

# **Creating and Consuming Pagelets with the Oracle WebCenter Pagelet Producer**

The goal of this module is to teach you about functionality introduced in Oracle WebCenter 11g R1 (11.1.1.4), commonly referred to as PS3. The product will be rebranded as Oracle WebCenter Portal in PS5 (11.1.1.6).

For more information, visit Oracle WebCenter Portal (<http://www.oracle.com/us/products/middleware/webcenter/index.html>) on the Oracle Technology Network (OTN).

# Objectives

After completing this module, you should be able to:

- Use the Pagelet Producer to create, edit, delete, and deploy resources and pagelets
- Add pagelets to web applications in a variety of scenarios, including standard web pages, portal pages, Oracle JDeveloper and Oracle WebCenter Spaces

# **1: Introduction: The Oracle WebCenter Pagelet Producer**

# What Is the Oracle WebCenter Pagelet Producer?

The Oracle WebCenter Pagelet Producer brokers transactions between client computers and external resources. This configuration allows you to:

- **Proxy** internal web applications to external addresses
- Manage **authentication**
- **Expose functionality** from internal applications that are on a private network or not readily accessible via web technology
- **Transform** proxied web applications, including URL-rewriting
- **Add, remove and modify functionality** from the external resource at runtime
- **Consume WSRP portlets** in Oracle WebCenter applications

Benefits to this configuration include **dynamic functionality** and **personalization, security** and **performance**.

# What Is the Oracle WebCenter Pagelet Producer?

The Oracle WebCenter Pagelet Producer allows you to publish content from back-end applications via **pagelets**.

A pagelet is a **reusable UI component** that can run on any web page or web application.

The pagelet development model is very lightweight and uses **standard web technology** (HTML, JavaScript and AJAX) instead of specific portlet standards.

Using the Pagelet Producer, you can:

- Add content and functionality from **existing back-end applications** to any web page on any platform
- **Integrate external content** into any portal

## **2. Using the Pagelet Producer Console: Creating and Configuring Resources, Pagelets, Injectors and Parsers**

# Pagelet Producer Console

The **Pagelet Producer Console** is a browser-based administration UI used to register resources and pagelets, manage proxy and transformation settings, and more.

To launch the Pagelet Producer Console, navigate to `http://host_name:port_number/pageletadmin/` in your browser. The console allows you to configure the Pagelet Producer and create the following objects:

- **Resource**
- **Pagelet**
- **Injector**
- **Parser**



# Configuring Pagelet Producer Settings

The **Settings** section in the Pagelet Producer Console includes important options that affect all resources and pagelets.

- **Logging:** Set logging levels for individual Pagelet Producer components.
- **Proxy:** Configure HTTP proxy access (URL, user name and password) and define a semicolon-separated list of URLs that will not be proxied.
- **Transform:** Define the path to the credential vault provider and configure secure and insecure ports.

# Creating a Resource

To create a new resource, select **Resources** from the dropdown list, choose any existing resource in the navigation pane, and click the **Create** icon (“+”) in the toolbar.

**ORACLE** WebCenter Pagelet Producer



You will be prompted to choose a producer type:



- **web**: Standard web resources comprised of HTML, AJAX and JavaScript
- **csp**: WebCenter Interaction (WCI) resources
- **ccf**: WSRP and JPDK resources

# Resource Settings: General

**Source URL:** The URL to the internal web application.

**Destination URL:** A relative URL that defines the endpoint in the Pagelet Producer proxy URL space where the proxied content will be presented.

**URL Rewriting:** The Pagelet Producer will rewrite all URLs in the proxied application that begin with the source URL prefix to point to the destination URL prefix.

The screenshot shows the Oracle WebCenter Pagelet Producer interface. The title bar reads "ORACLE WebCenter Pagelet Producer". Below it, there is a "Navigator" pane on the left and a "General ?" settings pane on the right. The "General ?" pane contains the following settings:

- \* Name: ModuleResource
- Description: New resource for the Creating and Consuming Pagelets course.
- \* Source URL: http://moduleServer/
- Source Timeout (in seconds): 30
- \* Destination URL: module/
- URL Rewriting:
- DHTML Rewriting:

Yellow callout boxes with arrows point from the text blocks to the corresponding fields in the settings pane: "Source URL" points to the Source URL field, "Destination URL" points to the Destination URL field, and "URL Rewriting" points to the URL Rewriting checkbox.

# Oracle WebCenter Pagelet Producer Proxy Warnings and Best Practices

- Configure the proxy carefully so users are not given direct access to unprotected private content.
- The Pagelet Producer transforms any URLs that use the Internal URL prefix configured for the resource unless **URL Rewriting** is deselected in the Resource editor.
- Static images and other binary data should be stored on a separate server so they are not proxied.
- To prevent unintentional transformation, encode all headers that are URLs.
  - In JSP, use `response.encodeURL()`
  - In .NET, use `HttpUtility.UrlEncode`

# Resource Settings: Policy

**Policy** settings limit access to the proxied resource to specific roles within Oracle WebCenter, configured in Oracle WebLogic Server.

- Enter the roles that should be allowed unauthenticated access to the resource. All other users will be prompted for credentials.
- If no roles are entered on this page, the resource will allow unauthenticated (“anonymous”) access to all users.



# Resource Settings: Autologin Form Login

The Oracle WebCenter Pagelet Producer can automatically log in to proxied resources through HTML forms.

**ORACLE** WebCenter Pagelet Producer

The screenshot shows the Oracle WebCenter Pagelet Producer interface. On the left is a Navigator pane with a tree view containing 'welcome\_resource', 'login\_resource', 'pagelet\_api', and 'ModuleResource'. Under 'ModuleResource', there are sub-items: 'General', 'CSP', 'Policy', 'Autologin' (highlighted), 'Headers', 'Pagelets', 'Injectors', and 'Parsers'. The main area is titled 'Remote Autologin ?' and contains several sections:

- Basic Login**: A collapsed section.
- Form Login**: An expanded section containing:
  - Login Form Identification**: A dropdown menu set to 'URL', a text input field containing 'http://moduleServer/login.asp', and a button labeled 'Automatically Detect Form Fields'.
  - Form Submit Location**: A dropdown menu set to 'None'.
  - Form Fields**: A table with columns 'Field Name', 'Source', 'Value', and 'Additional'. The table is currently empty, displaying 'No data to display'. There are '+ Create' and '- Delete' buttons to the right of the table.
- NTLM Login**: A collapsed section.

Three callout boxes provide instructions:

- A yellow callout box points to the 'URL' dropdown and the text input field, containing the text: "Enter the URL or RegEx to access the login form."
- A yellow callout box points to the 'Automatically Detect Form Fields' button, containing the text: "Enter the URL or RegEx to access the submit form."
- A yellow callout box points to the 'Form Fields' table, containing the text: "Map form fields manually or use auto-detect."

# Resource Settings: Autologin Basic and NTLM Login

To use basic or NTLM login, define the Username and Password keys to be used by the Pagelet Producer.

You must define an authentication source for all login fields:

- **Static:** Uses the specified value for all users.
- **Profile:** Uses properties from the user's Oracle WebCenter profile to supply credential data.
- **Vault and Shared Vault:** Prompts for credentials the first time the resource is accessed and uses stored credentials for subsequent access.

The screenshot shows the 'Resource Settings' interface for 'Autologin'. It features three main sections: 'Basic Login', 'Form Login', and 'NTLM Login'. The 'NTLM Login' section is expanded, revealing two sub-sections: 'Username' and 'Password'. In the 'Username' section, there is a dropdown menu currently set to 'Profile' and a text input field containing the value 'username'. In the 'Password' section, there is a dropdown menu with a list of options: 'None', 'Static', 'Profile', 'User Vault', and 'Shared Vault'. A mouse cursor is positioned over the 'None' option, which has triggered a tooltip bubble containing the text 'Select password source'.

# Resource Settings: Headers

You can customize which headers are passed to the back-end application and from the back-end site to the browser.

To block Request or Response header elements, click **Create** and specify the header name.

**ORACLE** WebCenter Pagelet Producer

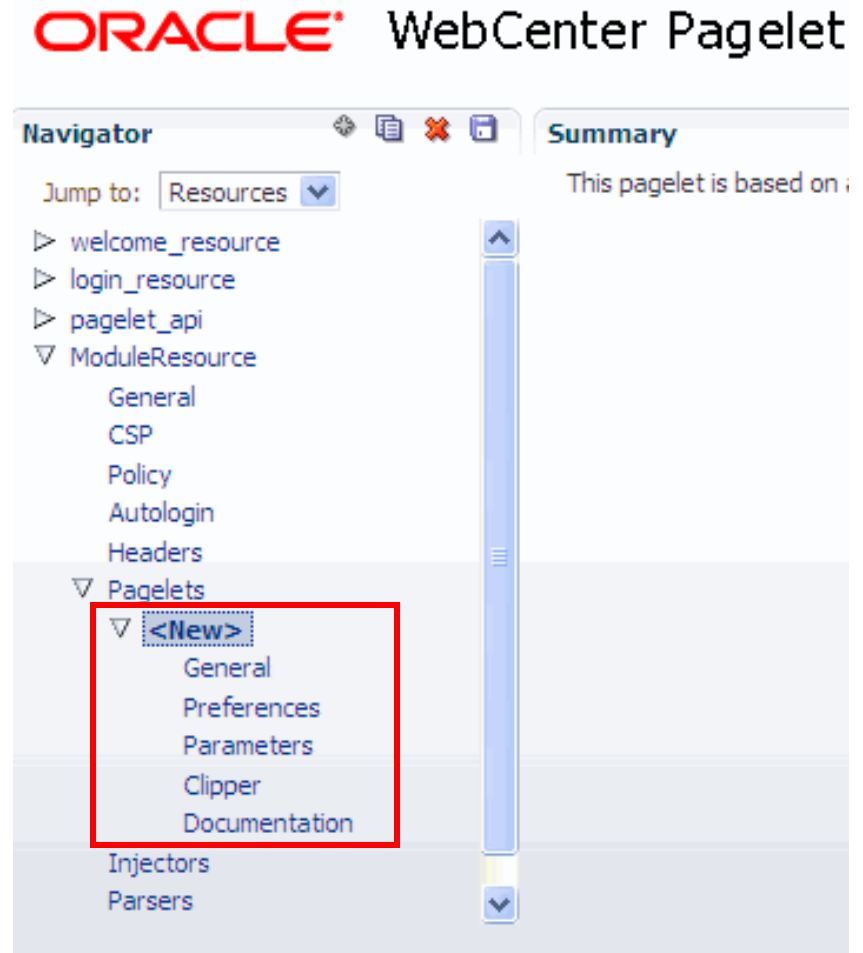
The screenshot displays the Oracle WebCenter Pagelet Producer interface. On the left is a 'Navigator' pane with a tree view containing 'welcome\_resource', 'login\_resource', 'pagelet\_api', and 'ModuleResource'. Under 'ModuleResource', there are sub-items: 'General', 'CSP', 'Policy', 'Autologin', 'Headers' (highlighted), 'Pagelets', 'Injectors', and 'Parsers'. The main area shows two panels: 'Request Headers To Drop' and 'Response Headers To Drop'. The 'Request Headers To Drop' panel has a '+ Create' button (highlighted with a red box) and a '- Delete' button. Below the buttons is a table with one entry: '\* Request Headers' and 'UnneededSSOFilter'. The 'Response Headers To Drop' panel also has '+ Create' and '- Delete' buttons, and its table is empty, displaying 'No data to display'. A 'Logout' link is visible in the top right corner.



# Creating Pagelets

Once you have configured a resource, you can create pagelets within the resource.

To create a pagelet, select the Pagelets section under the resource and click the Create icon (“+”) in the toolbar.



# Pagelet Settings: General

## ORACLE WebCenter Pagelet Producer

**URL Suffix:** The *relative* path to the pagelet. Do not include the Source URL prefix entered for the resource.

**Refresh Inline** enables refreshing the pagelet without reloading the page that hosts the pagelet.

The screenshot displays the 'General' configuration page for a pagelet. The left sidebar shows a tree view with 'Pagelets' expanded to 'WebPagelet', where 'General' is selected. The main panel contains the following settings:

- \* Name: WebPagelet
- \* Library: Module
- \* URL Suffix: webpagelet.html
- Refresh Inline:
- Description: (empty text area)

# Pagelet Settings: Preferences

On the **Preferences** page, enter *relative* URLs to any preference pages required by the pagelet.

Choose the preference level for the preference page:

- **Global** preferences apply to all instances of the pagelet in all pages.
- **Customize** preferences apply to a single instance of the pagelet in all users' pages.
- **Personalize** preferences apply to a single instance of the pagelet in a single user's page.

# Pagelet Settings: Preferences and Parameters

On the **Parameters** page, enter the **Payload Schema** or **Parameters** that should be passed to the pagelet.

For parameters, specify the name, whether or not the parameter is required, the data type, and the transport type:

- **Request Parameters** are added to the HTTP request and apply to all instances of a pagelet.
- **Administrative, Community and Pagelet Preferences** are used in Oracle WebCenter Interaction.

# Pagelet Settings: Clipper

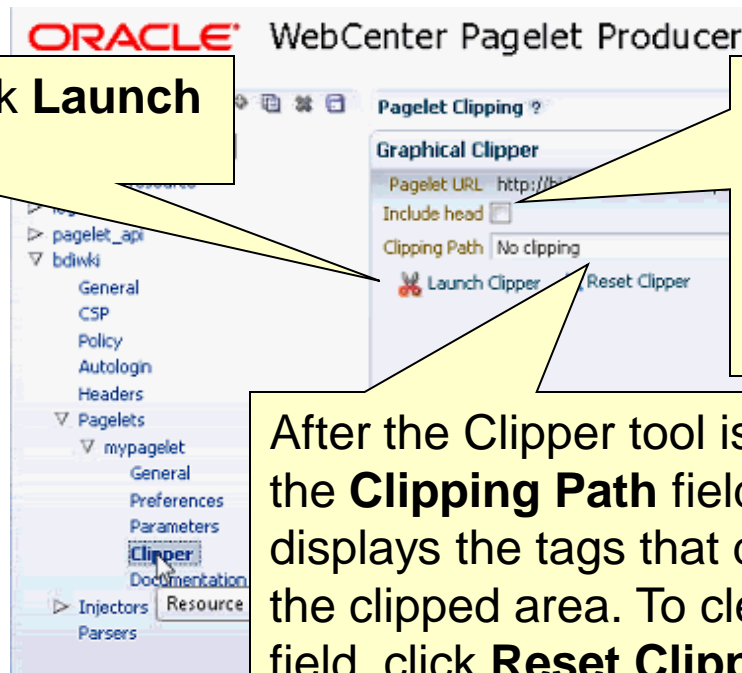
The **Clipper** allows you to create a pagelet by clipping a portion of a web page in a proxied application. **Clipping is done at runtime**, so clipped content is retrieved each time the pagelet is displayed.

The **Graphical Clipper** allows you to select page content using a mouse-driven graphical tool. The **Advanced Clipper** allows you to define HTML tag names and attributes to describe the clipped.

To start clipping, click **Launch Clipper**.

Use the **Include head** option to retrieve the CSS, JavaScript and other declarations from the `<head>` of the source page.

After the Clipper tool is run, the **Clipping Path** field displays the tags that define the clipped area. To clear the field, click **Reset Clipper**.



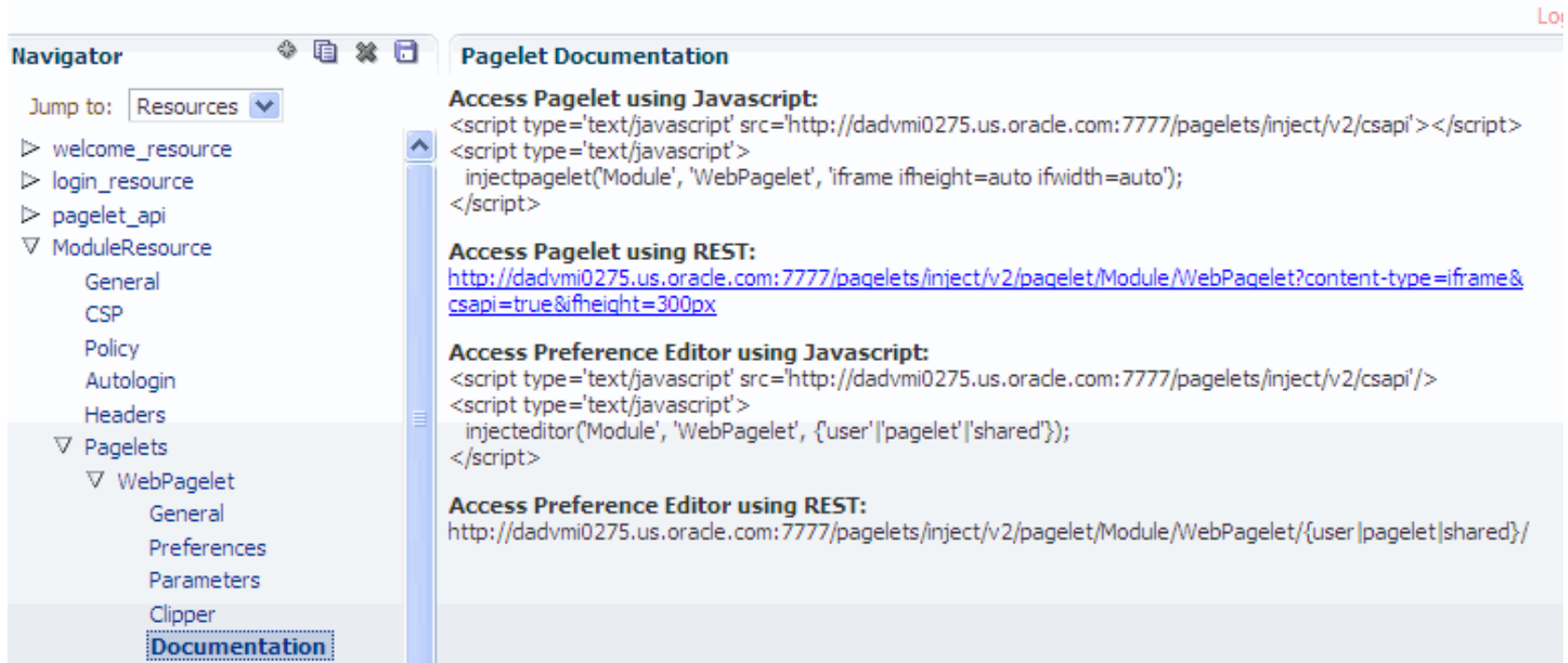
# Pagelet Clipper Best Practices

- To modify the clipped content, use an injector.
- If the back-end resource is accessed over HTTPS, the Pagelet Producer Console must be accessed over a secure port.
- If the source page requires a login, configure Autologin settings before using the clipper.
- The Suffix URL entered on the General page for the pagelet must reflect the final URL loaded by the browser after following all redirects.

# Pagelet Settings: Documentation

The **Documentation** page displays code to insert the pagelet and preference editor (if configured) into a web page using either the JavaScript or REST.

**ORACLE** WebCenter Pagelet Producer



**Pagelet Documentation**

**Access Pagelet using Javascript:**

```
<script type='text/javascript' src='http://dadvmi0275.us.oracle.com:7777/pagelets/inject/v2/csapi'></script>
<script type='text/javascript'>
  injectpagelet('Module', 'WebPagelet', 'iframe ifheight=auto ifwidth=auto');
</script>
```

**Access Pagelet using REST:**

<http://dadvmi0275.us.oracle.com:7777/pagelets/inject/v2/pagelet/Module/WebPagelet?content-type=iframe&csapi=true&ifheight=300px>

**Access Preference Editor using Javascript:**

```
<script type='text/javascript' src='http://dadvmi0275.us.oracle.com:7777/pagelets/inject/v2/csapi'>
<script type='text/javascript'>
  injecteditor('Module', 'WebPagelet', '{user}|pagelet|shared'});
</script>
```

**Access Preference Editor using REST:**

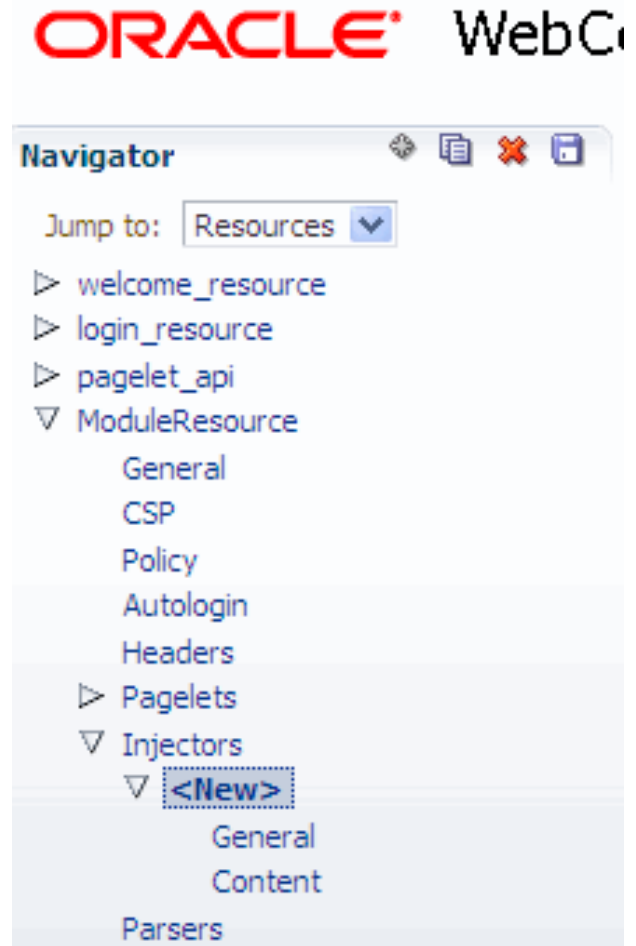
<http://dadvmi0275.us.oracle.com:7777/pagelets/inject/v2/pagelet/Module/WebPagelet/{user}|pagelet|shared}/>

# Creating Injectors

**Injectors** modify content at runtime, and insert content into a specified location in a proxied resource page.

The content may be **any text**, including HTML, CSS, JavaScript, and pagelet declarations.

To create an injector, select Injectors under the appropriate resource and click the Create icon (“+”) in the toolbar.





# Injector Settings: General

**URL Filter:** Applies the injector only to URLs within the resource that begin with the specified text.

**MIME Filter:** Restricts the injector to specific types of content (text/HTML, text/CSS).

## Inject Location:

- **Top** inserts the content first in the page.
- **Bottom** inserts the content last in the page.
- **Before/After/Replace:** Inserts the content into the page relative to the entry specified in the provided field.

## Oracle Pagelet Producer

General ?

\* Name

URL Filter

MIME Filter

**Inject Location**

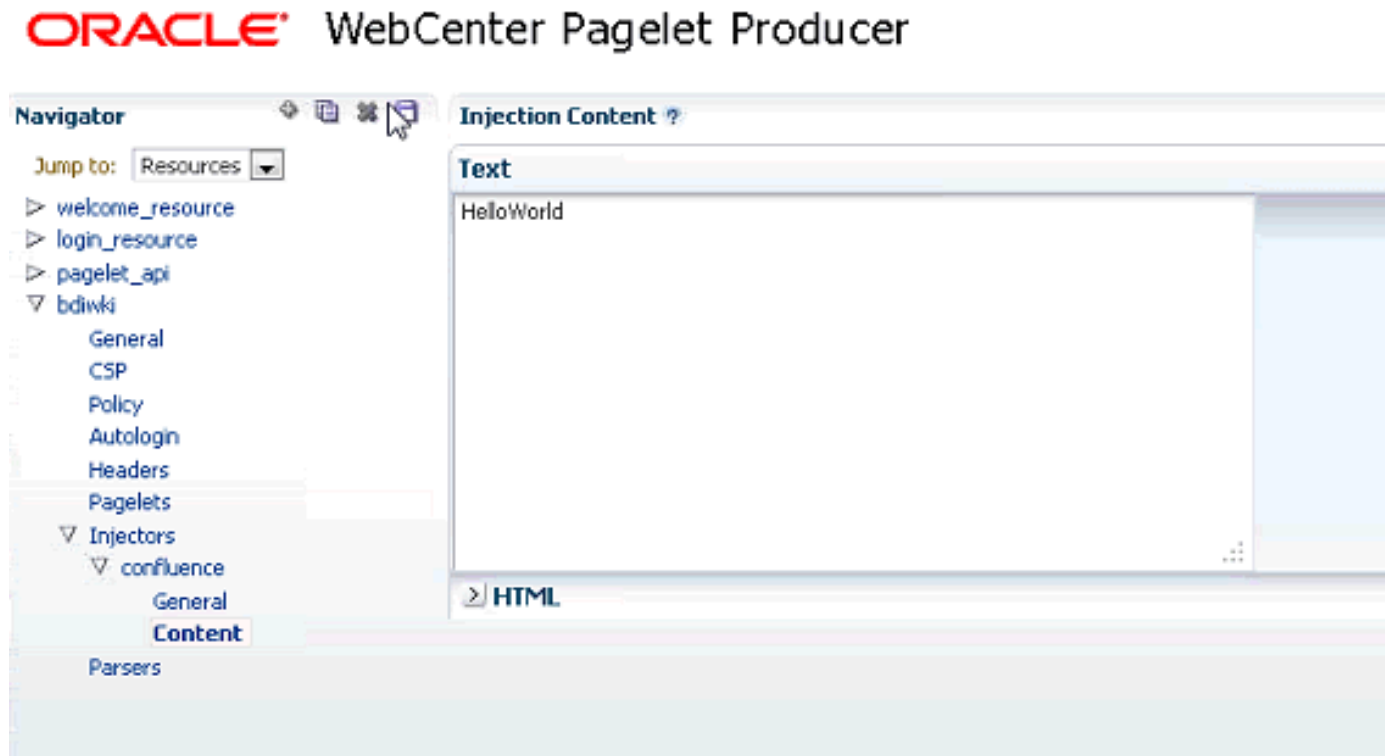
Top  Bottom

Before  After  Replace

Ignore case  Enclose tag

# Injector Settings: Content

Content to be injected can be any text, including HTML, CSS, JavaScript, and pagelet declarations.



# Creating Parsers

ORACLE® WebC

Custom **Parsers** allow you to supplement or change built-in logic for parsing content and finding URLs.

To create a parser, select **Parsers** under the appropriate resource and click the **Create** icon in the toolbar.



# Creating Parsers

**Parsers are implemented at runtime**, allowing you to modify content dynamically for consumption in a pagelet.

### General

Name

URL Filter

Use Base Parser

### Fragment Locations

[+ Create](#) [✖ Delete](#)

Regular Expression	Fragment Type
<code>))?(?&lt;p0&gt;[^\?#]*" + @"(?&lt;q1&gt; ?(?&lt;q0&gt;[^\#]*)?)?" + @"(?&lt;f1&gt;#(?&lt;f0&gt;.*)?)?";</code>	Static URL

## 3. Consuming WSRP Portlets

# Consuming WSRP Portlets (Patch 11809215)

The Pagelet Producer can expose WSRP portlets as pagelets for use in WebCenter Portal applications, WebCenter Spaces, and third-party portals.

**Note:** To use this feature in the PS3 version of Oracle WebCenter, install **patch 11809215**.

When you register a WSRP producer in the Pagelet Producer Console, the new resource will be populated automatically with pagelets to represent the portlets associated with the WSRP endpoint.

# Consuming WSRP Portlets

## Register WSRP Endpoint (WSDL)

In the Pagelet Producer Console, choose **Producers** from the dropdown menu.



Click **Register**.



# Consuming WSRP Portlets Configuration Settings

**ORACLE** WebCenter Pagelet Producer

**Navigator**  
Jump to: Producers

**Register Portlet Producer**

**Name And Type**  
\* Producer Name: pageletmodule  
Producer Type:  WSRP Producer  Oracle PDK-Java Producer

**Portlet Producer URL**  
\* WSDL URL: http://portlet.uk.oracle.com:9999/wsrp-testbridge/portlets/wsrp2?WSDL  
Use Proxy?   
Proxy Host:   
Proxy Port:

**Advanced Configuration**  
Specify additional (optional) information.  
Default Execution Timeout (Seconds):

**Security**  
Select the token profile used for authentication with this WSRP producer.  
Token Profile: None

Buttons: Test, Ok, Cancel, Logout

**Callout 1:** Choose **WSRP Producer** as the **Producer Type**.

**Callout 2:** Enter the **WSDL URL** for the WSRP endpoint.

**Callout 3:** If an HTTP proxy is required, select the **Use Proxy?** option and enter the host name and port for the proxy server.

**Callout 4:** To confirm that the configuration is correct, click **Test**.



# Consuming WSRP Portlets

The new producer is displayed with a list of the imported pagelets based on the WSRP portlets from the WSDL.

**ORACLE** WebCenter Pagelet Producer

Logout

Navigator



Portlet Producer

Jump to: Producers



Register



Edit



Deregister



Refresh

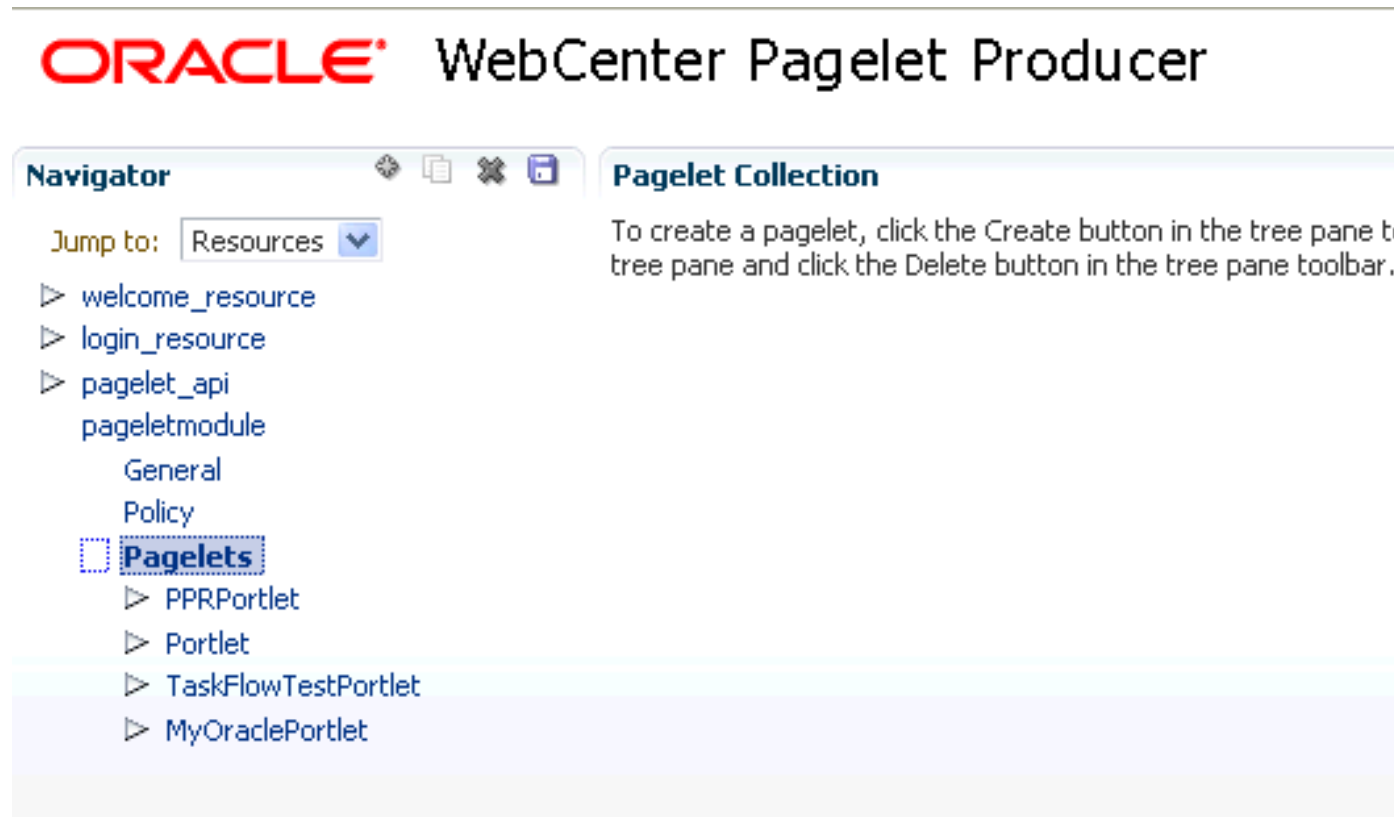


**pageletmodule** (WSRP Producer)

PPRPortlet, Portlet, TaskFlowTestPortlet, MyOraclePortlet

# Consuming WSRP Portlets

When a producer is registered, a new resource is automatically created, and the imported WSRP portlets are imported into the associated Pagelets collection.



**ORACLE** WebCenter Pagelet Producer

**Navigator**

Jump to: Resources ▾

- ▶ welcome\_resource
- ▶ login\_resource
- ▶ pagelet\_api
  - pageletmodule
    - General
    - Policy
    - Pagelets**
      - ▶ PPRPortlet
      - ▶ Portlet
      - ▶ TaskFlowTestPortlet
      - ▶ MyOraclePortlet

**Pagelet Collection**

To create a pagelet, click the Create button in the tree pane toolbar and click the Delete button in the tree pane toolbar.

## 4. Adding Pagelets to Pages and Applications

# Adding a Pagelet to a Page

Once you have deployed a pagelet, you can insert it into any proxied or non-proxied web page using:

- JavaScript
- REST

The code to insert a pagelet into a page is provided on the Documentation page for the pagelet in the Pagelet Producer Console.

You can also drag and drop pagelets onto WebCenter Portal pages using Oracle JDeveloper and in Oracle WebCenter Spaces using Oracle Composer.

# Adding a Pagelet to a Web Page: JavaScript

```
function injectpagelet(library, name, iframe_options,  
payload, params, context_id, element_id,  
is_in_community)
```

The `injectpagelet` interface includes the following attributes:

- **iframe\_options**: Optional. Specifies whether to use an IFRAME. To enable automatic resizing, set `ifheight` and `ifwidth` to 'auto'.
- **payload**: Optional. The XML payload to send with the request.
- **params**: The pagelet parameters in query string format.
- **context\_id**: Optional. The external identifier of the pagelet instance. It must be an integer.
- **element\_id**: Optional. The HTML element ID in which the pagelet content is injected.
- **is\_in\_community**: Optional. Specifies whether the pagelet is on a community (WebCenter Interaction) or group page.

# Adding a Pagelet to a Portal Page: REST

```
http://proxy:port/inject/v2/pagelet/libraryname/pageletname?  
content-type=iframe&csapi=true&ifheight=123px&ifclass=myclass
```

The following parameters are defined for the pagelet inject URL:

- **instanceid**: Optional. The instance ID of the pagelet.
- **content-type**: The return type:
  - **javascript**: Returns injectable code.
  - **html**: Returns pagelet markup with PTPortlet object.
  - **iframe**: Returns an iframe filled with the pagelet content.
- **csapi**: Whether the CSAPI will be included with the pagelet response.
- **onhttperror**: How error codes will be displayed:
  - **comment**: Replaces pagelet with HTML comment error code.
  - **inline**: Replaces pagelet with error code and server error page.
  - **fullpage**: Replaces entire page with HTTP error information.

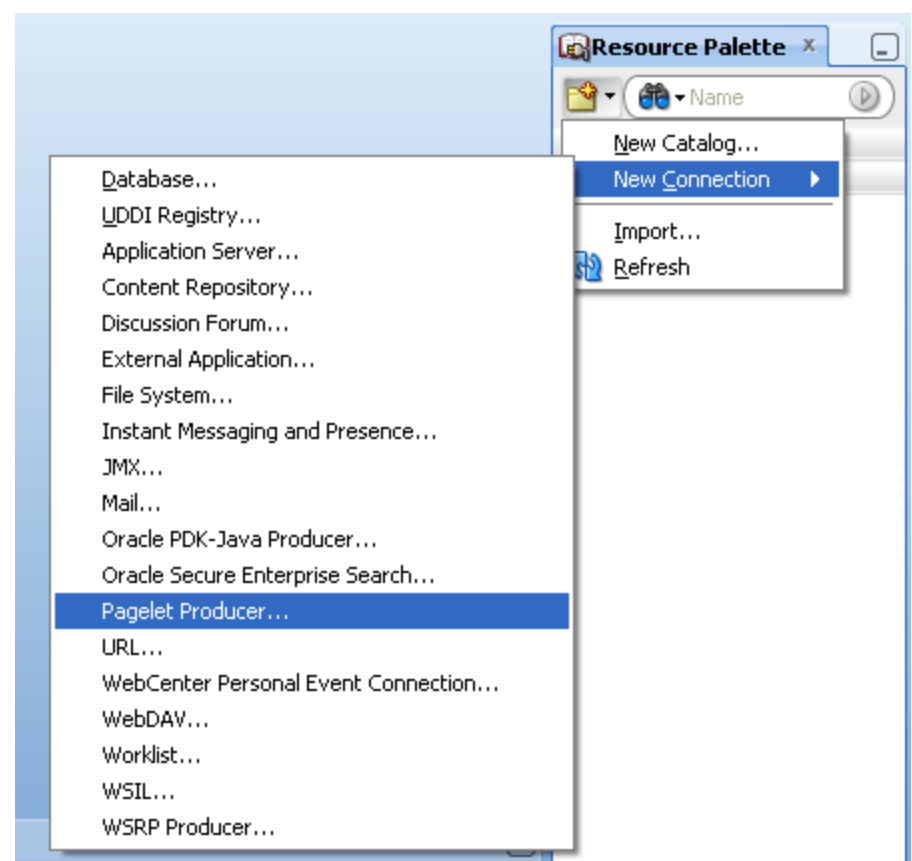
# Adding a Pagelet to a JSF Page in JDeveloper

## Register the Pagelet Producer

Before you can add a pagelet to a JSF page, you must register the Pagelet Producer with your WebCenter Portal application using the Resource Palette.

In the Application Resources panel, right-click **Connections**, choose **New Connection** and then choose **Pagelet Producer**.

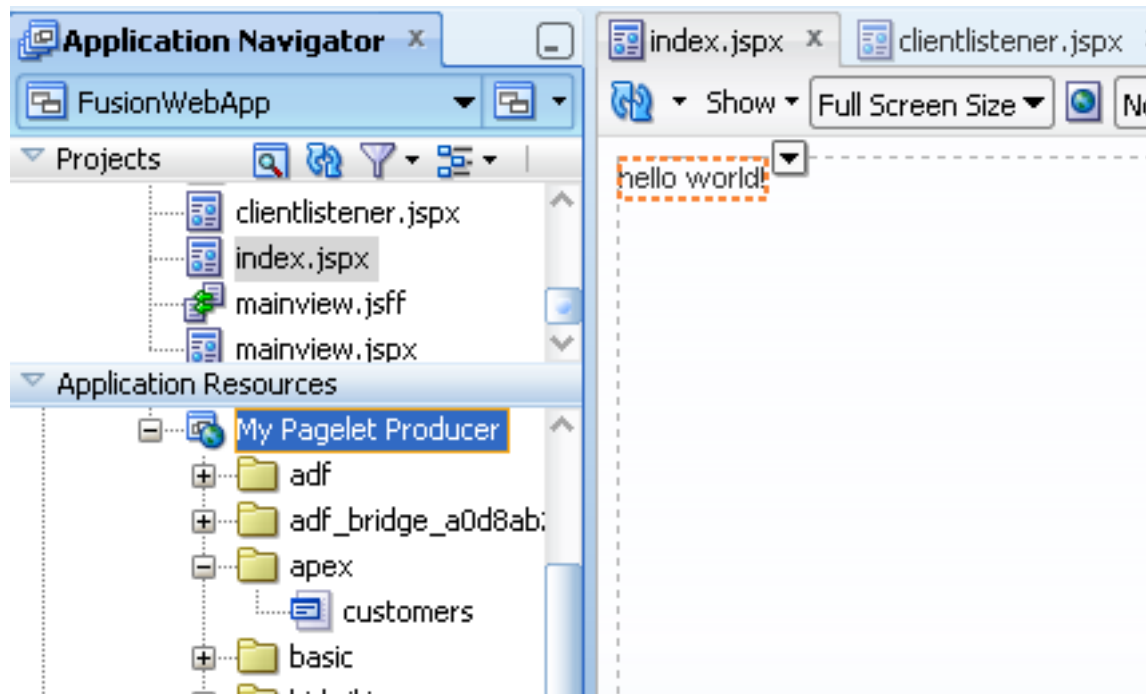
Enter a name for the producer and the URL to the Pagelet Producer (in the format `http://hostname:portnumber`)



# Adding a Pagelet to a JSF Page in JDeveloper

## Adding the Pagelet to the Page

In JDeveloper **Design View**, you can drag and drop pagelets onto a JSF page. In the Application Resources of the Application Navigator, expand the Pagelet Producer to display its contents.

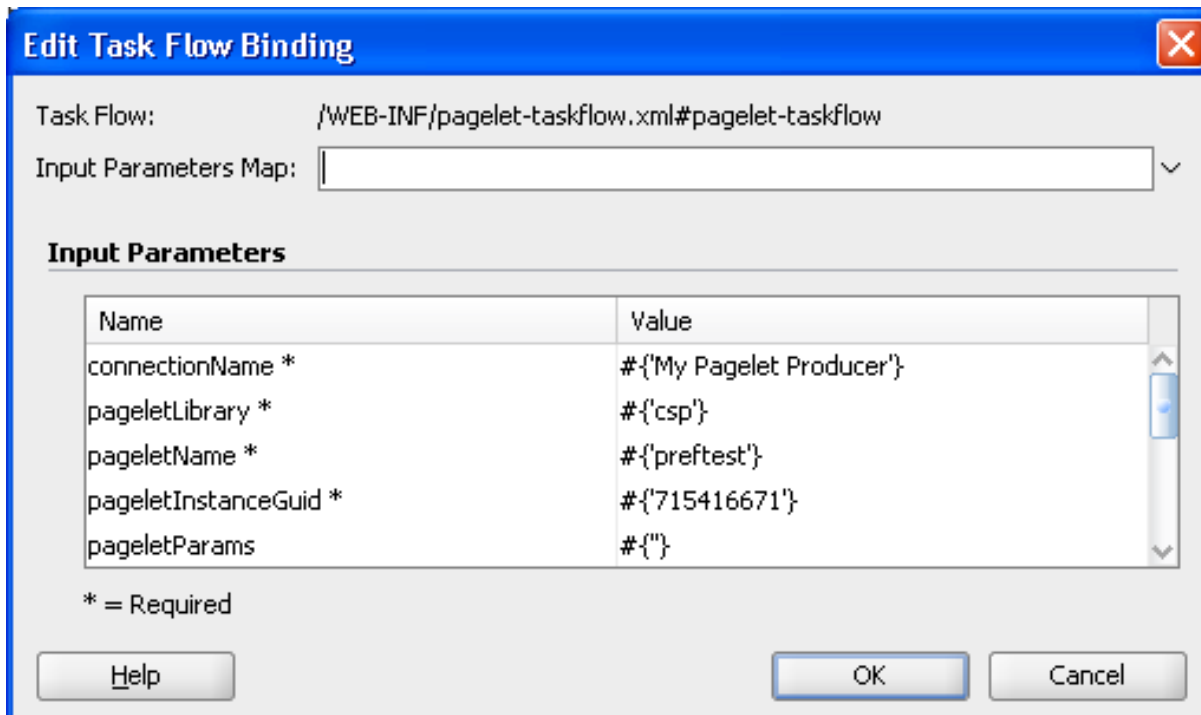




# Adding a Pagelet to a JSF Page in JDeveloper Additional Settings

In the **Add Pagelet to Page** dialog, choose whether to enable IFrame encapsulation.

In the **Edit Task Flow Binding** dialog, change any input parameters as necessary.



**Edit Task Flow Binding**

Task Flow: /WEB-INF/pagelet-taskflow.xml#pagelet-taskflow

Input Parameters Map:

**Input Parameters**

Name	Value
connectionName *	#{'My Pagelet Producer'}
pageletLibrary *	#{'csp'}
pageletName *	#{'preftest'}
pageletInstanceGuid *	#{'715416671'}
pageletParams	#{''}

\* = Required

Buttons: Help, OK, Cancel

# Adding a Pagelet to a Page in WebCenter Spaces

You can add a pagelet to a WebCenter Spaces page using Oracle Composer. All pagelets configured in the Pagelet Producer console are available through Oracle Composer in WebCenter Spaces. Using Composer, you can drag and drop pagelets onto any Spaces page.

For more information, see the section "Adding Resource Catalog Components to Pages" in *Oracle Fusion Middleware Users Guide for Oracle WebCenter Spaces*

# Summary

In this module, you should have learned how to:

- Use the Pagelet Producer to create, edit, delete, and deploy resources and pagelets
- Add pagelets to web applications in a variety of scenarios, including standard web pages, portal pages, Oracle JDeveloper and Oracle WebCenter Spaces