

Tempus Serva REST interface

Getting started

Enabling the interface

The REST service interface is by default disabled.

First allow the server to provide REST support

- “Modules” > “Configuration”: Set the system policy “EnableRestInterface” to “true”

Second enable REST access for each solution

- “Solution” > [solution] > “Advanced”: Check “Enable REST interface”

Setting up system user acces

The REST interface needs a system user to access the system. The user is treated no differently than normal users in regard to permissions, validation etc.

- Create an new user and assign permission groups
- Add access to the solutions in question

Producing the XSD for the connecting code

The URL for the REST schemas can be retrieved at this URL

- [ServerName]/[ApplicationName]/rest/[SolutionSystemName].wadl
- [ServerName]/[ApplicationName]/rest/[SolutionSystemName].xsd

The interface requires a login to retrieve, so before accessing the URL you login using normal authentication methods.

Note: The schema returned is a reflection of the user accessing the system. If a certain field is missing or you get no results, please check the current user permissions.

Understanding the REST interface

The URL structure is as follows

- /rest
 - /[Solution System Name]
 - .wadl
 - .xsd
 - /[DataID]+[]+[Data resume]
 - /[Solution System Name]
 - ...
 - ...

Example	Path
CRM model Schema	/TempusServa/rest/crmsystem.xsd
CRM service Schema	/TempusServa/rest/crmsystem.wadl
List of all CRM items	/TempusServa/rest/crmsystem
Specific CRM item	/TempusServa/rest/crmsystem/1234 Hello world company
Files for the Hello World company	/TempusServa/rest/crmsystem/1234 Hello world company/files

3 methods are supported

- GET:
 - For retrieving list of records
 - For retrieving a specific record
- PUT
 - For adding new records
- POST:
 - Used for updating records

Encoding

- All requests should be encoded as per REST specification
- HTTP encoded whitespaces (%20) are decoded server side

Parameters for all calls

Content	Name (default)	Type
User login name	login	Required if new session
User password	password	Required if new session

Parameters for list calls

Method	Path	Data returned
GET	/rest/[Solution System Name]	[solution]+ListItem

Content	Name	Type
List of fields to include in list	QUERY_FIELDS	Optional
Filter parameters	QUERY_PARAM_#	Optional
	QUERY_OPERATOR_#	Optional
	QUERY_VALUE_#	Optional
Reset filter and field selection	QUERY_NEW	Optional

is an integer

Parameters for item insert

Method	Path	Data returned
POST	/rest/[Solution System Name]	[solution]+Item

Content	Name	Type
Set field value	[FIELDNAME]	Optional

Parameters for item select

Method	Path	Data returned
GET	/rest/[Solution System Name]/[Data ID] ...	[solution]+Item

... = Extra descriptor information, not used or validated in any way

Content	Name	Type

Parameters for item update

Method	Path	Data returned
POST	/rest/[Solution System Name]/[Data ID] ...	[solution]+Item

Content	Name	Type
Set value	[FIELDNAME]	Optional

Handling connected resources

Related data/records

Related data are normally handled just by setting a unique String reference (aka the record Resume).

If needed DataID references can be used, for faster and safer operation.

Attached resources

Downloadable resources are available as per normal users

- Files
- Pictures
- Contacts

Theses ressources can be retrieved using normal HTTP GET requests, by accessing the URL's included in the XML structure for these fields.

Help for developers

Sample XSD schema file

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://localhost/TempusServa/rest/firmabiler/xsd"
xmlns="http://localhost/TempusServa/rest/firmabiler/xsd" elementFormDefault="qualified">
  <xs:element name="firmabilerList">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="firmabilerListItem" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="firmabilerListItem">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="DataID" type="xs:string" minOccurs="1" maxOccurs="1"/>
        <xs:element name="NUMMERPLADE" type="xs:string" minOccurs="0" maxOccurs="1"/>
        <xs:element name="MODEL" type="xs:string" minOccurs="0" maxOccurs="1"/>
        <xs:element name="FARVE" type="xs:string" minOccurs="0" maxOccurs="1"/>
        <xs:element name="SKADET" type="xs:string" minOccurs="0" maxOccurs="1"/>
        <xs:element name="StatusID" type="xs:string" minOccurs="0" maxOccurs="1"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="firmabilerItem">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="DataID" type="xs:string" minOccurs="1" maxOccurs="1"/>
        <xs:element name="NUMMERPLADE" type="xs:string" minOccurs="0" maxOccurs="1"/>
        <xs:element name="MODEL" type="xs:string" minOccurs="0" maxOccurs="1"/>
        <xs:element name="KLASSE" type="xs:string" minOccurs="0" maxOccurs="1"/>
        <xs:element name="FARVE" type="xs:string" minOccurs="0" maxOccurs="1"/>
        <xs:element name="SKADET" type="xs:string" minOccurs="0" maxOccurs="1"/>
        <xs:element name="SKADEBESKRIVELSE" type="xs:string" minOccurs="0" maxOccurs="1"/>
        <xs:element name="EFTERSET" type="xs:string" minOccurs="0" maxOccurs="1"/>
        <xs:element name="AFSTEMNINGER" type="xs:string" minOccurs="0" maxOccurs="1"/>
        <xs:element name="BILLEDE2" type="xs:string" minOccurs="0" maxOccurs="1"/>
        <xs:element name="StatusID" type="xs:string" minOccurs="0" maxOccurs="1"/>
        <xs:element name="Revision" type="xs:string" minOccurs="0" maxOccurs="1"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Sample Java REST Client connector

```
public class Firmabiler {  
    public static void main(String[] args) {  
        FirmabilerClient session = new FirmabilerClient();  
        FirmabilerList result =  
            session.getList(FirmabilerList.class, "kasper", "secret", "TITEL=Kasper" );  
        List <FirmabilerListItem> list = result.getFirmabilerListItem();  
        for(int i=0; i<list.size(); i++) {  
            FirmabilerListItem item = list.get(i);  
            System.out.println( item.getDataID() + "\t" + item.getNUMMERPLADE() );  
        }  
        session.close();  
    }  
}
```

Tips for working with the REST interface

- Test the REST interface through a browser
- Use a API generator to generate the method interfaces
 - Add Webservice to ide (wadl import)
 - Add REST Client to project (use above)
 - Add JAXB bindings to project (use XSD schema)
- Update the underlying XSD if you have changes to ...
 - Permission structure
 - Field validation
 - State model

Known problems

The following problems have been encountered using the interface

Problem	Workaround/Reason
Readlocked fields are not updated and the values are ignored	This is behavior by design. Current definitions do not provide a way of describing dependencies
The Schema changes when I add new fields or change permissions.	This is behavior by design.
There is no way to attach files	This will be addressed Q4 2014
No API support for contacts	You will need to parse the information yourself. This will be addressed Q4 2014.