

# AuraPlayer Server Manager User Guide

This document is the sole property of AuraPlayer Ltd., it cannot be communicated to third parties and/or reproduced without the written authorization from AuraPlayer Ltd.



## Contents

Contents	2
Introduction	3
Main Menu	3
Service Types	4
Viewing Services	4
Service Manager Toolbar	5
Creating a New Service	5
Service Details	7
Running AuraPlayer Externally	8
Editing a Service	8
Recording HTML Services	.13
AuraPlayer Dedicated Recordings	. 14
Importing a Recoding to the ServiceManager	. 18
Executing, Editing and Using HTML Services	. 18
Supported Commands	. 19
Creating DB Services	.21
DB Queries / Statements	.23
Testing Queries and Acquiring Output Parameters	.24
Executing, Editing and Using DB Services	.25
Creating JavaScript Services	.26
Adding Input Parameters	.27
Setting Output Parameters	.27
Calling other services	.29
Throwing Errors	.29
Logging and Debugging	.29
Getting Support	. 30



## Introduction

AuraPlayer's Service Manager is the tool used to create services and HTML pages as well as store, share and manage them. It also makes it possible to access the files from anywhere.

## Main Menu

AURAPLAYER				
<				
🖋 Services				
🗎 HTML Pages				
Applications				
AuraTester				
III Monitoring Agents				
📥 Integration				
😋 Admin				
<b>?</b> Support				

The Service Manager Menu has the following components:

Services	Create and manage Services – View current services, create new
	services, and perform various operations such as recording, testing
	and more.
HTML Pages	Manage HTML pages created by AuraPlayer's Visualizer – View
	existing HTML pages, edit and download them.
Applications	Group your Services and Pages to applications – An application
	defines a business logic or process, spread across multiple HTML
	pages.
AuraTester	Create, edit and run Test Cases & Test Groups – Test your services
	by creating Test Cases. Assemble multiple Test Cases to Test Groups,
	to help you test complex scenarios.
Monitoring agents	Create, run and monitor Test Agents – Create monitoring Agents that
	run predefined Test Cases or Test Groups in constant intervals. Charts
	and logs provide you information on the result of each run as well as
	performance.
Integration	Export your services to external providers. Currently, only an export
	to Oracle Mobile Cloud Service (MCS) and VBCS are supported.
Admin	Perform administrative tasks and set your preferences. Backup your
	data, customize system behavior and manage ORP files.



# **Services**

## **Service Types**

AuraPlayer's ServiceManager supports types of services. All services execute the backend implementation at background, exposing only input and output fields.

- Oracle Forms service: executes an Oracle Forms recording.
- **HTML service**: executes a recording of any website, for example: JDE or PeopleSoft.
- **DB service**: executes a database statement or query.
- JavaScript service: executes JavaScript code (may call other services and contain custom logic).

#### **Viewing Services**

All kinds of services are displayed under the "**Services**" tab from the main menu (left toolbar). Click on the "Services" tab and the services list will be displayed.

Se	Service Manager Create. Edit & Manage your services						
ŀ	Action -						
sł	now 10 entities						•
							search
~	Service Name ~	Туре	Status ~	Modified - ~	Created By 🌱	Actions	
	getRequisition	• HTMI	Active	09/02/17 10:48:01	admin	=	
	and an and a second	TTTWIE .		03/02/17 10:40:01	ournin .		
ш	acme_customerDetails	Forms	Active	08/02/17 16:56:59	manager	E /	2 🖉 🔍 🔪
	mcs_getAllCustomers	Forms	Active	08/02/17 16:56:59	admin	=	2 🖉 🖉 🔽
	sql_deleteStateCities	DB	Disabled	08/02/17 16:56:59	admin		2 🖉 💽 🔽
	sql_getAverqgeCities	DB	Active	08/02/17 16:56:59	admin	•	2 🖌 🖉 💽

The type of the service is listed under the "**Type**" column.

You may use the dropdown box under the column name to filter and display a particular type only.



# **Oracle Forms Services**

This section introduces services wrapping Oracle Forms/EBS. For other types of services, see the "HTML Services" and "DB Services" sections bellow.

## **Service Manager Toolbar**

Service Manager Create, Edit & Manage your services

Record/Create	Record/Create a new service
Upload	Upload a service from an existing ORP/HTML file
Action	Perform batch operations on a set of selected services.
	To select services, check the box next to the service/s
Activate/Disable	Activate or disable the selected services
Export	Export selected services to a csv file
Print	Print a list of the selected services
Delete	Delete the selected services.

#### **Creating a New Service**

1. On the Service Manager Toolbar, click on "Create"  $\rightarrow$  "Forms service". The "Service Record" dialog will be opened.



2. Fill in the service information:

	Service Record
Service Name	Service
Service Form Url	http://ec2-54-203-157-52.us-west-2.compute.amaz
begin cancel	



Service Name	A unique name for the new service. Name should reflect the business process.
Form URL	The URL of the Oracle Form you wish to connect and record from. You can only record from a Forms server that has AuraPlayer installed. See the install guide for details.

- 3. Click "Begin" to start creating your service.
- 4. The "Recording Toolbar" and the "Oracle Forms" windows will appear.

## **Recording Toolbar**

B

The Recording Toolbar has the following components:

AURAPLAYER	Service Name	The name you selected for this recording
Service Name	Recording Time	The duration of the recording
Service Recording Time	Capture Parameters	Capture all the form's current fields and values
00:02:18	Sava & Continua	as output parameters
Capture Parameters		continue recording later on. You may use this option to capture a set of values mid-way through recordings.
	Cancel	Cancel the current recording
Save & Continue	Save & Exit	Stop the recording, and save your new service
Cancel		

#### 5. Creating Input Parameters

To create an input parameter simply input text into the text box, click on a check box, select an option etc. In general, any insertion of values during the Forms session will mark the field and value as an input field.

#### 6. Creating Output Parameters

a. To capture specific output parameters - Click on the desired text fields in the Form, once clicked, they will be captured as output parameters.



- b. To capture ALL of the output parameters on a specific form Click the "Capture Parameters" button on the toolbar. All the fields in the Form will be captured as output parameters.
- 7. To stop recording without saving the service, click on the "Cancel" button.
- 8. To save and end the recording, click the "Save & Exit" button. The service is now saved and can be found in the 'Service Manager'. Once pressing the "Save & Exit" button, you will be routed to the 'Edit Service' page to finalize the creation of the Service.

#### **Recording Feature: Capture Parameters**

The "Capture Parameters" feature allows you to capture all fields in the Form that is being recorded, marking them as output parameters.

- 1. Once you reach the Form that has the fields you would like as output fields on your mobilized application, click the "Capture Parameters" button on the Recording Toolbar.
- 2. Upon success, your recording toolbar will display a success message on the top left corner.

For more details about editing parameters, see the "Edit a Service" Section.

#### **Service Details**

To view the service's details, click on the service name or use the Details button.

Se	Service Manager create, Edit & Manage your services						
ŀ	Action -						
sł	now 10 en 🔻						search
~	Service Name 🛛 👋	Service Workflow $\ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \$	Status $\checkmark$	Modified 👻 👋	Modified By $\stackrel{\scriptstyle \scriptstyle \sim}{}$	Actions	
	Service01	Service01.orp	Active	08/07/15 11:11:48	manager		2 🔹 🔨
	Service02	Service02.orp	Active	08/07/15 10:53:38	manager	E 🖉 🗸	<b>2</b>

The "Service Details" section displays the following information:

Service Name	The name of the service
Description	A brief description of what the service does
Service URL	The RESTful service URL. This URL will be used to create RESTful APi's
WSDL URL	SOAP URL for the service description file (WSDL)
Form URL	The URL of the Oracle Forms server that the service is running against



Status Service status - Active or Disabled

## **Running AuraPlayer Externally**

To run the AuraPlayer Service externally, copy the relevant URL (SOAP/REST) from the service details page and create the service client in any tool of your choice. It is also possible to copy the REST URL to your choice of browser.

## **Editing a Service**

To edit an existing service, click on the "Edit" button

Se	Service Manager create, Edit & Manage your services						
E	Action -						
s	how 10 en 🔻						search
$\sim$	Service Name	Service Workflow $\ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \$	Status $\checkmark$	Modified $\checkmark$	Modified By $\checkmark$	Actions	
	Service01	Service01.orp	Active	08/07/15 11:11:48	manager		2 🔹 🔨
	Service02	Service02.orp	Active	08/07/15 10:53:38	manager	E / 🗸	e 🔨

#### **Service Details**

Service Details	✓
Service Name	Service01
Description	description
Form URL	http://ec2-54-203-157-52.us-west-2.compute.amazonaws.com:9001/forms/frmservlet?config=Con

Service Name	The name of the service
Description	A brief description of what the service does (Optional)
Form URL	The URL of the Form server. You may enter a different URL than the one
	used at the time of recording.



#### **Input Parameters**

The 'Input Parameters' are the parameters that the service receives from the end-user.

Input	Paramet	ers			*
	Visible	Name	Label	Default Value	Move
+		MAIN_USERNAME_0	Name	MIA	V
÷		MAIN_PASSWORD_0	Password	•••••	T
+		S_CUSTOMER_ID_0	Id	202	

÷	Drag parameters up or down to change their order
Visible	Choose whether the parameter will be visible in the application or not. For example, you may set a hidden parameter with a fixed predefined value.
Name	Unique name of an input filed, as captured from the Oracle Form
Label	Label to be displayed to the end-user near this field
Default Value	Value to be set as the initial value of this field (Optional).
Move	Copy the current parameter to the Output Parameters list

#### **Output Parameters**

The 'Output Parameters' are the parameters the form returns in response to the input parameters.

Ou	tput	Parame	ters				*
		Visible	Name	Label	Multi Record	Actions	
	÷	$\checkmark$	S_CUSTOMER_NAME_0	Name Required		90 <b>8</b>	
	÷	$\checkmark$	S_CUSTOMER_PHONE_0	Phone		8 00	
	÷	$\checkmark$	S_CUSTOMER_ADDRESS_0	Address		8	
	÷	$\checkmark$	S_CUSTOMER_CITY_0	City		Fail if Response is empty	
	÷	$\checkmark$	S_CUSTOMER_STATE_0	State		Fail If Response does NOT contain value 'Washington'	

÷	Drag parameters up or down to change their order
Visible	Whether this parameter will included in the output
Name	Unique name of the input filed, as captured from the Oracle Form
Label	Label to be displayed to the end-user near this field
Multi Record	



#### Service validation (advanced feature)

You may add validations to the returned values of output parameters.

Click the button next to a parameter, and select one of the validations in the displayed popup. The service will return an error if validation condition is not filled. You may control the returned HTTP code in case of failure in the "Advanced details" section:

•

Validation Failure Status

400 - Bad	Request	
-----------	---------	--

#### **Advanced Details**

Advanced Details	×
Filename	getCustomerDetails1234.orp
Partial Service	
Enable Service	
Use Labels As Keys	
Number of Rows	20
Authentication Type	None •
Use JSON	
Handle Popups	
Validation Failure Status	200 - OK •

Filename	Name of the ORP file the service is saved from	
Partial Service	Partial Service is a recording that does not start from the beginning	
	of the scenario and is not independent. It relies on a previous service	
	to run in order to succeed	
Enabled Service	Enable or disable the service	
Use Labels As Keys Call the service with the labels as the parameter names, inste		
-	using the recorded names from the Oracle Form	
Number of Rows	Total number of rows you wish to display in the result set	
Authentication	None – service could be activated simply by accessing its url.	
	Basic Authentication – service will require a valid ServiceManager	
	username & password before running (in 'Authentication' header).	



	<b>Field Encryption</b> - service will require a username & password in 'Authentication' header, passing them to 2 of its input parameters.
Use JSON	Check to receive responses from your REST Service in JSON format
Handle Pop-ups	Deals with pop-ups displayed over the Form during service playback. Check this box if you would like to have the service automatically click on the default button & continue with the playback.
Validation Failure Status	See the "Service validation" section on the previous page.

Finally, click 'Update Service' to save your changes to this service.

## **Testing the Service**

1. To test your service, click the test button in the actions column.

Service Ma	NAGEL Create, E	Edit & Manag	ge your services			
+ Action -						
show 10 en 🔻						search
<ul> <li>Service Name</li> </ul>	Service Workflow $\scriptstyle$	Status $^{\scriptstyle \vee}$	Modified 👻 👋	Modified By $\checkmark$	Actions	
Service01	Service01.orp	Active	08/07/15 11:11:48	manager		2 • 🔨
Service02						

The 'Test Service' page is displayed.

Test serv	vice
	Test     View Log     Create Test Case     Reset Values     Back
Details	★
Service Name	Service01
Description	
Service Url	http://ec2-54-203-157-52.us- west-2.compute.amazonaws.com:7001/ServiceManager/Macro/ExecMacro/getCustomerDetails_byAlex_7



2. To change the input parameters default values, change the values in the "Default Value" column.

Input Parameters		<b>~</b>
Name	Label	Default Value
MAIN_USERNAME_0	Name	MIA
MAIN_PASSWORD_0	Password	•••••
S_CUSTOMER_ID_0	Id	202
		Form Default Values Clear Values Reset Values

3. Once all values are set, click the 'Test' button. The XML response will appear in the text box below

test service result for Service01	<b>*</b>
1 K?xml version="1.0" encoding="UIF-8" ?>	•
2 * <response></response>	
3 * <getcustomerdetails_byalex_ message=""></getcustomerdetails_byalex_>	
4 * <error></error>	
5	
6 * <popupmessages></popupmessages>	
/	
8 <statusbarmessages>Enter a query; press Ctrl+F11 to execute, F4 to cancel.;</statusbarmessages>	BarMessages>
9	
10 < <getcustomerdetails_byalex_ liements=""></getcustomerdetails_byalex_>	
11 <s_customer_address_0>6741_Takashi_Blvd.</s_customer_address_0>	
12 <s_customer_city_0>Osaka</s_customer_city_0>	
13 <s_customer_comments_0>Customer should always pay by cash until his credit rating im</s_customer_comments_0>	iproves
.	
14 <s_customer_credit_rating_0>POOR</s_customer_credit_rating_0>	
15 <s_customer_name_0>Simms Atheletics City</s_customer_name_0>	
16 <s_customer_phone_0>123456</s_customer_phone_0>	
17 - <s_customer_state_0></s_customer_state_0>	
18	
19 <s_customer_zip_code_0>55787</s_customer_zip_code_0>	
20 <s_ord_date_ordered_0>23-sep-2013</s_ord_date_ordered_0>	
21 <s_ord_date_shipped_0>25-oct-2013</s_ord_date_shipped_0>	
22 <s_ord_id_0>5007</s_ord_id_0>	
23 * <s_ord_order_filled_0></s_ord_order_filled_0>	*

The Forms error-messages, status-bar messages and pop-up messages can be found at the top of the response.



# **HTML Services**

HTML services is our Automation support for web, websites and HTML pages. The 'HTML Service' recorder captures website use cases for automation from our ServiceManager or other integration clients.

Your ServiceManager operates with HTML Services in a very similar way to Oracle Form Services.

#### **Recording HTML Services**

Before beginning, you must get the IDE installation from your AuraPlayer contact.

ile Edit Ac	tions Options	
ase URL htt	p://auravm01.us-west-2.compute.internal:8000/	
Fast Slow		(
Fort Care	Table Source	
Intitled		
intracu	# Command Target	Value
	Command	
	Value	Select Find
uns: 0	Target       Value       ServiceManager supported	Select Find
uns: 0 ailures: 0	Target       Value       V ServiceManager supported	Select Find
uns: 0 iilures: 0 Log Stored	Target Value ServiceManager supported	Select Find
uns: 0 iilures: 0 Log Stored	Target       Value       Ø ServiceManager supported	• Select Find
uns: 0 iilures: 0 Log Stored	Target       Value       Ø ServiceManager supported       -Vars       SM Reference       Reference	v Select Find
uns: 0 iilures: 0 Log Stored	Target       Value       Image: ServiceManager supported       d-Vars       SM Reference       Reference	Select Find
uns: 0 iilures: 0 Log Stored	Target Value ServiceManager supported	• Select Find
uns: 0 iilures: 0 Log Stored	d-Vars SM Reference Reference	Select Find
uns: 0 iilures: 0 Log Stored	Target Value ServiceManager supported	▼ Select Find

Once the installation is complete, open the IDE toolbar from the 'Tools' menu (or press Ctrl+Alt+S):

Windows7 [Running]	
	L ☆ @ → ♠ ♥ ≡
Q Search >	
! Q. Search 🖂 🚽 🖨 🦁 🚍	
moz://a	



The record button should be pressed automatically so make sure that it is pressed, and you are ready to go with your recording.

🥑 AuraPlay	er IDE 2.9.1		
<u>File</u> <u>E</u> dit	Actions Options		
Base URL	http://auravm01.us-west-2.compute.in	nternal:8000/	
Fast SI	🐃 🌬 💷 💀 🛨 🛨		
Test Case	Table Source		
Untitled	# Command Ta	arget Value	
-			
	Command		
	Command		
	Target		Select Find
	Value		
Runs:	0 ServiceManager supported		
Failures:	0		

In the browser window, perform the actions you would like to record: start by entering a URL to the address bar of the browser, and continue with the rest of your business process (for example – enter username and password, click on a login button, click on some links...).

## **AuraPlayer Dedicated Recordings**

To support our unique functionality, we require some minor adjustments to the recordings generated by the AuraPlayer IDE. They are mandatory to support the service structure, keep robust to changes as well as allow the optimizations we apply for better performance.

East Slow	p://auravmu1.us-west-2.co	mpute.internai:0000/		
ast Side	• • • • • • • • •	-		
fest Case	Table Source			
Intitled *	# Command	Target	Value	
	3 type	id=usernameField	operations	
	4 type	id=passwordField	welcome	
	5 clickAndWait	id=SubmitButton		
	6 assertTitle	Oracle Applications Home Page		
	7 open	/OA_HTML/RF.jsp?function_id=100	4	
	8 assertTitle	Oracle iProcurement: Shop		1
	9 clickAndWait	id=ICXPOR_REQSTATUS		
	10 assertTitle	Oracle iProcurement: Requisitions		
	11 storeText	id=N19:RequisitionNumber:0	Requisition	
	Command	storeText		-
	Target	id=N19:RequisitionNumber:0	- Select	Find
	Value	Requisition		
uns: 0	ServiceManager su	pported		
niures: 0		· · · · · · · · · · · · · · · · · · ·		
Log Store	-Vars SM Reference	Reference		
etem Text/le	ator variableName)			
Generated fro	m getText(locator)			
Arguments:				
	tor - an element locator			
loca     Returns:				
loca     Returns:     the	ext of the element			



This screenshot of an example recording will be referenced from the next paragraphs. **Input parameters** 

The input parameters of your service are automatically captured from the data you type, links you click, etc. Some types of actions generate input parameters, while other actions do not. Some of the actions that produce input parameters, produce them only when used with particular kinds of **targets**. These targets define how fields will be referred to during playback.

In the example above, the selected action (action #7) – *clickAndWait* – locates the element to click by id (*id=MyReqsTable:RequisitionNumber:2*). By clicking on the down arrow in the 'Target' field, you may see alternative methods to identify the field to be used by the action – all naming elements in the list refer to the same item.

If you require the data to be configurable, always **prefer a simple target**:

\* "link=14305"

Generates an input parameter with default value '14305'.

\* "//span[@id='MyReqsTable']/table[2]/tbody/tr[4]/td/a"

Does NOT generate an input parameter since it is too complex and hard to identify the part to be configured. In general, XPath expressions will NOT generate input parameters.

#### **Output parameters**

Output parameters cannot be implicitly inferred from the recording, you will have to explicitly define them.

AuraPlayer uses the *output (storeText)* action to declare the output parameters of a service. On the example above, the textual value of the element located by *id=N12:FunctionalAmount:0* is an output parameter of the service.



To add a new action, click on  $\uparrow$  (up arrow) "Insert Output Command".

ase URL htt Fast Slow	tp://auravm01.us-west-2.co	mpute.internal:8000/		•
est Case	Table Source	Jacob Output Command		
ntitled *	# Command	Target	Value	
	2 assertTitle	Login		*
	3 type	id=usernameField	operations	_
	4 type	id=passwordField	welcome	
	5 clickAndWait	id=SubmitButton		
	6 assertTitle	Oracle Applications H	ome Page	-
	7 open	/OA_HTML/RF.jsp?fur	nction_id=1004	-
	8 assertTitle	Oracle iProcurement:	Shop	
	9 clickAndWait	id=ICXPOR_REQSTAT	US	
	10 assertTitle	Oracle iProcurement:	Requisitions	
				-
	Command			-
	Target		Select	Find
	Value			
ins: 0	SenviceManager cu	ported		
ilures: 0	)	pported		
.og Store	d-Vars SM Reference Re	eference		
open(url) Arguments: • url - Opens an Ul proceeding, restrictions i new browset	- the URL to open; may be rela RL in the test frame. This acce ie. the "AndWait" suffix is im; in the browser (Same Origin P r session on that domain.	ative or absolute pts both relative and absolute 1 plicit. Note: The URL must be olicy). If you need to open an 1	JRLs. The "open" command waits for the j on the same domain as the runner HTML d JRL on another domain, use the Selenium :	bage to load before tue to security Server to start a

Then, click on '*Select*' button next to the '*Target*' field, and click on the value you wish to capture in the browser window that displays your website.

In the '*Value*' field (variable name) enter a meaningful label for the output parameter to capture the value into.

Capturing **table columns** - capturing the output field is similar to single row recording.

On the Service editing it is required to set in the fields as 'Multi Record'.

ame	Label	Multi Record	Configurations
y_idN39_NtfFromUser_0	From	<ul> <li>Image: A set of the set of the</li></ul>	
y_idN39_NtfType_0	Туре	~	
y_idN39_NtfSubject_0	Subject	~	
y_idN39_NtfSent_0	Sent	~	
ror			Fail if Response is NOT empty
ppupMessages			
atusBarMessages			



Note that there is also a checkbox '*Table has header*' at the *Advanced* section of the *Service Editor*, which should be checked - if the resulting table returns the column names as the first output line of the table.

A cheat sheet of toolbar commands that your ServiceManager supports is attached at the end of this section.

#### **Finish your recording**

Once you have completed your recording, click on the **Record** button in the toolbar window to stop the recording (see screenshot above).

Open the File menu, and click on Save Test Case...

Choose a name for your recording (with .html extension), and save it in convenient location. You will need to refer to this from within the ServiceManager to create the playback automation shortly.

New Te	et Care	Ctria NI kt-2.	compute int	ernal:8000/			
1400 10	scease	Culton	•				
Save Te	st Case	Ctrl+S					
Save Te	St Case A						
Export	lest Case	As	Tar	get	Value		
Recent	Test Case	s Nait	id=5	SubmitButton			
			Ora	cle Applications Home Page			
Addle	st Case	Ctrl+D	/0A	HTML/RF.jsp?function_id=1004			
Close (	S C	Ctrl+W	Ora	cle iProcurement: Shop			
	9	clickAndWait	id=1	CXPOR_REQSTATUS			
	10	assertTitle	Ora	cle iProcurement: Requisitions			
	11	storeText	id=1	id=N19:RequisitionNumber:0 Requisition		ion	1
	12	12 storeText id=		N19:ReqTotal:0	Total		
	13	storeText	id=1	N19:ApprovalStatus:0	Status		
	Co	mmand		storeText			
	Tar	rget		id=N19:ApprovalStatus:0	-	Select	Find
	Val	ue		Status			
Runs:	0	ServiceManager	supported				
ailures:	0						
				1 *			
Log Stor	ed-Vars	SM Reference	Reference				
storeText()	ocator, va	riableName)					
Argument	rom get te	xt(locator)					
• lo	ator - an e	element locator					
Paturna	text of th	e element					
the starting	U TO LEAL	C CICILICIAL					
the							



## Importing a Recoding to the ServiceManager

Open your ServiceManager on the 'Services' tab, and click on the **Upload Recording** button. **T** Select the recording file you created from the HTML toolbar, and the Service Editor will be displayed.

	Se	ervice Manager	Create, Edit & Manage your services	
Applications		Action -		
AuraTester	sh	ow to entitles		
III Monitoring Agents		Service Name	Linked Applications	Тур
Automation		approveRequisition		нт
<ul> <li>Admin</li> <li>Support</li> </ul>		getRequisitionList		ΗT
		Alex_names_in_input_table		Fo
		ApprovePurchaseOrder		JS
		Azura gatRaarar	Ature mass Ature clientAnn	

#### Viewing recorded HTML services

All services are listed in the 'Services' tab accessible from the left side toolbar. The 'Type' column in the services list, distinguishes **HTML** Services from other services. And the services can be filtered by clicking on the title and selecting your choice of service type.

#### **Executing, Editing and Using HTML Services**

All actions are to be performed in the same manner as with the regular Oracle Form Services (Page 8). Consult this manual for further instructions.

#### **HTML Services output**

The HTML Services maintain the same response structure as the Oracle Form Services. They can be tested using the test button from the "Services" detail or edit pages.



Service	Service test result for Recording_OpenRequisition				
1 -	{				
2	"success": true,				
3 -	"Response": {				
4	"Recording_OpenRequisitionSuccess": true,				
5 -	"Recording_OpenRequisitionElements": {				
6	"Amount": 248				
7	},				
8 -	"Recording_OpenRequisitionMessage": {				
9	"Error": "",				
10	"PopupMessages": "",				
11	"StatusBarMessages": "Oracle iProcurement: Requisitions"				
12	}				
13	}				
14	}				

The contents of the 3 special output parameters are:

- Error HTML playback errors, if any.
- PopupMessages not relevant for HTML type services (always returns an empty string).
- StatusBarMessages The page title of the execution of the last command.

#### **Supported Commands**

All supported commands, resources and description appear on the HTML Recorder at 'SM Reference' tab at the bottom of the recorder

😻 Untitled (unti	itled suite) - AuraPlayer IDE 2.9.1 *				
<u>Eile Edit Actions Options</u>					
Base URL http:	s://www.yahoo.com/				
Fast Slow	🕨 🗣 🛨 🛨				
Test Case	Table Source				
Untitled *	# Command Ta	get Value			
	1 createCookie				
	Command Target	createCookie			
Runs: 0 Failures: 0	ServiceManager supported				
	$\sim$	▼▼			
Log Stored-	Vars SM Reference Reference				
createCookie(n Arguments: • name' • option option value	nameValuePair, optionsString) ValuePair - name and value of the cool usString - options for the cookie. Curre usString's format is "path=/path/, max_ of 'max_age' is second. Note that spec	tie in a format "name=value" ttly supported options include 'path', 'max_age' and 'domain'. the age=60, domain=.foo.com". The order of options are irrelevant, the unit of the fying a domain that isn't a subset of the current domain will usually fail.			
		······································			
		▼			



A few main commands that we support:

Command	Usage
assertLocation	Get the absolute URL of the current page and assert that it
	matches a regex. If does not match - abort execution.
assertTitle	Get the title of the current page and assert that it matches/not-
assertNotTitle	matches a regex.
	If match result is different from expected - throw an exception,
	aborting execution.
click   clickAndWait	Clicks on a link, button, checkbox or radio button.
(locator, optionalAction*)	If the click action causes a new page to load – wait for it to finish.
	optionalAction – an optional JavaScript code to be injected as the
	'onclick' action of the button.
pause	Wait for the specified amount of time (in milliseconds).
(millisToWait)	
runScript	Creates a new "script" tag in the body of the current test window,
runScriptAndWait	and adds the specified text into the body of the command.
(script, placeholderValue*)	<pre>script - if starts by // (meaning comment), the script won't be</pre>
	executed by the IDE, but will be executed by the AuraPlayer
	ServiceManager.
	placeholderValue – all \${something} tokens in 'script' will be
	replaced with 'placeholderValue' before running the script.
select   selectAndWait	Selects an option on <select> element.</select>
(selectLocator,	Currently we support selection of values identified by
optionLocator)	<b>label=</b> type only.
selectWindow	Selects a popup/window using a window locator; once a window
(name)	has been selected, all commands go to that window.
	To select the main window again, null as the target.
storeChecked	Get whether a toggle-button (checkbox/radio) is checked, and
(locator, variableName)	store it in an output parameter called 'variableName'.
storeHtmlSource	Returns the entire HTML source between the opening and closing
(variableName)	"html" tags.
	It is saved in the 'temp' folder, in a file called 'variableName'.
storeText	Get the text of an element, and store it in an output parameter
(locator, variableName)	called 'variableName'.
submitAndWait	Submit the specified form.
(locator, optionalAction*)	optionalAction - an optional JavaScript code to be injected as the
	'action' attribute of the form.
type	Sets the value of an input field, as though you typed it in.
(locator, value)	



uncheck   uncheckAndWait (locator)	Same as <i>check   checkAndWait</i> – but the generated input parameter is initialized to <i>false</i> value.
waitForElementPresent (locator, shouldWait*)	Waits until the specified element appears, or a timeout is reached. shouldWait – playback will ignore this command if set to 'false' (in order to save performance).
waitForPageToLoad (timeout)	Waits for a new page to load (with max wait time).

# **DB Services**

The 'DB Service' executes database queries/operations, and returns the result set. Your ServiceManager operates with DB Services in a very similar way to Oracle Form Services.

### **Creating DB Services**

1. On the Service Manager page, click on "Add"  $\rightarrow$  "DB service".



2. The "Service Editor" will be opened, with dedicated database configuration view.



Service Manager / S	ervice Configuration / New	
	1	Create Service Bac
Service Details		~
Service Name	dbService	
Description	description	
DB Host:Port	host	: port
DB Name / SID	Name / SID	
DB Connector	Select type	
Query	e.g. SELECT * FROM customers Query is required	
	${f O}$ Refer to input parameters in query by using ${\rm Parameter-name}$ placed	holders.

- 3. Enter a service name and description (optional).
- 4. Configure the database connection:
  - Fill in the **hostname** of the database server.
  - Fill in the database **port**.
  - Default ports are: 1521 for Oracle databases, 3306 for MySQL, and 5432 for PostgreSQL.
  - Enter the database Name or SID.
  - Select the database **type**.
- 5. Click on "**Test connection**" to ensure that your connection is properly configured.
- 6. A username/password dialog will open, enter your credentials to the database server and confirm.

	DB Authentication		
Username	mia		
Password			
<ul> <li>Your credentials will be stored as input parameters, but you may clear them after closing the dialog.</li> <li>OK Cancel</li> </ul>			

Your username and password will be automatically stored as the values of the corresponding input parameters. You may clear or hide them if you do not wish them to be exposed in your service.



Success/failure status will be displayed in popup message.
 If the connection failed, do not continue any further - check your DB configuration and retry.



8. Enter a "Query" as described in the following section.

## **DB Queries / Statements**

The last field under "Service details" of a Database Service is the "Query" field.

Enter a valid SQL statement.

You should enter a single statement (without semicolon). In order to execute multiple commands, wrap them in function or procedure and call it from here.

Examples:

- SELECT STATE FROM CITIES WHERE POPULATION < 1000
- INSERT INTO CITIES (STATE, CITY, POPULATION, WHITE, BLACK, HISPANIC, ASIAN, OTHER) VALUES ('DE', ' Muenchen', 1300000, 0, 0, 0, 0, 0)

#### **Dynamic queries (injecting input parameters)**

Your query may vary according to input provided to the service.

For example, assuming that you are managing a CITIES table, and your service updates the population of the city, then the city name and population are probably input provided by the user executing the service. Meaning, instead of writing:

- UPDATE CITIES SET POPULATION=40000 WHERE CITY='Muenchen'

You may write:

- UPDATE CITIES SET POPULATION=\${population} WHERE CITY='\${city}'

Now, the **population** and **city** values are expected to be found in the input parameters of the service. You should add an input parameter named **population** and and input parameter named **city** to the input parameters table:



	Visible	Name	Label	Default Value	Actions
÷	$\checkmark$	username	username	mia	
÷	<ul> <li></li> </ul>	password	password		T
÷	<ul> <li>Image: A second s</li></ul>	city	city	Muenchen	

Click on the add button 🧕 , to append a new row at the bottom of the input parameters table.

+	city		city	Muenchen	
	name Parameter	name is required	label	default value	

Fill in the name (must match to the query; use **population** in our example), label and default value for parameter. Finally click on the save icon.

You may now add more input parameters – add **city** for this example.

#### Query examples

SELECT	<pre>SELECT * FROM \${table} WHERE POPULATION &lt; \${max_population}</pre>
UPDATE	UPDATE CITIES SET POPULATION =0 WHERE CITY='\${city}'
INSERT	INSERT INTO CITIES (STATE, CITY, POPULATION, WHITE, BLACK, HISPANIC,
	ASIAN, OTHER) VALUES ('\${state}', '\${city}', \${population}, 0, 0, 0, 0, 0)
DELETE	DELETE FROM CITIES WHERE state = '\${state}'
Calling a function	SELECT GET_POPULATION('\${city}') as result FROM dual
Calling a function	SELECT GET_POPULATION('\${city}') as result FROM dual Since the name of the selected value is dynamic, we have to add "as
Calling a function	SELECT GET_POPULATION('\${city}') as result FROM dual Since the name of the selected value is dynamic, we have to add "as result" (or any other name) so the output will be returned by that name and we
Calling a function	SELECT GET_POPULATION('\${city}') as result FROM dual Since the name of the selected value is dynamic, we have to add "as result" (or any other name) so the output will be returned by that name and we would be able to capture the value with an output parameter named "result".
Calling a function Executing a	SELECT GET_POPULATION('\${city}') as result FROM dual Since the name of the selected value is dynamic, we have to add "as result" (or any other name) so the output will be returned by that name and we would be able to capture the value with an output parameter named "result". EXECUTE INCREASE_POPULATION('\${city}')

#### **Testing Queries and Acquiring Output Parameters**

After configuring your database connection and query, click on the "**Get output**" button to execute your query for the first time.

In addition to executing the query, it also adds any table columns returned from the query to the "**Output parameters**" table.

► Get output



Your query may not return any values, and no output parameters will be added (except from the traditional *Error, PopupMessages,* and *StatusBarMessages*).

The number of new output parameters added to the table will be displayed in a popup message:



You may also manually add, hide, or delete any output parameters.

Finish by saving the service.

#### **Viewing recorded DB services**

All services are listed in the 'Services' tab from the left toolbar. The 'Type' column in the services list, distinguishes **DB** Services from other services.

#### **Executing, Editing and Using DB Services**

All actions are to be performed in the same manner as with the regular Oracle Form Services. Consult this manual for further instructions.

#### **DB Services output**

The DB Services maintain the same response structure as the Oracle Form Services:





The contents of the 3 special output parameters are:

- Error DB errors, if any.
- StatusBarMessages Number of affected rows if the executed statement has no output.

## JavaScript Services

The 'JavaScript Service' executes JavaScript code.

Your ServiceManager operates with JavaScript Services in a very similar way to other types of services.

## **Creating JavaScript Services**

1. On the Service Manager page, click on "Add"  $\rightarrow$  "Javascript service".



 The "Service Editor" will be opened. In the "JavaScript Scenario" section, you enter your JavaScript code.

Javascript Scenario	*
1 · return { 2 };	
<ul> <li>① Add single return (); statement to return output from your code.</li> <li>① Every key of the returned object must appear in new line.</li> <li>① Right click in editor to open Auto Complete suggestions.</li> </ul>	Unlock edit

First, click on the 'Unlock edit' button to open the editor for editing. When editing, you cannot add/remove input parameters nor view output parameters.



Once you done writing your code, click on the 'Lock edit' button to configure service parameters. The editor will be disabled and grayed out again.

## **Adding Input Parameters**

To edit input parameters, make sure that the JavaScript editor is locked.

Use 'Add'/'Delete' buttons in the "Input Parameters" section to create/delete input parameters.

put P	ramete	ers			
	Visible	Name	Label	Default Value	Actions
÷	$\checkmark$	input1	input1	1st value	
÷	$\checkmark$	input2	input2	2nd value	•

To refer input parameters in the JavaScript code, right click in the editor, and select the input parameter from the context menu.

Javas	cript Scenario			
	(	Right click here)		
1	var input1 = •			
2	return {	Refer input •	input1	
▲ 4	};	Call WebService	input2	
		Reset value		

An appropriate expression will be inserted at the current cursor position. In runtime, this expression will be evaluated as the string value of the parameter.



## **Setting Output Parameters**

The JavaScript code returns output using a single 'return' statement at the topmost level (i.e. from a single non-nested 'return'). The returned value must always be a valid JavaScript object.





Each property in the returned object evaluates to output parameter: the name of the parameter will be the key of the property, and its value will be assigned accordingly after execution. Every key in the returned object must appear in new line.

The following code declares two output parameters: output1, output2.

Javascript Scenario				
<pre>i 1 var input1 = ServiceManager.getInput("input1") </pre>				
2 3 return {				
4 'output1': 'This is ' + input1, 5 'output2': 84				
6 };				

Output parameters are automatically derived from the 'return' statement once you lock the editor.

Outpu	t Parame	eters				*
	Visible	Name	Label	Multi     Record	Actions	
÷	~	output1	label		05	
÷	~	output2	label		<b>Q</b> \$	

If you wish to return **array** value, you should mark the generated output parameter as **multi-record**. Returning array value into non-multi-record output parameter will result in assignment of the first array element to the parameter.



In the same matter you may also return values to the special "StatusBarMessages" and "PopupMessages" output parameters. However, returning value to the "Error" field is done by throwing an error – see "Throwing errors" section bellow.





#### **Calling other services**

The JavaScript service may invoke other web services on your ServiceManager, and read their result. **Right Click** (in the editor)  $\rightarrow$  **Call WebService**, to insert a template for web service call:

Javascript Scenario				
<pre>1 var response = ServiceManager.callWebService("mcs_getCustomer", { 2     "MAIN_USERNAME_0": "MIA", 3     "5_CUSTOMER_ID_0": 201 4 });</pre>				
<pre>5 6 console.log(JSON.stringify(response)); 7 return { 8</pre>				

An expression that calls the 'callWebService' method is inserted. Replace the 1<sup>st</sup> argument with the name of the service to call, and the 2<sup>nd</sup> argument with an object whose properties are the input names and assigned values.

The returned value is a JSON object, in the exact structure as a service's JSON response. You may address each field by using the full JSON path, such as: response.Response.mcs\_getCustomerElements.NameRequired.

#### **Throwing Errors**

Your JavaScript service may exit abnormally by throwing an error.

Use a 'throw' command to terminate execution and report error in the 'Error' output parameter.



## Logging and Debugging



Logging to console using 'console.log()' (or similar), will append the message to the system log (Admin  $\rightarrow$  View Log).



Additionally, it will append the message to the 'StatusBarMessages' output parameter, if its value is not set in the 'return' statement (see "Setting output parameters" section above). To suppress propagation of console message to 'StatusBarMessages', assign value to it in the 'return' clause – an empty string may be used if necessary.

Service test result for jsService
1 - 1
2 "success": true,
3 - "Response": {
<pre>4 "jsServiceSuccess": true,</pre>
5 - "jsServiceElements": {
6 "output1": "This is 1st",
7 "output2": 84
8 },
9 • "jsServiceMessage": {
10 "Error": "",
11 "StatusBarMessages": "input1 is 1st;"
12 }
13 }
14 }

Supported console commands are: console.log, console.error, console.warn, console.debug, console.info, console.trace.

The system log will preserve log level (console.log will be logged as console.debug). 'StatusBarMessages' (if not overridden), will store all log messages regardless of their severity.

# **Support**

#### **Getting Support**

Please feel free to visit: support.auraplayer.com to open a service request.

Or email <a href="mailto:support@auraplayer.com">support@auraplayer.com</a>

#### Thank you for choosing AuraPlayer.