



docuplex Integration

docuplex wurde konsequent in Anlehnung an das Modell einer serviceorientierten Architektur (SOA) entworfen. Die gesamte Dienste-Architektur der norpa trägt in Ableitung den Namen „doa“ für dienstorientierte Architektur.

docuplex ist ein solcher, in Java implementierter, doa-Dienst. Jede Funktionalität von docuplex verfügt demnach über einen entsprechenden Methodenaufruf in diesem Dienst. Somit ist es möglich, jedwede Aktion, die ein Benutzer über ein Frontend vornehmen kann, auch automatisiert durch eine Fremdanwendung durchzuführen. Bei vollständiger Integration aller Methoden in eine Dritt-Anwendung besteht keine zwingende Notwendigkeit, dem Benutzer eine andere, fremde Anwendung zur Verfügung zu stellen.

Funktionsprinzip

Unter einem URL (z.B.: <https://localhost:8080/docuplex/ServiceController?perform>) kann ein Client eine Anfrage an docuplex stellen. Die Anfrage und auch die Antwort folgen einer festgelegten XML-Struktur, dem Request- bzw. Response-Format.

Beide Formate untergliedern sich in einen Head- und einen Body-Bereich. Dabei führt der Head-Bereich hauptsächlich solche Informationen, die für die Ansteuerung der Methoden eines Dienstes von Bedeutung sind. Die eigentlichen Nutzdaten befinden sich im jeweiligen Body.

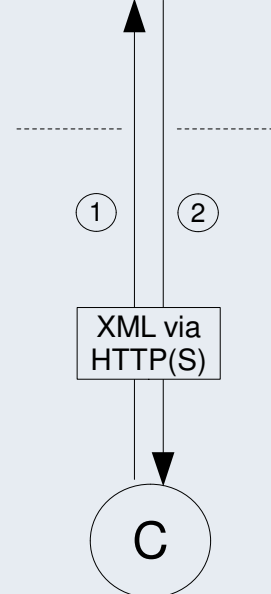
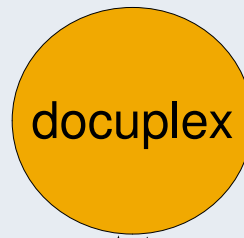
Bei einer Anfrage enthält der Body immer einen `params`-Knoten mit den Daten für den Dienst. Die Antwort signalisiert im Head-Bereich über den Knoten `responseCode` ein positives oder negatives Ausführungsergebnis.

Bei erfolgreicher Ausführung enthält der Response-Body einen `result`-Knoten mit den zurückgelieferten Daten. Bei fehlerhafter Ausführung ist stattdessen ein `error`-Knoten mit Fehlerinformationen vorhanden.

Das Schaubild auf der nächsten Seite soll dieses Prinzip verdeutlichen.

docuplex Request

```
<?xml version="1.0" encoding="windows-1252"?>
<serviceRequest>
  <requestHead>
    <module>docuplex</module>
    <service>FileHandling</service>
    <method>browseFolder</method>
    <clientID>dsch-laptop</clientID>
    <requester>DSCH</requester>
    <password>202cb962ac64b034b70</password>
    <requestTimestamp>08:41:24</requestTimestamp>
  </requestHead>
  <requestBody>
    <params>
      <file client="bau gmbh"
repository="Buchhaltung"
name="/Umsatzsteuer"/>
    </params>
  </requestBody>
</serviceRequest>
```



```
<?xml version="1.0" encoding="windows-1252"?>
<serviceResponse>
  <responseHead>
    <responseCode>200</responseCode>
    <responseText>OK</responseText>
  </responseHead>
  <responseTimestamp>16:04:01</responseTimestamp>
  <responseBody>
    <result>
      <rowset>
        <row id="1">
          <file client="bau gmbh"...>
            <file name="/Umsatzsteuer/2008.pdf".../>
          </file>
        </row>
      </rowset>
    </result>
  </responseBody>
</serviceResponse>
```

1. Client stellt Anfrage an docuplex
2. docuplex liefert Daten an Client aus

Die Timestamps und Parameter sind aus Platzgründen gekürzt.
Ein Timestamp hat immer die Form yyyy-mm-dd hh:mm:ss
Die vollständigen Parameter sind der Service-Dokumentation zu entnehmen.





Beispiel einer Dienstanfrage

In dem vorstehenden Beispiel fordert ein Client den docuplex-Dienst auf, den Inhalt eines bestimmten Verzeichnisses zu übermitteln. Dabei werden als Parameter die Namen des Mandanten, der Ablage und des Verzeichnisses mitgegeben.

Die Anfrage des Clients (Request-Struktur) ist auf der linken Seite dargestellt.

Die rechte Seite (Response-Struktur) zeigt die Antwort des Dienstes bei positiver Ausführung. Darin liefert der Dienst die Dateien zurück.

Dienst-Antwort bei Fehlern

Eine Antwort im Fehlerfall hat immer die nachfolgende Struktur.

Im Beispiel könnte es sein, dass das Verzeichnis nicht vorhanden ist.

Zunächst wird durch den Response-Code 500 angezeigt, dass ein Fehler vorliegt. Die Details zu diesem Fehler sind im `error`-Knoten enthalten. Dieser beinhaltet immer die Knoten `code` und `text`.

Die Fehlercodes sind seitens des Dienstes spezifiziert. Auch der Fehlertext wird vom Dienst geliefert.

```
<?xml version="1.0" encoding="WINDOWS-1252" ?>
<serviceResponse>
  <responseHead>
    <responseCode>500</responseCode>
    <responseText>error</responseText>
    <responseTimestamp>2009-08-18 08:12:14</responseTimestamp>
  </responseHead>
  <responseBody>
    <error>
      <code>-2</code>
      <text>directory does not exist</text>
    </error>
  </responseBody>
</serviceResponse>
```

Auf den folgenden Seiten werden einige der Schnittstellen des docuplex-Dienstes exemplarisch vorgestellt.

Modul	docuplex
Service	Info
Methode	browseRepositories
Bereich	Repository
benötigte Rechte	Jeder in diesem Mandanten vorhandene Benutzer
Funktion	Liefert eine Liste der Ablagen, in denen der anfragende Benutzer mindestens ein Recht besitzt
Bemerkung	
siehe auch	

Aufruf		
<pre><params> <client name="bau gmbh"/> </params></pre>		
Attribut	Pflicht	Beschreibung
name	x	Name des Mandanten

Rückgabe	
<pre><result> <rowset> <row id="1"> <client name="bau gmbh"> <repositories> <repository name="Buchhaltung" accessmode="readwrite" versioning="true"/> <repository name="Sonstiges" accessmode="createreadonly"/> <repository name="unsere Preise" accessmode="readonly" versioning="true"/> </repositories> </client> </row> </rowset> </result></pre>	
Attribut	Beschreibung
name	Name der Ablage
accessmode	Zugriffsmodus der Ablage
versioning	Wenn True, dann verwaltet die Ablage Datei-Versionen

Modul	docuplex
Service	FileHandling
Methode	browseFolder
Bereich	Folder
Funktion	Liefert eine Liste von Dateien und Verzeichnissen
benötigte Rechte	browseFolders
Bemerkung	
siehe auch	searchFolder

Aufruf

```
<params>
  <file client="bau gmbh" repository="Buchhaltung" name="/Finanzamt"
        folderonly="false" filefilter="*" includetrash="true"
        detectsubfolder="true" recursive="true" withattributes="true"/>
</params>
```

Attribut	Pflicht	Beschreibung
client	x	Name des Mandanten
repository	x	Name der Ablage
name	x	Name des Verzeichnisses
folderonly		True, wenn nur Verzeichnisse zurückgegeben werden sollen
filefilter		Nur Objekte zurückliefern, die dieser Namensmaske entsprechen
includetrash		True, wenn auch der Mülleimer zurückgegeben werden soll
detectsubfolder		True, wenn geprüft werden soll, ob ein Verzeichnis auch Unterverzeichnisse besitzt
recursive		True, wenn auch die Dateien in den Unterverzeichnissen ausgegeben werden sollen
withattributes		True, wenn Dateiattribute mit ausgegeben werden sollen

Rückgabe

```
<result>
  <rowset>
    <row id="1">
      <file client="bau gmbh" repository="Buchhaltung" name="/Finanzamt"
            filefilter="*" includetrash="true" detectsubfolder="true"
            recursive="true" withattributes="true">
        <files>
          <file name="/Finanzamt/2008" isfolder="true" hassubfolder="true"
                lastmodified="2010-08-13 14:19:58"/>
          <file name="/Finanzamt/Finanzamt_Auskunftersuchen.pdf"
                size="35587" lastmodified="2011-01-11 13:08:26" priority="0"
                reminder="2012-03-01" tags="FINANZAMT AUSKUNFT STEUER"/>
          <file name="/Finanzamt/KSt_GewSt_Vz_2008_07.pdf"
                size="31053" lastmodified="2011-01-11 13:08:26" priority="1"
                reminder="2012-09-01" tags="FINANZAMT STEUER"/>
        </files>
      </file>
    </row>
  </rowset>
</result>
```

Attribut	Beschreibung
name	Name der Datei oder des Verzeichnisses
isfolder	Objekt ist ein Verzeichnis
hassubfolder	True, wenn das Verzeichnis Unterverzeichnisse besitzt
lastmodified	Datum der letzten Bearbeitung der Datei
hasannotation	True, wenn die Datei oder das Verzeichnis Anmerkungen enthält
size	Größe der Datei in Byte
priority	Priorität als Ziffer (mgl. Werte:-1,0,1;default=0)
reminder	Wiedervorlagedatum im Format „JJJ-MM-TT“
tags	Liste von Schlagwörtern, leerzeichensepariert

Modul	docuplex
Service	FileHandling
Methode	createFile
Bereich	File
Funktion	Legt eine Datei blockweise an
benötigte Rechte	createFiles (+ setAnnotations, wenn Anmerkung mitgegeben wird; + deleteFile, wenn replaceexisting = true ist)
Bemerkung	Die Größe der Blocks beträgt immer 1MB (außer letzter Block < 1MB)
siehe auch	

Aufruf (für Datei)
<pre><params> <file client="bau gmbh" repository="Buchhaltung" name="/Finanzamt/Erklärung.doc" annotation="VGvzdCBieSBKQUQ" ready="true" size="13464645" session="dtz3t9z34tzd4c" lastmodified="2011-05-11 13:08:26" replaceexisting="true"> 0M8R4KGxGuEAAAAAAAAAAAAAAAAAAAAAAAAAPgADAP7... </file> </params></pre>

Aufruf (für Verzeichnis)
<pre><params> <file client="bau gmbh" repository="Buchhaltung" name="/Finanzamt/Ordner1/" annotation="VGvzdCBieSBKQUQ" lastmodified="2011-05-11 13:18:20" isfolder="true"> </file> </params></pre>

Attribut	Pflicht	Beschreibung
client	x	Name des Mandanten
repository	x	Name der Ablage
name	x	Name der Datei oder des Verzeichnisses
annotation		Inhalt der Anmerkung (Base64-kodiert)
ready		True, wenn der aktuelle Request der letzte für diese Datei ist
size	x	Größe der gesamten Datei in Byte
session		Eindeutiger Schlüssel, welcher durch das anfragende Programm erzeugt wird
lastmodified		Änderungsdatum
replaceexisting		True, wenn eine vorhandene Datei überschrieben werden soll
isfolder		True, wenn ein Verzeichnis angelegt werden soll

Knoten	Pflicht	Beschreibung
Datei		Datei als Base64-kodierter Text

Rückgabe
keine

Modul	docuplex
Service	FileHandling
Methode	deleteFile
Bereich	File
Funktion	Löschen von Dateien und Verzeichnissen (diese rekursiv)
benötigte Rechte	deleteFiles
Bemerkung	
siehe auch	emptyTrash

Aufruf

```
<params>
  <file client="bau gmbh" repository="Buchhaltung" name="/Finanzamt/Erklärung.doc"
    ignoreexistence="true" notrash="true"/>
</params>
```

Attribut	Pflicht	Beschreibung
client	x	Name des Mandanten
repository	x	Name der Ablage
name	x	Name der Datei
ignoreexistence		True, wenn das Löschen einer nicht existierenden Datei keinen Fehler werfen soll
notrash		True, wenn das Löschen nicht über den Mülleimer erfolgen soll

Rückgabe

```
keine
```