**SLING SERVLET :-**

It is used to handle the HTTP Servlet request and responses in AEM.

**List of HTTP Methods and Their Characteristics**

**GET: Retrieve data from the server.**

Safe

Idempotent

Cacheable

**HEAD: Similar to GET, but without the response body.**

Safe

Idempotent

Cacheable

**OPTIONS: Describe the communication options for the target** resource.

Safe

Idempotent

**TRACE: Perform a message loop-back test along the path to the target resource.**

Safe

Idempotent

**POST: Submit data to the server, typically resulting in a change in state or side effects.**

Not Safe

Not Idempotent

**PUT: Replace all current representations of the target resource with the request payload.**

Not Safe

Idempotent

**DELETE: Remove the specified resource.**

Not Safe

Idempotent

**PATCH: Apply partial modifications to a resource.**

Not Safe

Not Idempotent

Other Sling Annotations

@SlingServletMethods: Defines which HTTP methods the servlet should handle.

**Uses of GET and POST Methods in Servlets**

**3.1 GET Method**

Purpose: Retrieve information from the server

**Characteristics:**

Parameters are sent as part of the URL.

Limited amount of data can be sent (due to URL length restrictions).

Typically used for requesting data, not modifying it.

Safe and idempotent (does not change server state).

**Example Usage**: Fetching data, search queries, fetching resources.

**3.2 POST Method**

Purpose: Send data to the server.

**Characteristics:**

Parameters are sent in the request body.

No size limitations (practically, the size is limited by the server's configuration).

Typically used for form submissions, uploading files, and any action that changes server state.

Not idempotent (may change server state).

**Example Usage:**Submitting form data, uploading files, creating new records.

In Apache Sling, servlets are mapped to resources and methods using annotations. When dealing with HTTP methods, Sling provides annotations to declare which methods are considered "safe" or "all safe."

**1. @SlingServletPaths**

This annotation is used to define the paths under which the servlet should be registered.

**@SlingServletPaths(value = { "/bin/example" })public class ExampleServlet extends SlingAllMethodsServlet {**

**// servlet implementation**

**}**

**2. @SlingServletResourceTypes**

This annotation is used to define the resource types under which the servlet should be registered.

@SlingServletResourceTypes(resourceTypes = { "example/components/page" })public class ExampleServlet extends SlingAllMethodsServlet {

// servlet implementation

}

**SLING SAFE METHODS AND SLING ALL SAFE METHODS :-**

**Difference between them and uses :-**

1. **SlingSafeMethodsServlet**

Purpose: To handle only safe HTTP methods.

**Safe Methods: GET, HEAD, OPTIONS, TRACE**

Characteristics:

These methods are considered safe because they are intended only for information retrieval and do not alter the server state.

**ex :-**

@SlingServletPaths(value = { "/bin/safe" })

public class SafeServlet extends SlingSafeMethodsServlet {

@Override

protected void doGet(SlingHttpServletRequest request, SlingHttpServletResponse response)

throws ServletException, IOException {

response.getWriter().write("This is a GET request");

}

}

**2. SlingAllMethodsServlet**

Purpose: To handle all HTTP methods, including those that can alter server state.

**Methods: GET, POST, PUT, DELETE, PATCH, HEAD, OPTIONS, TRACE**

Characteristics:

This servlet can handle both safe and unsafe methods, making it suitable for handling CRUD operations.

ex;-

@SlingServletPaths(value = { "/bin/all" })public class AllMethodsServlet extends SlingAllMethodsServlet {

@Override

protected void doGet(SlingHttpServletRequest request, SlingHttpServletResponse response)

throws ServletException, IOException {

response.getWriter().write("This is a GET request");

}

@Override

protected void doPost(SlingHttpServletRequest request, SlingHttpServletResponse response)

throws ServletException, IOException {

response.getWriter().write("This is a POST request");

}

}

**REQUIREMENT :-**  
I want to print the jcr:title using the resorcetype option

**CODE :-**

**With this code we are using a servlet in order to get the page jcr:title variable. wherever the page we are loading it checks if the page is having the resource type which we mentioned in the code -- >** mysite/components/page. **> so if the page is having the resource type of mysite/components/page we will get the jcr:title on the page automatically as below.**



package com.adobe.aem.mysite.site.core.servlets;

import org.apache.sling.api.SlingHttpServletRequest;

import org.apache.sling.api.SlingHttpServletResponse;

import org.apache.sling.api.servlets.SlingAllMethodsServlet;

import org.apache.sling.api.resource.Resource;

import org.apache.sling.api.resource.ResourceResolver;

import org.apache.sling.api.servlets.SlingSafeMethodsServlet;

import org.osgi.service.component.annotations.Component;

import org.osgi.framework.Constants;

import javax.servlet.Servlet;

import javax.servlet.ServletException;

import java.io.IOException;

@Component(

service = { Servlet.class },

property = {

Constants.SERVICE\_DESCRIPTION + "=Sample Servlet",

"sling.servlet.methods=" + "GET",

"sling.servlet.resourceTypes=" + "mysite/components/page"

}

)

public class SampleServlet extends SlingSafeMethodsServlet {

@Override

protected void doGet(SlingHttpServletRequest request, SlingHttpServletResponse response) throws ServletException, IOException {

response.setContentType("text/html");

Resource resource = request.getResource();

// Get properties from the resource or perform other operations

String title = resource.getValueMap().get("jcr:title", String.class);

response.getWriter().write("<h1>Title: " + title + "</h1>");

}

}