View Management Web Services

Update Views

This section covers a set of web services to update a Yellowfin view. These work in conjunction with one another and must be used in the following sequence:

1. Use the **EDITVIEW** web service to put a specified view in draft mode.
2. Then using the **ADDCOLUMNTOVIEW** function, update your view by adding database columns to it.
3. To save these changes, call the **PUBLISHVIEW** web service.

This web service sets the specified view into “draft” mode so that it can be edited by other web service calls.

Request Parameters

The following parameters should be passed with this request:

Request Element	Data Type	Description
LoginId	String	An admin account to connect to Yellowfin web services. This can be the user ID or the email address, depending on the Logon ID method. This account must have the “web services” role enabled, and must belong to the default (i.e. primary) org
Password	String	Password of the above account.
OrgId	Integer	Default (i.e. primary) organization ID within Yellowfin. Always set this to 1.
OrgRef	String	Optional. Client organization the view can be found in.
Function	String	Web service function. Set this to "EDITVIEW".
ViewId	Integer	Internal ID of the view to be edited.

Request Example

Below is a SOAP JAX example for this request:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:web="http://webservices.web.
mi.hof.com/">
   <soapenv:Header/>
   <soapenv:Body>
      <web:remoteAdministrationCall>
         <arg0>
          <loginId>admin@yellowfin.com.au</loginId>
            <password>test</password>
            <orgId>1</orgId>
            <function>EDITVIEW</function>
            <viewId>60543</viewId>
         </arg0>
      </web:remoteAdministrationCall>
   </soapenv:Body>
</soapenv:Envelope>
```

Response Parameters

The returned response will contain these parameters:

Response Element	Data Type	Description
StatusCode	String	Status of the web service call. Possible values include: • SUCCESS • FAILURE

Response Example

The service will return the below response, according to our SOAP example:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
   <S:Body>
      <ns2:remoteAdministrationCallResponse xmlns:ns2="http://webservices.web.mi.hof.com/">
         <return>
            <errorCode>0</errorCode>
            <messages>Successfully Authenticated User: admin@yellowfin.com.au</messages>
            <messages>Web Service Request Complete</messages>
            <sessionId>d256811ce7cdfc856baae8dd00737e88</sessionId>
            <statusCode>SUCCESS</statusCode>
         </return>
      </ns2:remoteAdministrationCallResponse>
   </S:Body>
</S:Envelope>
```

This web service call adds a new database column to the specified view, using the naming and category provided in an **AdminsitrationViewField** object.

Request Parameters

The following parameters should be passed with this request:

Request Element	Data Type	Description
LoginId	String	An admin account to connect to Yellowfin web services. This can be the user ID or the email address, depending on the Logon ID method. This account must have the “web services” role enabled, and must belong to the default (i.e. primary) org.
Password	String	Password of the above account.
OrgId	Integer	Default (i.e. primary) organization ID within Yellowfin. Always set this to 1.
OrgRef	String	Optional. Client organization the view can be found in.
Function	String	Web service function. Set this to "ADDCOLUMNTOVIEW".
ViewId	Integer	Internal ID of the view to be edited.
Parameters	String []	String array containing the database table and column name to be added. For example, the column “first_name” in the “person” table: {“person”,”first_name”}.
Field	**AdministrationViewField**	Object containing the new column definitions. See table below.

The following are required parameters to the **AdministrationViewField** object.

Object Element	Data Type	Description
ShortDescription	String	Primary name of the column to be added as seen in the view builder.
LongDescription	String	Description of the column to be added.
FieldCategory	String	Category the newly created field will be placed under. For e.g. “DIMENSION” or “METRIC”.

Request Example

Below is a SOAP JAX example for this request:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:web="http://webservices.web.
mi.hof.com/">
   <soapenv:Header/>
   <soapenv:Body>
      <web:remoteAdministrationCall>
         <arg0>
          <loginId>admin@yellowfin.com.au</loginId>
            <password>test</password>
            <orgId>1</orgId>
            <function>ADDCOLUMNTOVIEW</function>
            <viewId>60543</viewId>
            <parameters>person</parameters>
            <parameters>first_name</parameters>
            <field>
                    <shortDescription>Person Name</shortDescription>
                    <longDescription>This is the name this person wants to be called</longDescription>
                    <fieldCategory>People</fieldCategory>
            </field>
         </arg0>
      </web:remoteAdministrationCall>
   </soapenv:Body>
</soapenv:Envelope>
```

Response Parameters

The returned response will contain these parameters:

Response Element	Data Type	Description
StatusCode	String	Status of the web service call. Possible values include: • SUCCESS • FAILURE

Response Example

The service will return the below response, according to our SOAP example:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:web="http://webservices.web.
mi.hof.com/">
   <soapenv:Header/>
   <soapenv:Body>
      <web:remoteAdministrationCall>
         <arg0>
          <loginId>admin@yellowfin.com.au</loginId>
            <password>test</password>
            <orgId>1</orgId>
            <function>ADDCOLUMNTOVIEW</function>
            <viewId>100937</viewId>
            <parameters xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001
/XMLSchema-instance" xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/" SOAP-ENC:arrayType="xsd:string
[1]" xsi:type="SOAP-ENC:Array">
                        <xsd:string>CAMP</xsd:string>
                        <xsd:string>CAMPID</xsd:string>
            </parameters>
            <field>
                <shortDescription>CAMPID</shortDescription>
                <longDescription>Camp identification number.</longDescription>
                <fieldCategory>ID Folder</fieldCategory>
            </field>
         </arg0>
      </web:remoteAdministrationCall>
   </soapenv:Body>
</soapenv:Envelope>
```

This web service saves and publishes a view to use reports. If this function is not called after EDITVIEW, the view will remain un-editable until the web service session times out.

Request Parameters

The following parameters should be passed with this request:

Request Element	Data Type	Description
LoginId	String	An admin account to connect to Yellowfin web services. This can be the user ID or the email address, depending on the Logon ID method. This account must have the “web services” role enabled, and must belong to the default (i.e. primary) org.
Password	String	Password of the above account.
OrgId	Integer	Default (i.e. primary) organization ID within Yellowfin. Always set this to 1.
OrgRef	String	Optional. Client organization the view can be found in.
Function	String	Web service function. Set this to " PUBLISHVIEW".
ViewId	Integer	Internal ID of the view to be published.

Request Example

Below is a SOAP JAX example for this request:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:web="http://webservices.web.
mi.hof.com/">
   <soapenv:Header/>
   <soapenv:Body>
      <web:remoteAdministrationCall>
         <arg0>
          <loginId>admin@yellowfin.com.au</loginId>
            <password>test</password>
            <orgId>1</orgId>
            <function>PUBLISHVIEW</function>
            <viewId>60543</viewId>
         </arg0>
      </web:remoteAdministrationCall>
   </soapenv:Body>
</soapenv:Envelope>
```

Response Parameters

The returned response will contain these parameters:

Response Element	Data Type	Description
StatusCode	String	Status of the web service call. Possible values include: • SUCCESS • FAILURE

Response Example

The service will return the below response, according to our SOAP example:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
   <S:Body>
      <ns2:remoteAdministrationCallResponse xmlns:ns2="http://webservices.web.mi.hof.com/">
         <return>
            <errorCode>0</errorCode>
            <messages>Successfully Authenticated User: admin@yellowfin.com.au</messages>
            <messages>Web Service Request Complete</messages>
            <sessionId>50b03dd1649bbb123605aa801829095b</sessionId>
            <statusCode>SUCCESS</statusCode>
         </return>
      </ns2:remoteAdministrationCallResponse>
   </S:Body>
</S:Envelope>
```

This function allows the creation of a new folder in a view.

Request Parameters

The following parameters should be passed with this request:

Request Element	Data Type	Description
LoginId	String	An admin account to connect to Yellowfin web services. This can be the user ID or the email address, depending on the Logon ID method. This account must have the “web services” role enabled, and must belong to the default (i.e. primary) org.
Password	String	Password of the above account.
OrgId	Integer	Default (i.e. primary) organization ID within Yellowfin. Always set this to 1.
Function	String	Web service function. Set this to "ADDFOLDERTOVIEW".
ViewId	Integer	Internal ID of the view to be edited.
Parameters	String []	String array containing the database table and column name to be added. For example, the column “first_name” in the “person” table: {“person”,”first_name”}.

Request Example

Below is a SOAP JAX example for this request:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:web="http://webservices.web.
mi.hof.com/">
    <soapenv:Header/>
    <soapenv:Body>
        <web:remoteAdministrationCall>
            <arg0>
                <loginId>admin@yellowfin.com.au</loginId>
                <password>test</password>
                <orgId>1</orgId>
                <function>ADDFOLDERTOVIEW</function>
                <viewId>127187</viewId>
                <parameters xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001
/XMLSchema-instance" xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/" SOAP-ENC:arrayType="xsd:string
[1]" xsi:type="SOAP-ENC:Array">
                        <xsd:string>Hey New Folder</xsd:string>
                </parameters>
            </arg0>
        </web:remoteAdministrationCall>
    </soapenv:Body>
</soapenv:Envelope>
```

Response Parameters

The returned response will contain these parameters:

Response Element	Data Type	Description
StatusCode	String	Status of the web service call. Possible values include: • SUCCESS • FAILURE

Response Example

The service will return the below response, according to our SOAP example:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
   <S:Body>
      <ns2:remoteAdministrationCallResponse xmlns:ns2="http://webservices.web.mi.hof.com/">
         <return>
            <errorCode>0</errorCode>
            <messages>Successfully Authenticated User: admin@yellowfin.com.au</messages>
            <messages>Web Service Request Complete</messages>
            <sessionId>50b03dd1649bbb123605aa801829095b</sessionId>
            <statusCode>SUCCESS</statusCode>
         </return>
      </ns2:remoteAdministrationCallResponse>
   </S:Body>
</S:Envelope>
```

Also included is a JavaScript example demonstrating how each of these web services are used together.

Complete example of View web services

Below is a full example of the EDITVIEW, ADDCOLUMNTOVIEW, and PUBLISHVIEW web services. (Contained in a single file, as they are meant to be used together.) To use it for yourself, carry out the following the steps:

1. Copy the code and save it as ws_editview.jsp.
2. Put the file in the root folder, which is Yellowfin/appserver/webapps/ROOT.
3. Adjust host, port, admin user and user to add details according to your environment.
4. Run http://<host>:<port>/ws_editview.jsp from your Internet browser.

```
<%@ page language="java" contentType="text/html; charset=UTF-8" %>
<%@ page import="java.text.*" %>
<%@ page import="java.util.*" %>
<%@ page import="com.hof.mi.web.service.*" %>
<%@ page import="com.hof.mi.web.service.schedule.*" %>
<%@ page import="com.hof.data.*" %>
<%@ page import="com.hof.util.*" %>
<%@ page import="com.hof.web.form.*" %>
<html>
<body>
<%
String userId = "admin@yellowfin.com.au";
String password = "test";
String cliOrgRef = null;
%>
<%
   String baseuri = request.getRequestURI();
   int index = baseuri.lastIndexOf('/');
   if (index >= 0) baseuri = baseuri.substring(0, index);
   String self = request.getServletPath();
   self = self.substring(self.lastIndexOf('/') + 1);
   self = baseuri + "/" + self;
%>
<form action="<%=self%>" method="post">
        <input type="hidden" name="cmd" value="editView" />
        <h3>Edit View</h3>
        <input type="text" name="editViewId" />
        <input type="submit" />
</form>
<%
        AdministrationServiceResponse rs = null;
        AdministrationServiceRequest rsr = new AdministrationServiceRequest();
        AdministrationServiceService ts = new AdministrationServiceServiceLocator("localhost", 8080, "/services
/AdministrationService", false);
        AdministrationServiceSoapBindingStub rssbs = (AdministrationServiceSoapBindingStub) ts.
getAdministrationService();

        String cmd = request.getParameter("cmd");
        String viewID = "";

        /*
        * Place the view into DRAFT mode to enable web service editing
        */
        if ("editView".equals(cmd)) {
                viewID = request.getParameter("editViewId");
                rsr.setLoginId(userId);
                rsr.setPassword(password);
                rsr.setOrgId(new Integer(1));

                //Client organization the specified view can be found in
                rsr.setOrgRef(cliOrgRef);
                rsr.setFunction("EDITVIEW");
                if (viewID!=""){

                        //ViewId of the primary view entry

                        rsr.setViewId(new Integer(viewID));
                }
                rs = rssbs.remoteAdministrationCall(rsr);

                if ("SUCCESS".equals(rs.getStatusCode())) {
                        out.write("Success");
                } else {
                        out.write("Failure");
                }
        }
%>
<br><br>
<form action="<%=self%>" method="post">
        <input type="hidden" name="cmd" value="addView" />
```

```
        <h3>Add Column to View</h3>
        ViewID: <input type="text" name="addViewId" /><br><br>
        Database Table Name: <input type="text" name="addTableName" /> Column Name: <input type="text" name="
addColName" /><br><br>
        Field Name: <input type="text" name="addFieldName" /><br>
        Field Description: <input type="text" name="addFieldDesc" /><br>
        Field Category: <input type="text" name="addFieldCat" /><br>
        <input type="submit" />
</form>
<%
        cmd = request.getParameter("cmd");
        viewID = "";
        String dbTbl="";
        String dbCol="";
        String fieldName="";
        String fieldDesc="";
        String fieldCat="";
        if ("addView".equals(cmd)) {
                viewID = request.getParameter("addViewId");
                dbTbl = request.getParameter("addTableName");
                dbCol = request.getParameter("addColName");
                fieldName =  new String(request.getParameter("addFieldName").getBytes("iso-8859-1"),"UTF-8");
                fieldDesc = new String(request.getParameter("addFieldDesc").getBytes("iso-8859-1"),"UTF-8");
                fieldCat = new String(request.getParameter("addFieldCat").getBytes("iso-8859-1"), "UTF-8");
                rsr = new AdministrationServiceRequest();

                /*
                * Web service function to add a database column into a currently existing view.
                */

                rsr.setLoginId(userId);
                rsr.setPassword(password);
                rsr.setOrgId(new Integer(1));
                rsr.setOrgRef(cliOrgRef);
                rsr.setFunction("ADDCOLUMNTOVIEW");
                if (viewID!=""){
                        rsr.setViewId(new Integer(viewID));
                }
                AdministrationViewField field = new AdministrationViewField();
                field.setShortDescription(fieldName);
                field.setLongDescription(fieldDesc);
                field.setFieldCategory(fieldCat);
                rsr.setField(field);
                rsr.setParameters(new String[]{dbTbl, dbCol});
                rs = rssbs.remoteAdministrationCall(rsr);

                if ("SUCCESS".equals(rs.getStatusCode())) {
                        out.write("Success");
                } else {
                        out.write("Failure");
                }
        }
%>
<br><br>

<form action="<%=self%>" method="post">
        <input type="hidden" name="cmd" value="publishView" />
        <h3>Publish View</h3>
        <input type="text" name="publishViewId" />
        <input type="submit" />
</form>
<%
        cmd = request.getParameter("cmd");
        viewID = "";

        /*
        *After all changes have been made, publish the view.
        */

        if ("publishView".equals(cmd)) {
                viewID = request.getParameter("publishViewId");
```

```
                rsr = new AdministrationServiceRequest();
                rsr.setLoginId(userId);
                rsr.setPassword(password);
                rsr.setOrgId(new Integer(1));
                rsr.setOrgRef(cliOrgRef);
                rsr.setFunction("PUBLISHVIEW");
                if (viewID!=""){
                        rsr.setViewId(new Integer(viewID));
                }
                rs = rssbs.remoteAdministrationCall(rsr);

                if ("SUCCESS".equals(rs.getStatusCode())) {
                        out.write("Success");
                } else {
                        out.write("Failure");
                }
        }
%>
```

Other

This web service is used to delete a view in Yellowfin. You can specify the view by providing either its ID or UUID.

Request Parameters

The following parameters should be passed with this request:

Request Element	Data Type	Description
LoginId	String	An admin account to connect to Yellowfin web services. This can be the user ID or the email address, depending on the Logon ID method. This account must have the “web services” role enabled, and must belong to the default (i.e. primary) org.
Password	String	Password of the above account.
OrgId	Integer	Default (i.e. primary) organization ID within Yellowfin. Always set this to 1.
Function	String	Web service function. Set this to "DELETEVIEW".
Parameters	String[]	The ID or UUID of the view that is to be deleted.

Response Parameters

The returned response will contain these parameters:

Response Element	Data Type	Description
StatusCode	String	Status of the web service call. Possible values include: • SUCCESS • FAILURE

Instructions

See below for step-by-step instructions on how to perform this call, using a Java example:

- Define the request for this function, which includes logging in as the admin user and specifying the web service call to perform:

```
AdministrationServiceRequest rsr = new AdministrationServiceRequest();
rsr.setLoginId("admin@yellowfin.com.au");
rsr.setPassword("test");
rsr.setOrgId(new Integer(1));

rsr.setFunction("DELETEVIEW");
```

- Specify the view to be deleted by providing its ID or UUID:

```
rsr.setParameters(new String[] {
     "70103"
});
```

- Once the request is configured, perform the call:

```
AdministrationServiceResponse rs = adminService.remoteAdministrationCall(rsr);
```

  Initialize the Administration web service. Click here to learn how to do this.

- The response will contain the StatusCode. (See details in the Response Parameters table above.)

Complete Example

Below is a full example of this web service call. To use it for yourself, carry out the following the steps:

1. Copy the code and save it as ws_deleteview.jsp.
2. Put the file in the root folder: *Yellowfin/appserver/webapps/ROOT*.
3. Adjust the host, port, and admin user details according to your environment.
4. Run *http://<host>:<port>/ws_deleteview.jsp* from your Internet browser.

```
<%
/*                    ws_deleteview.jsp                       */
%>
<%@ page language="java" contentType="text/html; charset=UTF-8" %>
<%@ page import="com.hof.util.*, java.util.*, java.text.*" %>
<%@ page import="com.hof.web.form.*" %>
<%@ page import="com.hof.mi.web.service.*" %>

        AdministrationServiceResponse rs = null;
    AdministrationServiceRequest rsr = new AdministrationServiceRequest();
    AdministrationServiceService ts = new AdministrationServiceServiceLocator("localhost", 8080, "/services
/AdministrationService", false);
    AdministrationServiceSoapBindingStub rssbs = (AdministrationServiceSoapBindingStub) ts.
getAdministrationService();

    rsr.setLoginId("admin@yellowfin.com.au");
    rsr.setPassword("test");
    rsr.setOrgId(new Integer(1));
    rsr.setFunction("DELETEVIEW");

        //Specify the view to be deleted by providing its ID or UUID
    rsr.setParameters(new String[] {
            "70103"
    });

    rs = rssbs.remoteAdministrationCall(rsr);

    if ("SUCCESS".equals(rs.getStatusCode())) {
        out.write("Success </br>");
    } else {
        out.write(rs.getStatusCode());
        out.write(rs.toString());
    }
```