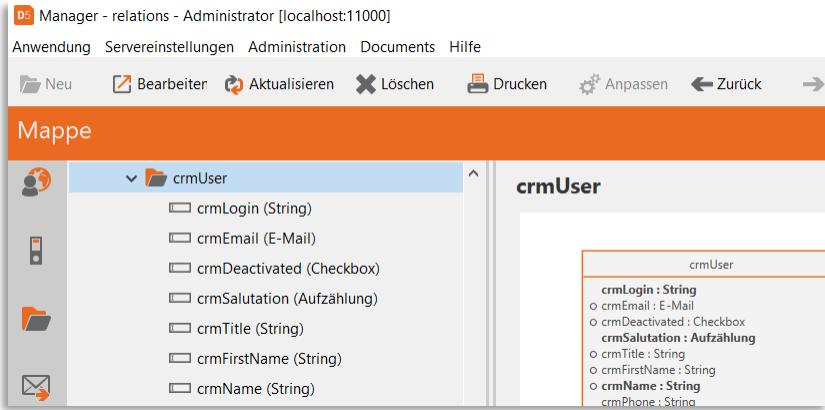


```
/**
 * @param {string} param
 * @returns {string}
 */
function myFunction(param) {
  param.
```

IntelliSense suggestion list:

- localeCompare
- match
- normalize
- repeat
- replace
- search
- slice
- small

# IntelliSense für Mappentypen (Wdh.)



context.createFile  
→ Typ `crmUser`

```
var myDocFile1 = context.createFile("crmUser");
myDocFile1....
  ↗ crmDeactivated
  ↗ crmEmail
  ↗ crmFax
  ↗ crmFirstName
  ↗ crmLogin
  ↗ crmMobilePhone
  ↗ crmName
```

context.file  
→ Typ `DocFile`

```
var myDocFile2 = context.file;
myDocFile2....
  ↗ abort
  ↗ addDocumentFromFileSystem
  ↗ addPDF
  ↗ archive
  ↗ archiveAndDelete
  ↗ asJSON
  ↗ cancelWorkflow
```

context.file mit JSDoc  
→ Typ `crmUser`

```
/** @type {crmUser} */
var myDocFile2 = context.file;
myDocFile2....
  ↗ crmDeactivated
  ↗ crmEmail
  ↗ crmFax
  ↗ crmFirstName
  ↗ crmLogin
  ↗ crmMobilePhone
```

`crmUser.crmName`  
→ Typ `String`

```
/** @type {crmUser} */
var myDocFile2 = context.file;
var myName = myDocFile2.crmName;
myName....
  ↗ charAt
  ↗ charCodeAt
  ↗ codePointAt
  ↗ concat
  ↗ endsWith
```

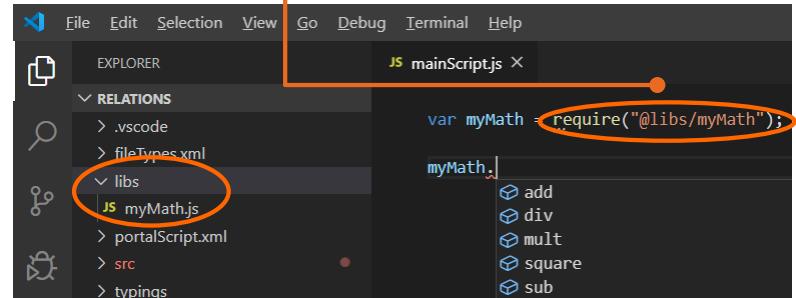
# Tipps für VS Code

## IntelliSense und require (Beispiel):

- myMath.js aus scriptlibs wird im PortalScript verwendet
- myMath.js liegt im Projekt in ./libs/
- IntelliSense für myMath im PortalScript:
  - In jsconfig.json → compilerOptions: Alias für Pfad zu libs setzen
  - Im PortalScript: Alias in require verwenden
- DOCUMENTS entfernt alle Präfixe aus require



```
{  
  "compilerOptions": [  
    "target": "es2016",  
    "lib": ["es2016", "scripthost"],  
    "baseUrl": "./src",  
    "paths": {  
      "@libs/*": ["./libs/*"]  
    }  
  ]  
}
```

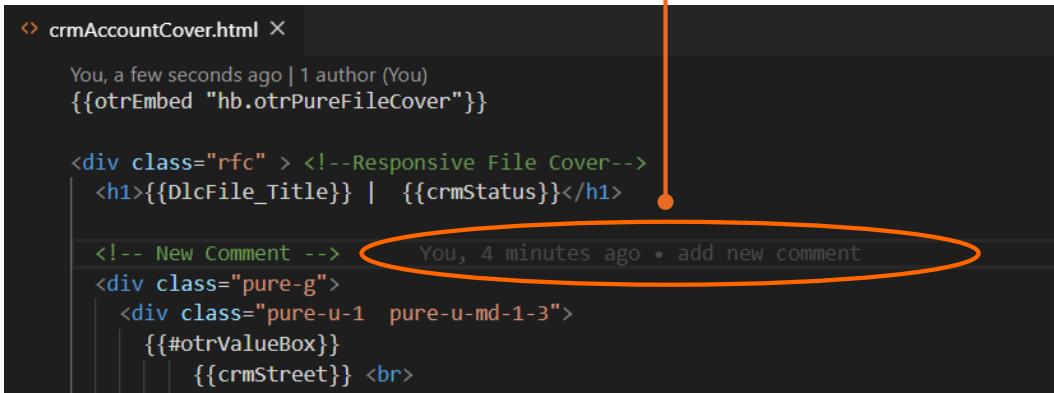


```
File Edit Selection View Go Debug Terminal Help  
EXPLORER  
RELATIONS  
> .vscode  
> fileTypes.xml  
> libs  
  > myMath.js  
> portalScript.xml  
> src  
> typings  
JS mainScript.js  
var myMath = require("@libs/myMath");  
myMath.
```

# Tipps für VS Code

## GitLens: VS Code Extension für Git

- Umfangreiche Informationen in eigener Side Bar
- Informationen der Git Commits im Editor anzeigen
  - Committer, Zeitstempel und Message
  - Wird für die markierte Zeile angezeigt



```
crmAccountCover.html X

You, a few seconds ago | 1 author (You)
{{otrEmbed "hb.otrPureFileCover"}}

<div class="rfc" > <!--Responsive File Cover-->
  <h1>{{DlcFile_Title}} | {{crmStatus}}</h1>

  <!-- New Comment --> You, 4 minutes ago • add new comment
  <div class="pure-g">
    <div class="pure-u-1 pure-u-md-1-3">
      {{#otrValueBox}}
        {{crmStreet}} <br>
```

