(https://3.bp.blogspot.com/-19TI3sgc-r0/Wc0_cEMNF-I/AAAAAAAADpw/EYCyhNZlfyEj3jat4AGnxcN8-07KPAi6gCLcBGAs/s1600/Portlet%2BFilters%2BInLiferay%2B7%2BDXP.png)

Portlet Filter is used to intercept and manipulate the request and response before it is delivered to the portlet at given life cycle such as action, render and resource

What Is Portlet Filter?

Portlet Filter is used to intercept and manipulate the request and response before it is delivered to the portlet at given life cycle such as action, render and resource

Types Of Filters For A Liferay Portlet

Action Filter

Render Filter

Resource Filter

Base Filter

Which Filter To Use?

Action Filter : To Intercept only The Action Requests

Render Filter : To Intercept only The Render Requests

Resource Filter : To Intercept only The Resource Requests

Base Filter : To Intercept all the http request comes to your portlet.

How To Implement Portlet Filter In Liferay 7 DXP?

Step 1 : Create A Module Project for Portlet

If you want to know how to create a portlet module project (http://www.liferaystack.com/2017/03/creating-portlet-in-liferay-7-dxp_21.html) in Liferay 7 DXP, please refer my earlier blog, Below is my Portlet Controller Class.

Send message

```
package com.liferaystack.portlet;

import java.io.IOException;
import javax.portlet.ActionRequest;
import javax.portlet.ActionResponse;
import javax.portlet.Portlet;
import javax.portlet.PortletException;
import javax.portlet.RenderRequest;
import javax.portlet.RenderResponse;
import javax.portlet.ResourceRequest;
import javax.portlet.ResourceResponse;
import com.liferay.portal.kernel.log.Log;
import com.liferay.portal.kernel.log.LogFactoryUtil;
import com.liferay.portal.kernel.portlet.bridges.mvc.MVCPortlet;
import com.liferaystack.constants.MyPortletKeys;
import org.osgi.service.component.annotations.Component;

/**
 * @author Syed Ali
 */

@Component(
        immediate = true,
        property = {
                "com.liferay.portlet.display-category=LiferayStack",
                "com.liferay.portlet.instanceable=true",
                "javax.portlet.display-name=Portlet Filter - Liferay Stack",
                "javax.portlet.init-param.template-path=/",
                "javax.portlet.init-param.view-template=/view.jsp",
                "javax.portlet.name=" + MyPortletKeys.My,
                "javax.portlet.resource-bundle=content.Language",
                "javax.portlet.security-role-ref=power-user,user"
        },
        service = Portlet.class
)

public class MyPortlet extends MVCPortlet {

        private static Log _log = LogFactoryUtil.getLog(MyPortlet.class);

        @Override
        public void processAction(ActionRequest actionRequest, ActionResponse actionR
                _log.info("processAction.....");
                super.processAction(actionRequest, actionResponse);
        }

        @Override
        public void render(RenderRequest renderRequest, RenderResponse renderResponse
                _log.info("render.....");
                super.render(renderRequest, renderResponse);
        }

        @Override
        public void serveResource(ResourceRequest resourceRequest, ResourceResponse r
                _log.info("serveResource.....");
                _log.info("serveResource.....");
                resourceResponse.getWriter().print("Serve Resource Triggered....");
                //super.serveResource(resourceRequest, resourceResponse);
        }
}
```

Step 2 : Create A Portlet Filter Component Class

You can create Portlet Filter Component Class Using Eclipse, by right clicking on you portlet module project and select select "New Component Class" and Select "Portlet Filter" Component Template, I prefer to create a class manually for now, I Am creating a Class named "MyRenderFilter"

Step 3 : Component Declaration

Declare the Class as an OSGI Component Using the Below Code Snippet

```
import org.osgi.service.component.annotations.Component;
import javax.portlet.filter.PortletFilter;
@Component(immediate = true,
 property = {
  "javax.portlet.name="+MyPortletKeys.My,
 },
 service = PortletFilter.class
)
```

Note : Make sure you give correct **portlet name** and service must be **PortletFilter.class**

Step 4 : Extend The RenderFilter

```
import javax.portlet.filter.RenderFilter;
public class MyRenderFilter implements RenderFilter{}
```

Step 5 : Implement Unimplemented Methods of the Parent Class RenderFilter

```
public void init();
public void destroy();
public void doFilter();
```

Complete MyRenderFilter.java Code

```

package com.liferaystack.filter;

import java.io.IOException;
import javax.portlet.PortletException;
import javax.portlet.RenderRequest;
import javax.portlet.RenderResponse;
import javax.portlet.filter.FilterChain;
import javax.portlet.filter.FilterConfig;
import javax.portlet.filter.PortletFilter;
import javax.portlet.filter.RenderFilter;
import com.liferay.portal.kernel.log.Log;
import com.liferay.portal.kernel.log.LogFactoryUtil;
import com.liferaystack.constants.MyPortletKeys;
import org.osgi.service.component.annotations.Component;

/**
 * @author Syed Ali
 */

@Component(immediate = true,
        property = {
                "javax.portlet.name="+MyPortletKeys.My,
        },
        service = PortletFilter.class
)

public class MyRenderFilter implements RenderFilter{

        private static Log _log = LogFactoryUtil.getLog(MyRenderFilter.class);

        @Override
        public void init(FilterConfig filterConfig) throws PortletException {
                _log.info("init.....");
        }

        @Override
        public void destroy() {
                _log.info("destroy.....");
        }

        @Override
        public void doFilter(RenderRequest request, RenderResponse response, FilterCh
                _log.info("RenderDoFilter.....");
                chain.doFilter(request, response);
        }
}
```

Each time when you Hit a Render URL doFilter() method the MyRenderFilter will be triggered.

Portlet Filter For Action Request

Similarly You can create the filter for all Action Request By extending ActionFilter Class, as shown below

```
package com.liferaystack.filter;

import java.io.IOException;

import javax.portlet.ActionRequest;
import javax.portlet.ActionResponse;
import javax.portlet.PortletException;
import javax.portlet.filter.ActionFilter;
import javax.portlet.filter.FilterChain;
import javax.portlet.filter.FilterConfig;
import javax.portlet.filter.PortletFilter;
import com.liferay.portal.kernel.log.Log;
import com.liferay.portal.kernel.log.LogFactoryUtil;
import com.liferaystack.constants.MyPortletKeys;
import org.osgi.service.component.annotations.Component;

/**
 * @author Syed Ali
 */

@Component(immediate = true,
        property = {
                "javax.portlet.name="+MyPortletKeys.My,
        },
        service = PortletFilter.class
)

public class MyActionFilter implements ActionFilter{

        private static Log _log = LogFactoryUtil.getLog(MyActionFilter.class);

        @Override
        public void init(FilterConfig filterConfig) throws PortletException {
                _log.info("init.....");
        }

        @Override
        public void destroy() {
                _log.info("destroy.....");
        }

        @Override
        public void doFilter(ActionRequest request, ActionResponse response, FilterCh
                _log.info("ActionDoFilter.....");
                chain.doFilter(request, response);
        }
}
```

Portlet Filter For Resource Request

For all Resource request you can extend your class with ResourceFilter Class, as shown below

```
package com.liferaystack.filter;

import java.io.IOException;
import javax.portlet.PortletException;
import javax.portlet.ResourceRequest;
import javax.portlet.ResourceResponse;
import javax.portlet.filter.FilterChain;
import javax.portlet.filter.FilterConfig;
import javax.portlet.filter.PortletFilter;
import javax.portlet.filter.ResourceFilter;
import com.liferay.portal.kernel.log.Log;
import com.liferay.portal.kernel.log.LogFactoryUtil;
import com.liferaystack.constants.MyPortletKeys;
import org.osgi.service.component.annotations.Component;

/**
 * @author Syed Ali
 */

@Component(immediate = true,
        property = {
                "javax.portlet.name="+MyPortletKeys.My,
        },
        service = PortletFilter.class
)

public class MyResourceFilter implements ResourceFilter{

        private static Log _log = LogFactoryUtil.getLog(MyResourceFilter.class);

        @Override
        public void init(FilterConfig filterConfig) throws PortletException {
                _log.info("init.....");
        }

        @Override
        public void destroy() {
                _log.info("destroy.....");
        }

        @Override
        public void doFilter(ResourceRequest request, ResourceResponse response, Filt
                _log.info("ResourceDoFilter.....");
                chain.doFilter(request, response);
        }
}
```

Servlet Filter In Liferay 7 DXP

This Filter Intercepts all the requests comes to the portlet, whether it is action, render, resource and it also can be HttpServletRequest , if you want filter all the Http Servlet Request comes to your portlet please use this filter, your filter component just have to extend "BaseFilter" Class, refer the below code

```
package com.liferaystack.filter;

import javax.servlet.Filter;
import javax.servlet.FilterChain;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import com.liferay.portal.kernel.log.Log;
import com.liferay.portal.kernel.log.LogFactoryUtil;
import com.liferay.portal.kernel.servlet.BaseFilter;
import org.osgi.service.component.annotations.Component;

/**
 * @author Syed Ali
 */

@Component(
        immediate = true,
        property = {
                "servlet-context-name=",
                "servlet-filter-name=http-filter",
                "url-pattern=/*"
        },
        service = Filter.class
)

public class MyBaseFilter extends BaseFilter{

        private static final Log _log = LogFactoryUtil.getLog(MyBaseFilter.class);

        @Override
        protected void processFilter(HttpServletRequest request, HttpServletResponse
                _log.info("processFilter BaseFilter..........");
                //filterChain.doFilter(request, response);
                super.processFilter(request, response, filterChain);
        }

        @Override
        protected Log getLog() {
                return _log;
        }
}
```

Note : This BaseFilter Will be Triggered Before all the other filters such as ActionFilter, RenderFilter and ResourceFilter.

I have created a portlet which can easily make you understand flow of Portlet Filters and Portlet Requests In detail, please download the portlet and deploy to test, it can give more insight into the portlet Filters In Liferay, when you add this portlet to the page init() and doFilter() method will be triggered.

Email Facebook Twitter WhatsApp More

3 COMMENTS:

1. Unknown (https://www.blogger.com/profile/05482393262504235900)
   October 3, 2017 at 5:21 PM (https://www.liferaystack.com/2017/09/portlet-filters-in-liferay-7-dxp.html?showComment=1507031492459#c2671607179512136987)
   I have made changes on mine portlet to run filters but nothing is working except Servlet Filter,
   plz help me out related to filters.

   Thanks

   REPLY DELETE (HTTPS://WWW.BLOGGER.COM/DELETE-COMMENT.G?BLOGID=6330292770501738275&POSTID=2671607179512136987)

2. Syed Ali (https://www.blogger.com/profile/10486568263079908670)
   Admin
   October 3, 2017 at 7:45 PM (https://www.liferaystack.com/2017/09/portlet-filters-in-liferay-7-dxp.html?showComment=1507040140726#c6586876420317347678)
   Please give more details about your issue,which portlet filter you are using action,render or resource, download the source code and refer it... its a tested example make sure all the configurations done properly

REPLY DELETE (HTTPS://WWW.BLOGGER.COM/DELETE-COMMENT.G?BLOGID=6330292770501738275&POSTID=6586876420317347678)

3. Andreas (https://www.blogger.com/profile/03303815740475896409)

November 17, 2020 at 12:44 PM (https://www.liferaystack.com/2017/09/portlet-filters-in-liferay-7-dxp.html?showComment=1605597287450#c1892706444575101206)

In hopes that this might get answered...Is it possible to use the BaseFilter & Servlet Filter implementation to create a module that acts as a Filter for all other modules deployed at the portal? If yes, how should I go about it?

Does the Servlet Filter actually have the ability to monitor all requests comming to the portal?

REPLY DELETE (HTTPS://WWW.BLOGGER.COM/DELETE-COMMENT.G?BLOGID=6330292770501738275&POSTID=1892706444575101206)

Enter your comment...

Comment as: Vegan (Google Sign out

Publish Preview Notify me

(https://www.blogger.com/comment-iframe.g?blogID=6330292770501738275&postID=7621976902094831054&blogspotRpcToken=5789474)

FACEBOOK (HTTPS://WWW.FACEBOOK.COM/LIFERAYSTACK/)

TWITTER (HTTPS://TWITTER.COM/LIFERAYSTACK)

WHATSAPP (WHATSAPP://SEND?TEXT=PORTLET FILTERS IN LIFERAY 7 DXP | LIFERAY STACK HTTPS://WWW.LIFERAYSTACK.COM/2017/09/PORTLET-FILTERS-IN-LIFERAY-7-DXP.HTML)

GOOGLE + (HTTPS://PLUS.GOOGLE.COM/U/0/111838714362004655991)

PINTEREST (HTTPS://IN.PINTEREST.COM/IAMSYEDALI/LIFERAYSTACK/)

YOUTUBE (HTTPS://WWW.YOUTUBE.COM/CHANNEL/UCC7WPSUSBNTJXEWECRBHRGQ/FEATURED)