

Oracle SALT How To: expose Tuxedo Service as a SOAP Web Service



Introduction

This document is designed to help you implement a SALT use-case in a matter of minutes. This can be used to bootstrap projects from scratch, quickly add web services capabilities to an existing project, perform prototyping or evaluating of the product.

Note that this HOWTO document is built around an example service that can be easily replaced with any service, from TOUPPER to an existing application in your own environment.

Pre-requisites:

- Tuxedo and SALT 12.2.2 installed – Download Tuxedo from [here] - <http://www.oracle.com/technetwork/middleware/tuxedo/downloads/index.html>
- Tuxedo [RP004](#) and SALT [RP002](#)
- Configure Tuxedo domain with uBike server – Download the uBike.zip file from here: <http://www.oracle.com/technetwork/indexes/samplecode/tuxedo-sample-522120.html>
- Set environment variables to Tuxedo and domain (TUXDIR, APPDIR, SALTCONFIG)

Add GWWS, TMADMSVR and TMMETADATA to UBBCONFIG

Edit the UBB file and add the following lines to enable SALT if not done already:

```
TMMETADATA  SRVGRP=GROUP1  SRVID=2
             CLOPT="-A -- -f meta.repos"
TMADMSVR    SRVGRP=GROUP1  SRVID=3
             CLOPT="-A -- -a http://localhost:4011"
GWWS        SRVGRP=GROUP1  SRVID=4
             CLOPT="-A -- -i GWWS1"
```

Replace 'localhost' with a hostname or IP address suitable for your environment if necessary. Also use a different port if 4011 is already in use on your machine.

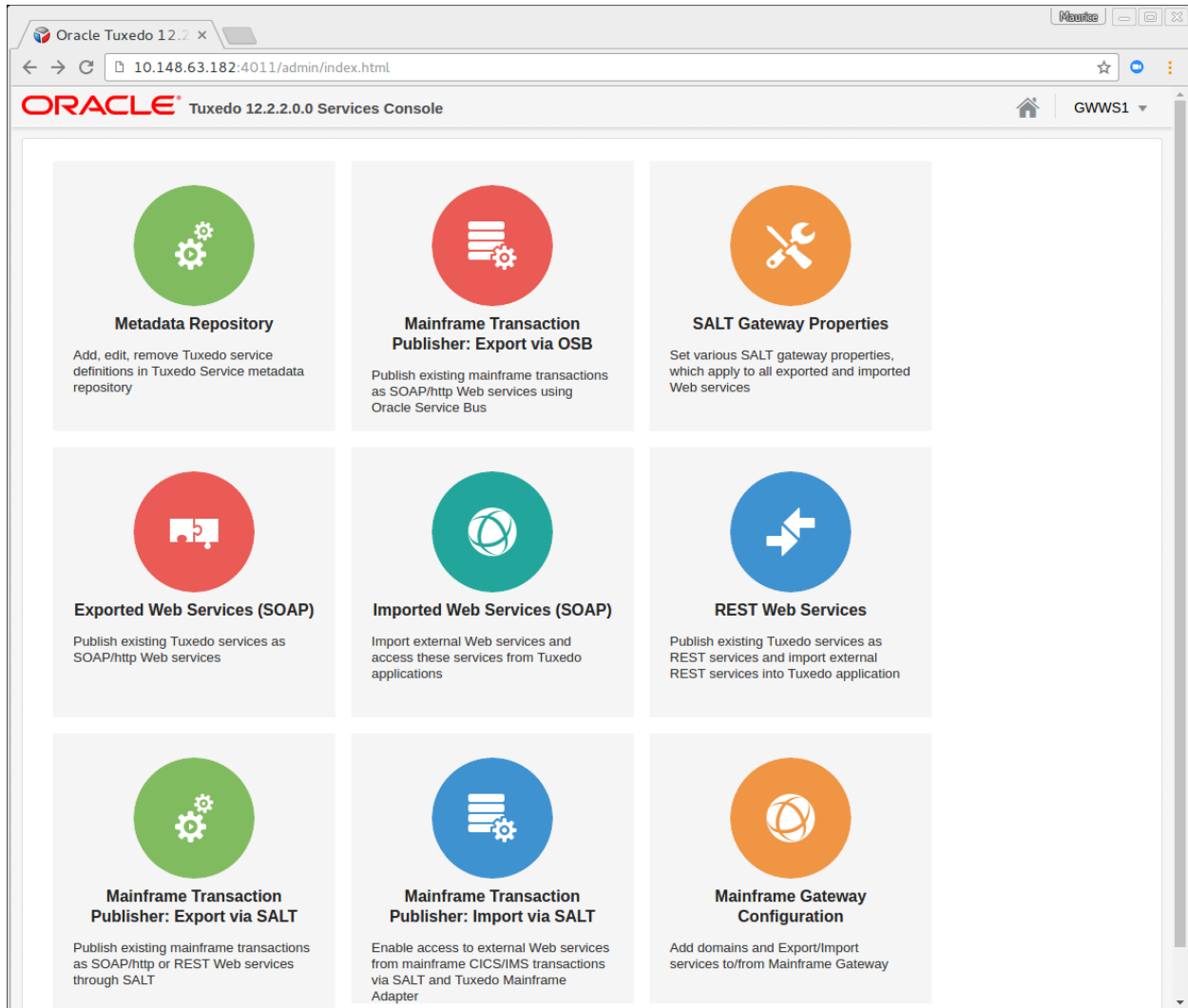
Reload tuxconfig and boot using commands such as:

```
$ tmloadcf -y ubbconfig
$ tmboot -y
```

Note that it is not necessary to create a SALTCONFIG file, GWWS will automatically generate one.

Log into the Tuxedo Services console and activate service contract discovery

Point your browser to the following URL: <http://localhost:4011/admin>



Go to **Metadata Repository** and click on **Enable Service Discovery**:

Oracle Tuxedo 12.2.2.0.0 Services Console

Tuxedo Services Metadata

Services

Filter by service name

Enable Service Discovery

Service name	SOAP	REST
No data to display.		

1

Delete Selected Service

Test Selected Service via Jolt

Service Details

Save Create New Service

* Service Enter a metadata service name

* Tuxedo Service Enter a service name

Service Description

Jolt Export Export JOLT service, Y/N

Service Type request-response

Input Buffer STRING

Output Buffer STRING

Error Buffer

Service Parameters

Manage Parameters

This will activate automatic population of the Tuxedo Service Metadata Repository. In order to expose a Tuxedo service as a web service, it is necessary that its Service Metadata be present so that a proper WSDL can be generated.

Perform typical service call to discover service interface (contract)

By calling an application with service discovery turned on, SALT will automatically gather the necessary information to create a Service Repository entry corresponding to the information exchanged during this call. Note that depending on the data exchanged, this interface may be different across calls. For example a call to the same Tuxedo service may return an ADDRESS field in one context, but not in another.

This can be performed using your existing application, or by simulating it. In this case we use UD32 with the following input, for example:

```
SRVCNM      SEARCHINVENTORY
COLOR ORANGE
```

Save this content in a file, for example uBike.ud32.

Calling ud32 with this input should display a number of lines:

```
$ ud32 < uBike.ud32
```

```
SENT pkt(1) is :
```

```
SRVCNM      SEARCHINVENTORY
```

```
COLOR ORANGE
```

```
RTN pkt(1) is :
```

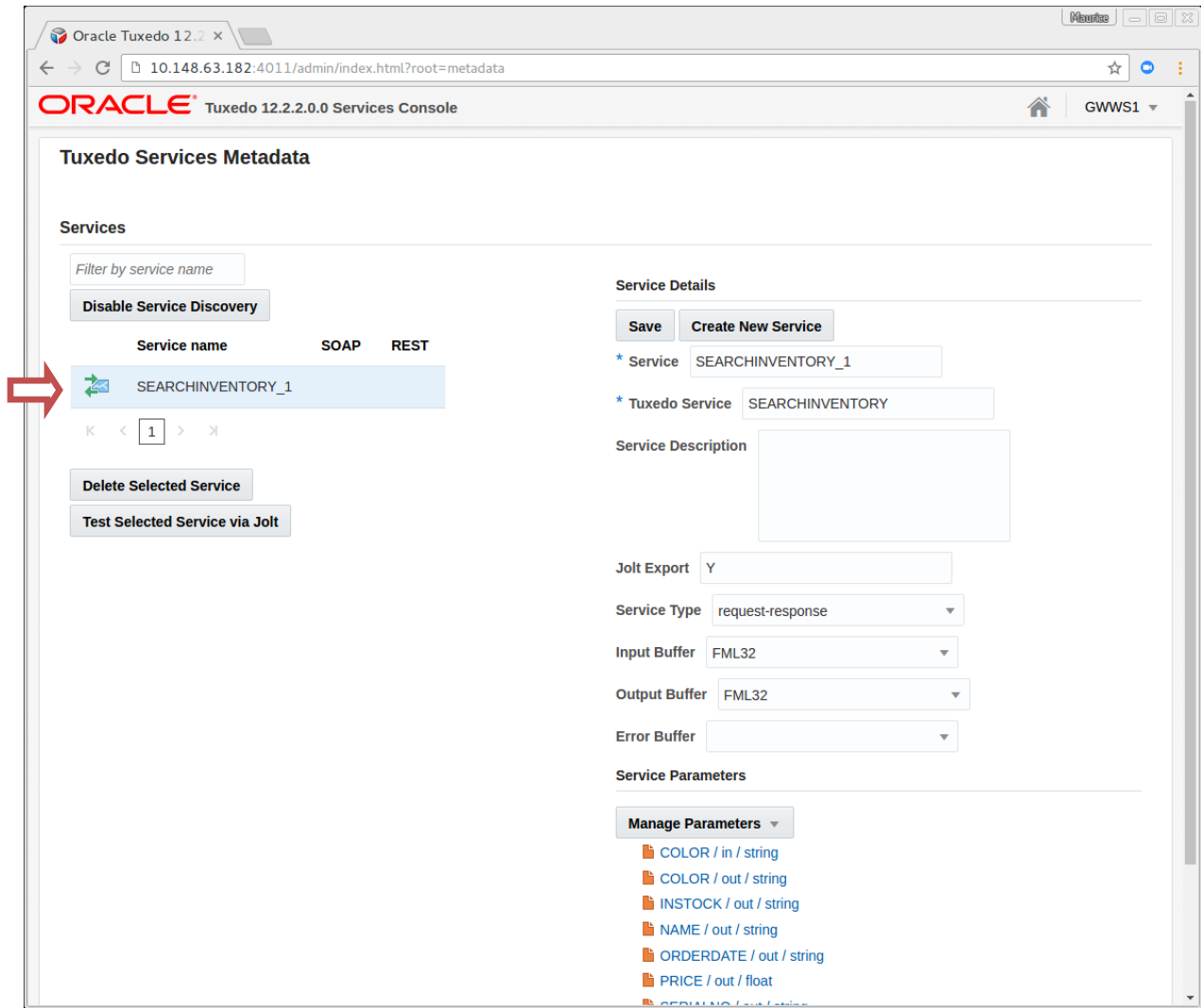
```
SIZE 58
```

```
SIZE 21
```

```
SIZE 16
```

```
...
```

After this call, re-loading the Metadata Repository page will show the new entry:



Clicking on it will show the service details.

Create a web service

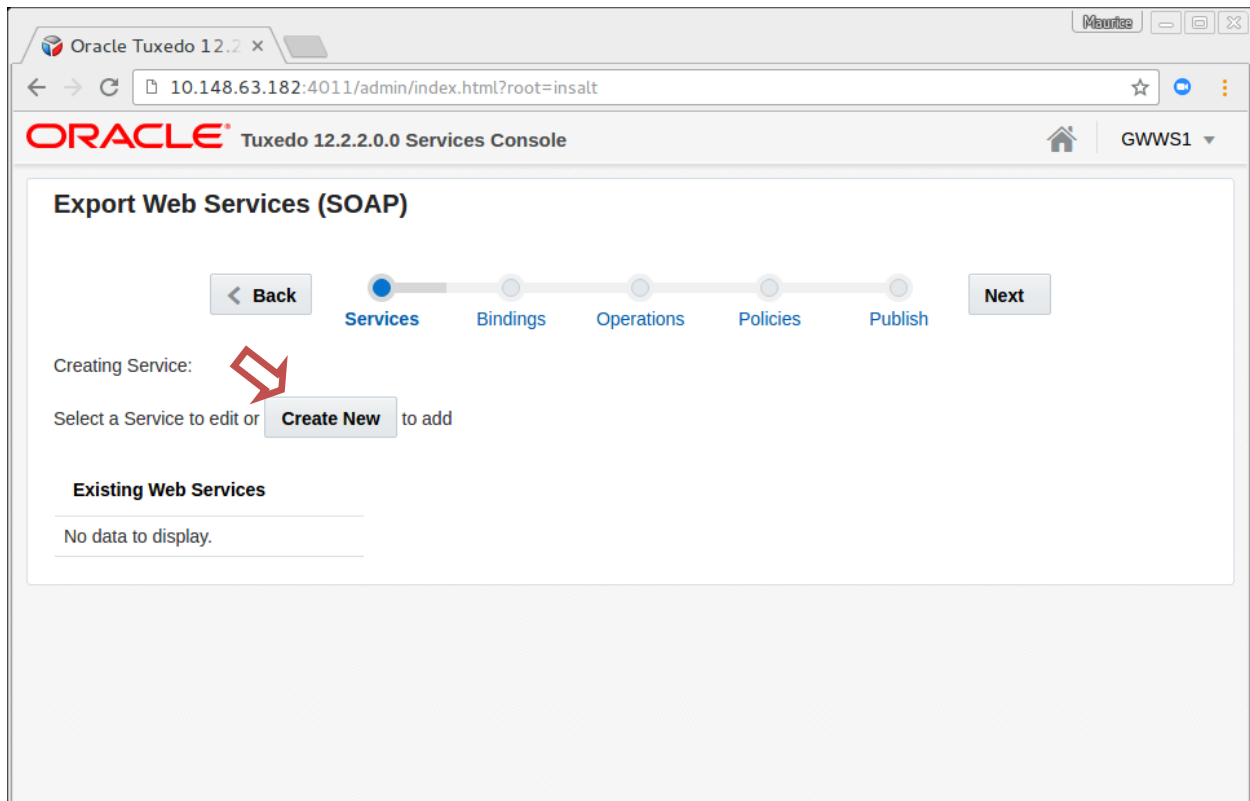
Go back to the home page by clicking on the home icon:



Then click on **Exported Web Services (SOAP)** to go to the native web services management page:



Click on **Create New**



The screenshot shows the Oracle Tuxedo 12.2.2.0.0 Services Console interface. The browser address bar shows the URL `10.148.63.182:4011/admin/index.html?root=insalt`. The page title is "ORACLE Tuxedo 12.2.2.0.0 Services Console" and the user is logged in as "GWWS1".

The main content area is titled "Export Web Services (SOAP)". It features a progress bar with five steps: "Services", "Bindings", "Operations", "Policies", and "Publish". The "Services" step is currently active, indicated by a blue dot. Navigation buttons include "Back" and "Next".

Below the progress bar, the text reads "Creating Service:" followed by a red arrow pointing to a "Create New" button. The text continues: "Select a Service to edit or **Create New** to add".

Under the heading "Existing Web Services", there is a table with the message "No data to display."

Enter a name for the service:

Oracle Tuxedo 12.2 x

10.148.63.182:4011/admin/index.html?root=insalt

ORACLE Tuxedo 12.2.2.0.0 Services Console GWWS1

Export Web Services (SOAP)

< Back Services **Bindings** Operations Policies Publish Next

Creating Service: uBikeSvc1

* Web Service Definition **Delete Service**

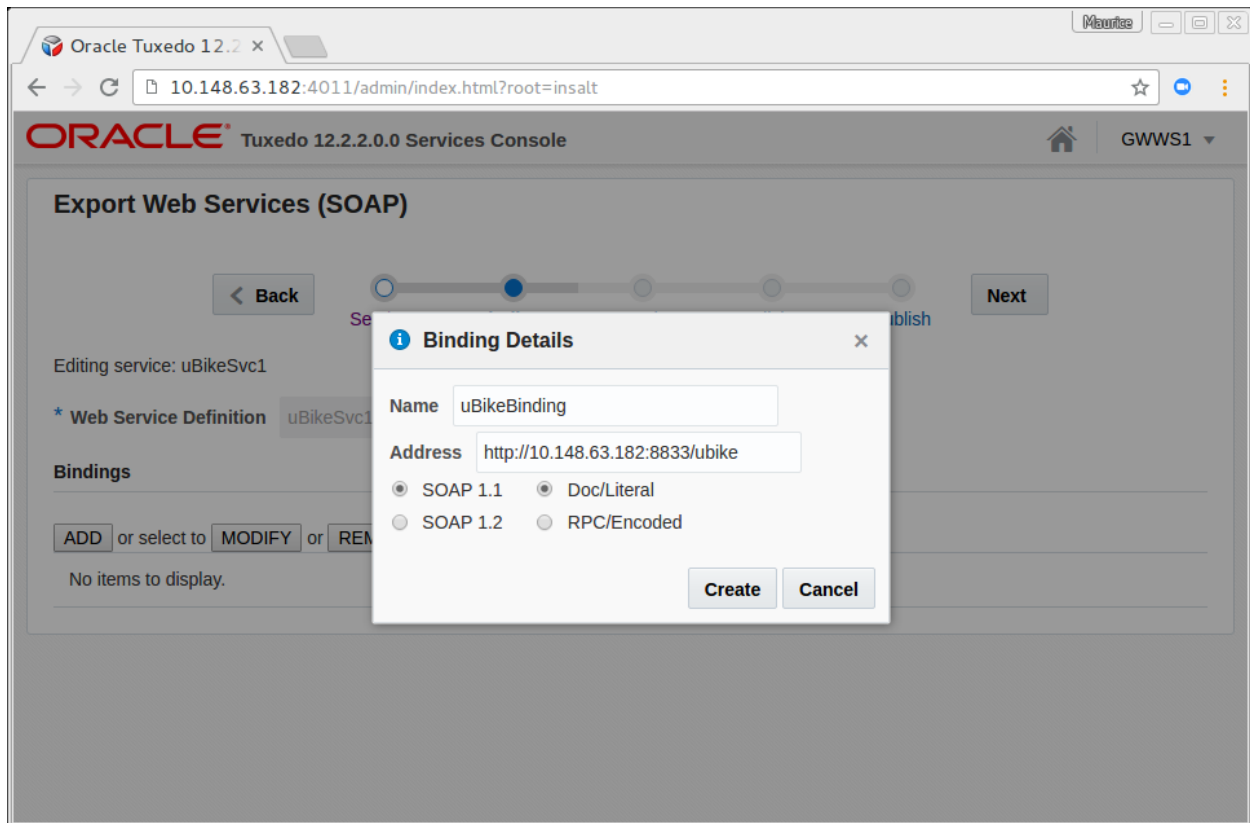
enter a web service definition name

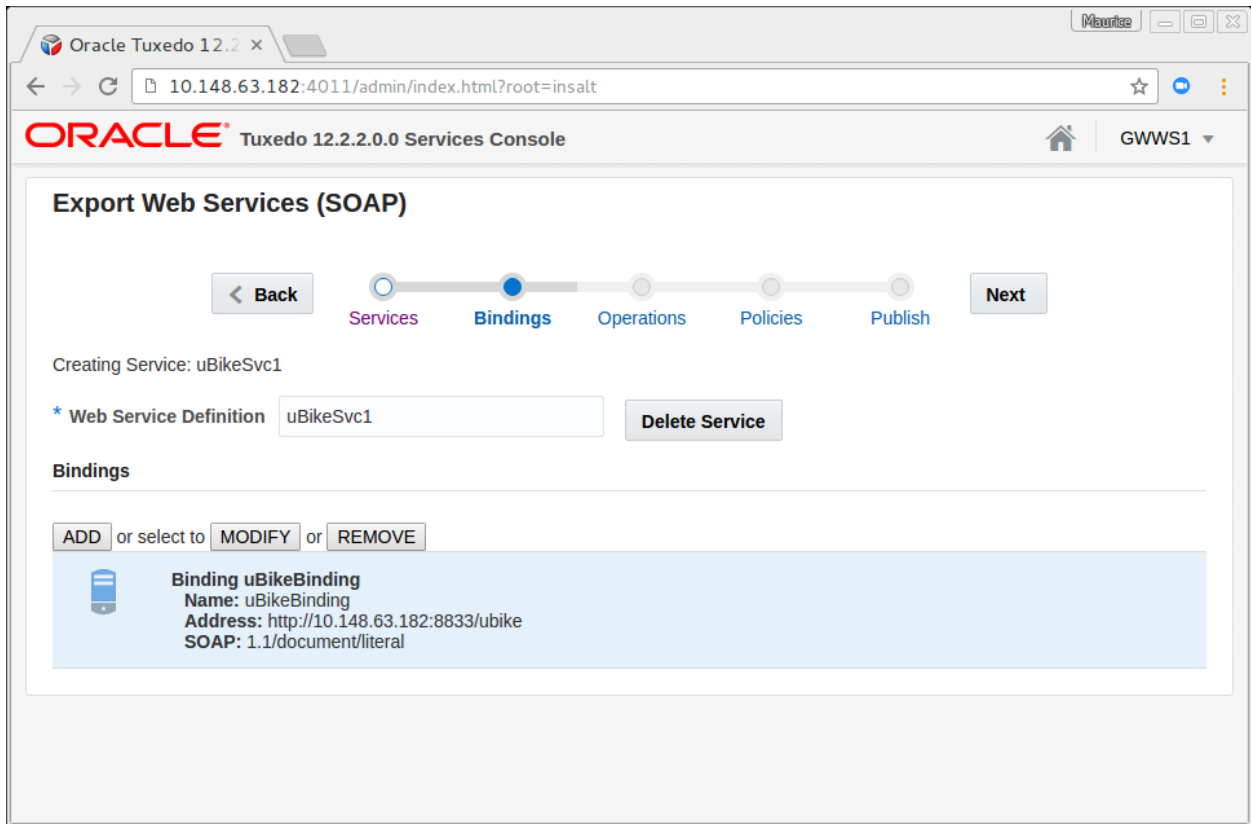
Bindings

ADD or select to **MODIFY** or **REMOVE**

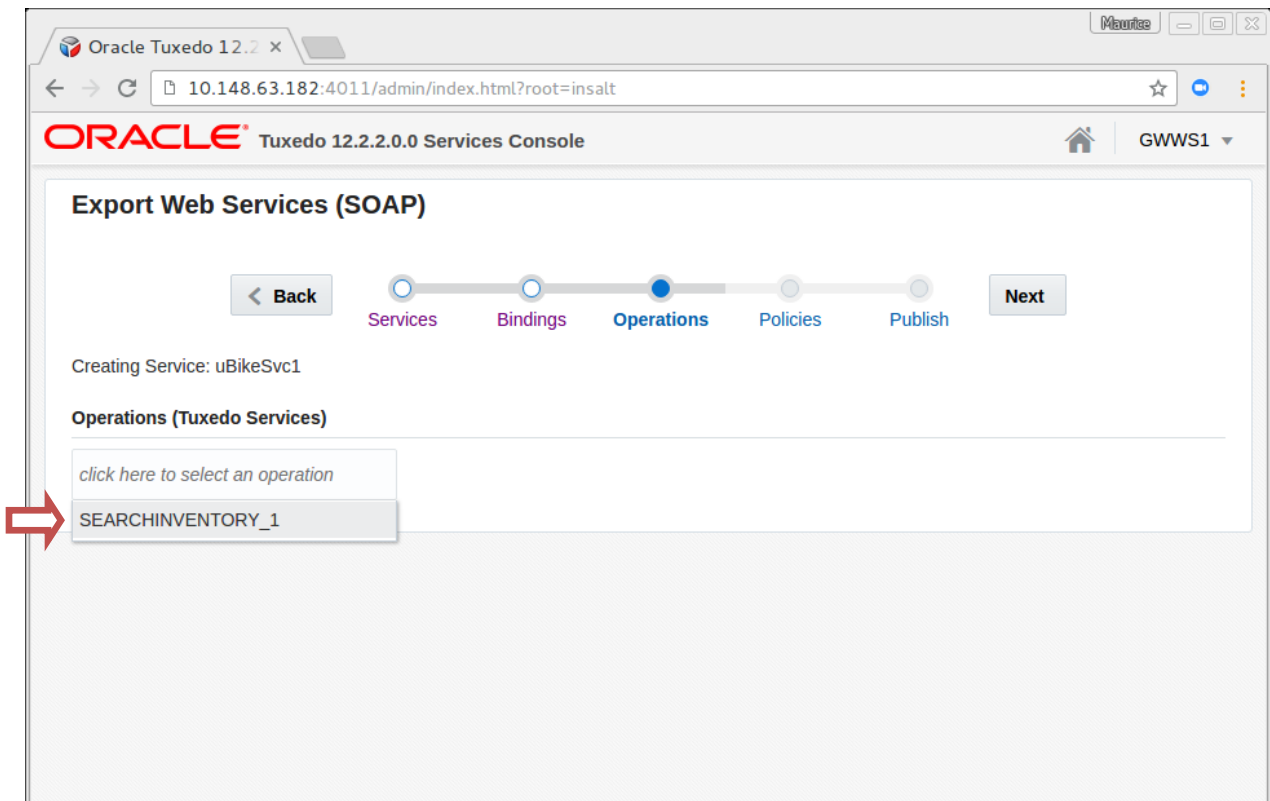
No items to display.

Then add a binding. The binding will be used to specify this service endpoint (network address where it can be invoked) and SOAP characteristics (version used and payload encoding). In this example we show the most common ones:





Click Next then add the newly created Metadata Service as an operation for this web service:



Oracle Tuxedo 12.2 x

10.148.63.182:4011/admin/index.html?root=insalt

ORACLE Tuxedo 12.2.2.0.0 Services Console GWWS1

Export Web Services (SOAP)

[Back](#) Services Bindings **Operations** Policies Publish [Next](#)

Creating Service: uBikeSvc1

Operations (Tuxedo Services)

SEARCHINVENTORY_1 x

Policies are not a very common use-case, skip over it and go to publish, where a summary of the service is displayed:

Oracle Tuxedo 12.2 x

10.148.63.182:4011/admin/index.html?root=insalt

ORACLE Tuxedo 12.2.2.0.0 Services Console GWWS1

Export Web Services (SOAP)

< Back Services Bindings Operations Policies **Publish** Finish

Creating Service: uBikeSvc1

Web Service Definition uBikeSvc1

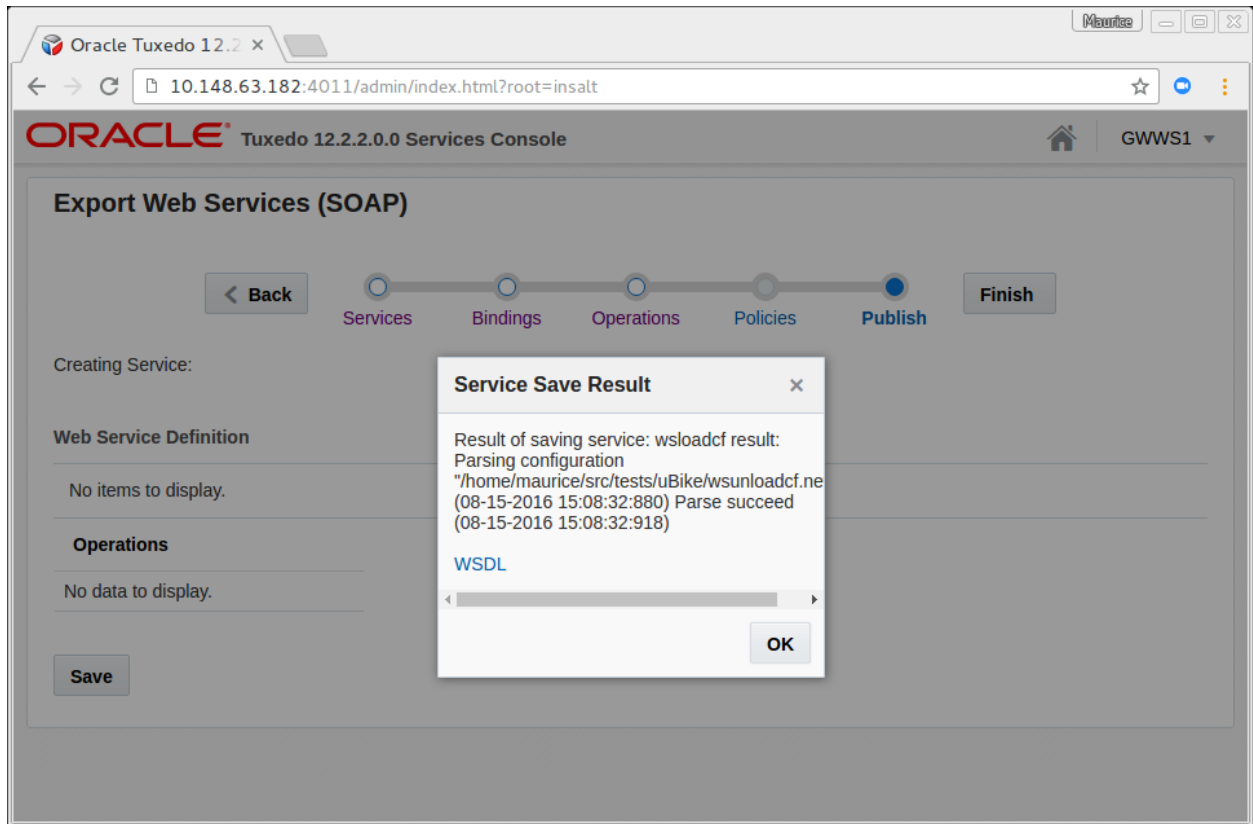
Binding uBikeBinding
Name: uBikeBinding
Address: http://10.148.63.182:8833/ubike
SOAP: 1.1/document/literal

Operations

SEARCHINVENTORY_1

Save

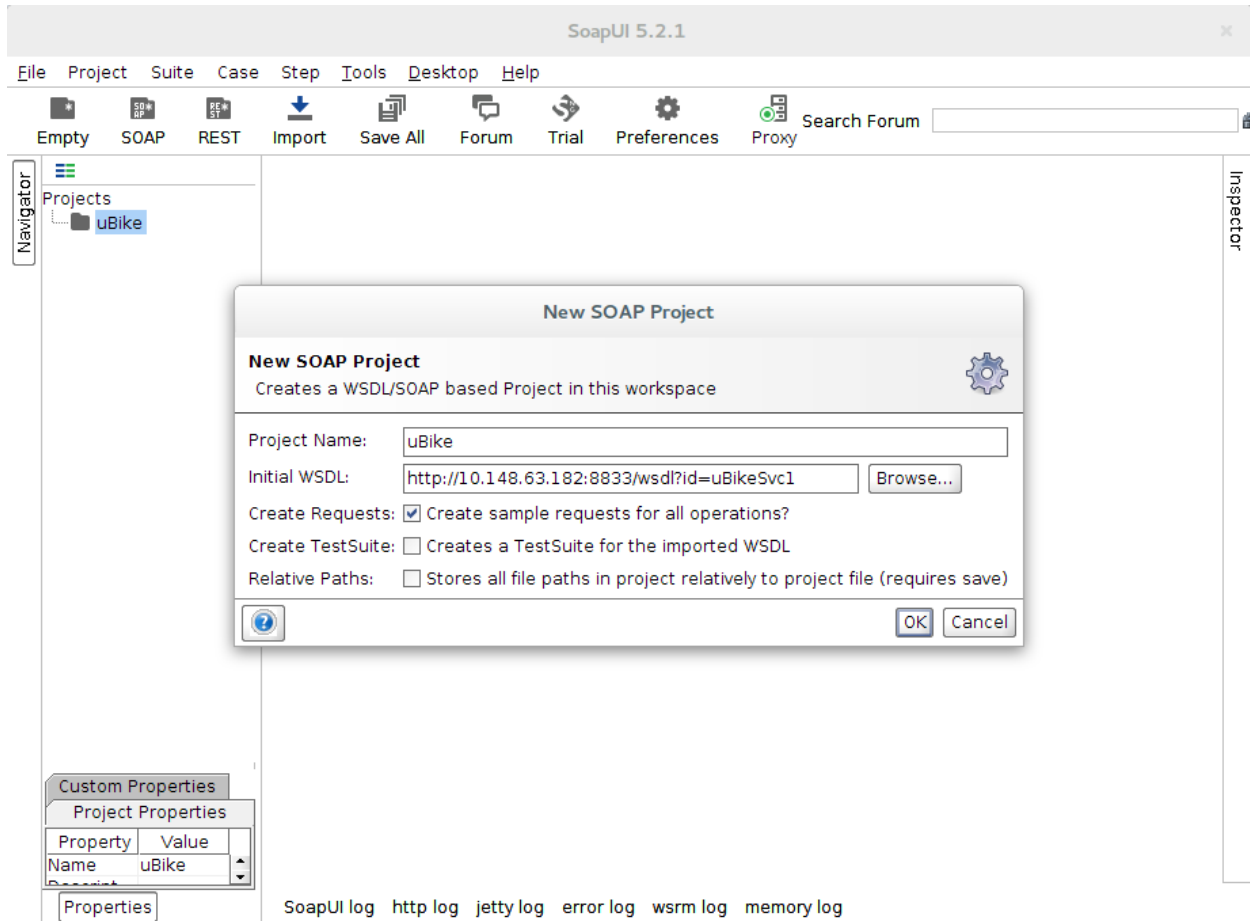
Click **Save** and the service is created. It is ready to be invoked.



You can click on the WSDL link and see its WSDL. It can be used to test the service in the next steps.

Verify by invoking web service using SOAPUI

Once the service has been created, start SOAPUI and create a SOAP project: menu **File**, select **New SOAP Project...**:



You can either manually enter the initial WSDL by entering the host, port used for the binding endpoint followed by /wsdl and the id with the name you entered for this web service. See example in screenshot.

Leave all default and click OK, then double-click on the **Request** under the operation name:

The screenshot shows the SoapUI application interface. The main window displays a SOAP request configuration for 'Request 1' at the URL 'http://10.148.63.182:8833/ubike'. The request body is shown in XML format:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Header/>
  <soapenv:Body>
    <urn:SEARCHINVENTORY_1>
      <urn:inbuf>
        <urn:COLOR?</urn:COLOR>
      </urn:inbuf>
    </urn:SEARCHINVENTORY_1>
  </soapenv:Body>
</soapenv:Envelope>
```

The interface includes a menu bar (File, Project, Suite, Case, Step, Tools, Desktop, Help), a toolbar with icons for Empty, SOAP, REST, Import, Save All, Forum, Trial, Preferences, and Proxy. A search bar is located on the right. The Navigator pane on the left shows a project tree with 'uBike' and 'SEARCHINVENTORY' folders, and 'Request 1' is highlighted under 'SEARCHINVENTORY'. The Request Properties table at the bottom left shows:

Property	Value
Name	Request 1
Description	
Message S...	341

At the bottom, there are log options: SoapUI log, http log, jetty log, error log, wsrn log, memory log. The Inspector pane on the right is currently empty.

You can then enter an argument in the **COLOR** element, similar to what you did in the UD32 script, for example enter the color **ORANGE**:

The screenshot shows the SoapUI interface with a SOAP request editor. The main window displays the following XML code:

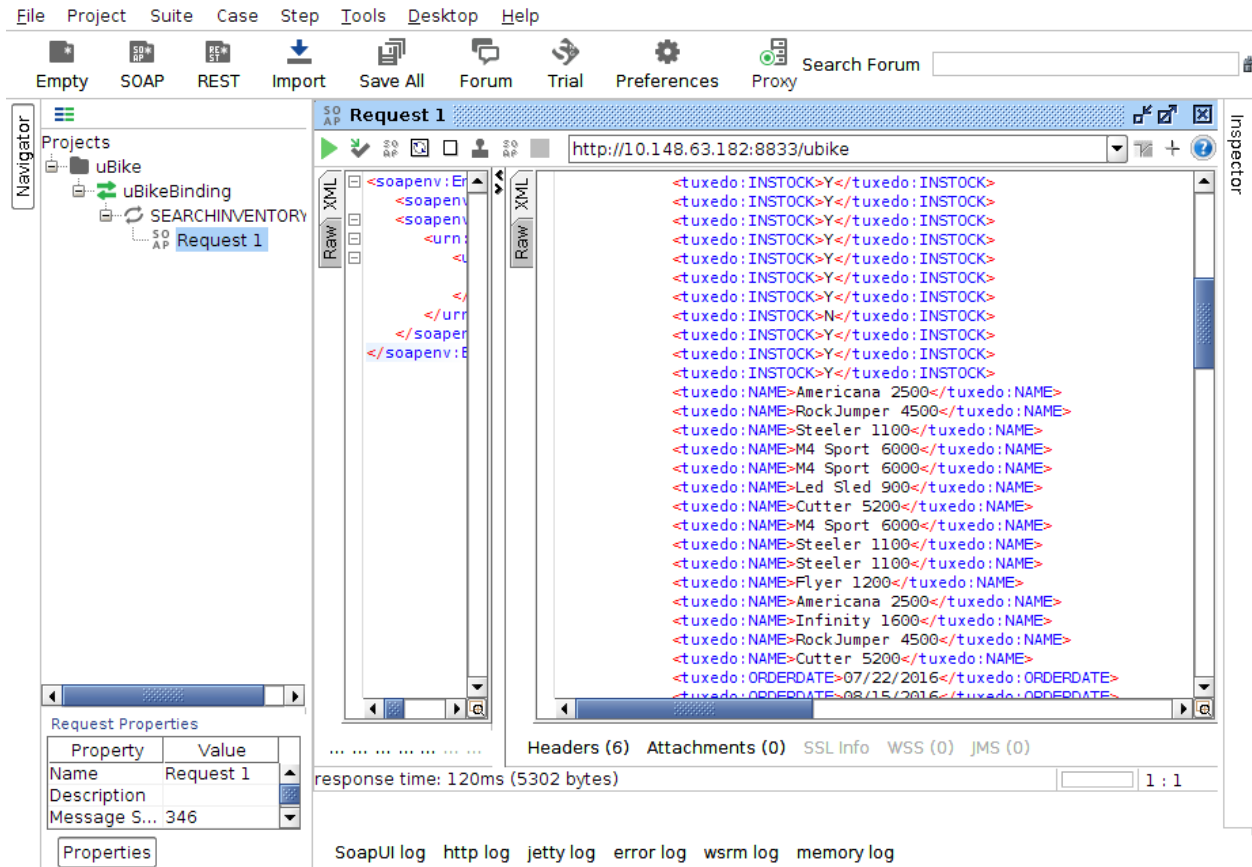
```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Header/>
  <soapenv:Body>
    <urn:SEARCHINVENTORY_1>
      <urn:inbuf>
        <urn:COLOR>ORANGE</urn:COLOR>
      </urn:inbuf>
    </urn:SEARCHINVENTORY_1>
  </soapenv:Body>
</soapenv:Envelope>
```

The **Request Properties** table is visible at the bottom left:

Property	Value
Name	Request 1
Description	
Message S...	346

At the bottom of the interface, there are log options: SoapUI log, http log, jetty log, error log, wrsm log, memory log.

Click on the green arrow, then:



Conclusion

This HOWTO shows how to create a SOAP web service from an existing Tuxedo service in a matter of minutes. This shows how all the configuration details are hidden, and all actions are dynamic: there is no need to restart any process.

Services of any complexity can be exposed in this manner, from simple STRING-type based services to complex VIEW or FML rich payloads.

Once set up, more capabilities can be added, such as encryption and security (SSL with Basic Access Authentication), WS-* features (WS-Security, WS-Addressing, WS-RM, WS-TX).