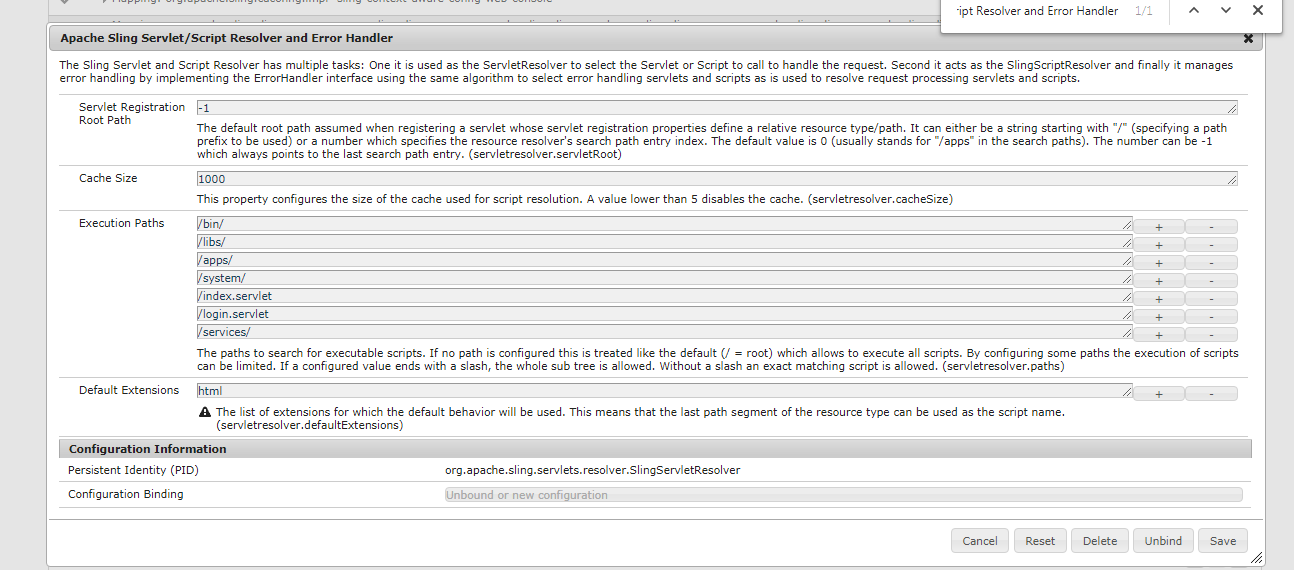
For registering a servlet with path, Path should start with the below configuration.



[1. How to register Servlet using Java annotations?](http://aeminterviewquestions.com/servlet.html#collapse1)

There are two ways of doing this

* 1. The @SlingServlet annotation

@SlingServlet(   
resourceTypes = "sling/servlet/default",   
selectors = "hello",   
extensions = "html",   
methods = "GET")   
public class MyServlet extends SlingSafeMethodsServlet {   
@Override   
protected void doGet(SlingHttpServletRequest request, SlingHttpServletResponse response) throws ServletException, IOException {   
...   
}   
}

The @Properties and @Property annotations

@Component(metatype = true)   
@Service(Servlet.class)   
@Properties({   
@Property(name = "sling.servlet.resourceTypes", value = "sling/servlet/default"),   
@Property(name = "sling.servlet.selectors", value = "hello"),   
@Property(name = "sling.servlet.extensions", value = "html"),   
@Property(name = "sling.servlet.methods", value = "GET")   
})   
public class MyServlet extends SlingSafeMethodsServlet {   
@Override   
protected void doGet(SlingHttpServletRequest request, SlingHttpServletResponse response) throws ServletException, IOException {   
...   
}   
}

[2. How are different ways to register servlet in AEM?](http://aeminterviewquestions.com/servlet.html#collapse2)

You can register a Servlet using the two Standard approaches:

* 1. Registering the servlet by path

@SlingServlet(   
paths={"/bin/customservlet/path"} )   
@Properties({   
@Property(name="service.pid", value="com.day.servlets.SampleServlet",propertyPrivate=false),   
@Property(name="service.description",value="SampleDescription", propertyPrivate=false),   
@Property(name="service.vendor",value="SampleVendor", propertyPrivate=false)   
})   
public class SampleServletname extends SlingAllMethodsServlet   
{   
@Override   
protected void doGet(SlingHttpServletRequest request, SlingHttpServletResponse response) throws ServletException, IOException   
{   
}   
}

* 2. Register servlet by ResourceType

@SlingServlet(   
resourceTypes = "sling/servlet/path",   
selectors = "json",   
extensions = "html",   
methods = "GET")   
public class MyServlet extends SlingSafeMethodsServlet {   
@Override   
protected void doGet(SlingHttpServletRequest request, SlingHttpServletResponse response) throws ServletException, IOException {   
...   
}   
}

The resource type(s) supported by the servlet. The property value must either be a single String, an array of Strings or a Vector of Strings. This property is ignored if the sling.servlet.paths property is set. More on **Register servlet by Resource Type**watch this[video](https://www.youtube.com/watch?v=ABEmXz2VLUE)

#### [. Difference between SlingSafeMethodsServlet and SlingAllMethodsServlet.](http://aeminterviewquestions.com/servlet.html#collapse3)

**SlingSafeMethodsServlet**- Helper base class for read-only Servlets used in Sling. This base class is actually just a better implementation of the Servlet API HttpServlet class which accounts for extensibility. So extensions of this class have great control over what methods to overwrite. It supports GET, HEAD, OPTIONS etc methods. [Read more](https://sling.apache.org/apidocs/sling5/org/apache/sling/api/servlets/SlingSafeMethodsServlet.html)

**SlingAllMethodsServlet**- Helper base class for data modifying Servlets used in Sling. This class extends the SlingSafeMethodsServlet by support for the POST, PUT and DELETE methods. [Read more](https://sling.apache.org/apidocs/sling5/org/apache/sling/api/servlets/SlingAllMethodsServlet.html)

#### [Suppose I have added sling.servlet.paths and sling.servlet.resourceTypes both in servlet. What will happen?](http://aeminterviewquestions.com/servlet.html#collapse4)

Suppose I have added sling.servlet.paths and sling.servlet.resourceTypes both in servlet. **sling.servlet.resourceTypes property is ignored if the sling.servlet.paths property is set.**

#### [What is resourceresolver?](http://aeminterviewquestions.com/servlet.html#collapse6)

The ResourceResolver defines the service API which may be used to resolve Resource objects. The resource resolver is available to the request processing servlet through the SlingHttpServletRequest.getResourceResolver() method. A resource resolver can also be created through the ResourceResolverFactory. The ResourceResolver is also an Adaptable to get adapters to other types.

[Read more](https://docs.adobe.com/docs/en/aem/6-0/develop/ref/javadoc/org/apache/sling/api/resource/ResourceResolver.html)

[7. Registering the servlet by path vs ResourceType?](http://aeminterviewquestions.com/servlet.html#collapse7)

Registering the servlet by resourceType is more preferential than path , because

* use of resourceType is that the Sling Engine will take care of permissions for you. Users who cannot access a particular resource will not be able to invoke the servlet. Hence register by resourcetype is more secure.
* While defining a path , you must be specific what all paths are allowed to be used in the ServletResource OSGi service. If you define something randomly, your servlet might not be fucntional. Only a limited paths are allowed and the rest are blocked unless you open them up. This is resolved using resourceType.

#### [what is serialVersionUID in servlet?](http://aeminterviewquestions.com/servlet.html#collapse8)

The serialization runtime associates with each serializable class a version number, called a serialVersionUID, which is used during deserialization to verify that the sender and receiver of a serialized object have loaded classes for that object that are compatible with respect to serialization. If the receiver has loaded a class for the object that has a different serialVersionUID than that of the corresponding sender's class, then deserialization will result in an InvalidClassException. A serializable class can declare its own serialVersionUID explicitly by declaring a field named "serialVersionUID" that must be static, final, and of type long:

ANY-ACCESS-MODIFIER static final long serialVersionUID = 1L;

If a serializable class does not explicitly declare a serialVersionUID, then the serialization runtime will calculate a default serialVersionUID value for that class based on various aspects of the class, as described in the Java(TM) Object Serialization Specification. However, it is strongly recommended that all serializable classes explicitly declare serialVersionUID values, since the default serialVersionUID computation is highly sensitive to class details that may vary depending on compiler implementations, and can thus result in unexpected InvalidClassExceptions during deserialization

http://aeminterviews.blogspot.com/2016/10/tricky-questions-in-aem.html