Implementing AEM Servlets for Page Creation and Search

1. Introduction

This document provides a step-by-step guide for implementing three servlets in Adobe Experience Manager (AEM):

- `SampleServlet` using `SlingAllMethodsServlet` registered via `resourceType`

- `CreatePageServlet` using `SlingSafeMethodsServlet` registered via `path`

- `SearchServlet` utilizing `PredicateMap` for searching content

2. Prerequisites

- AEM instance running

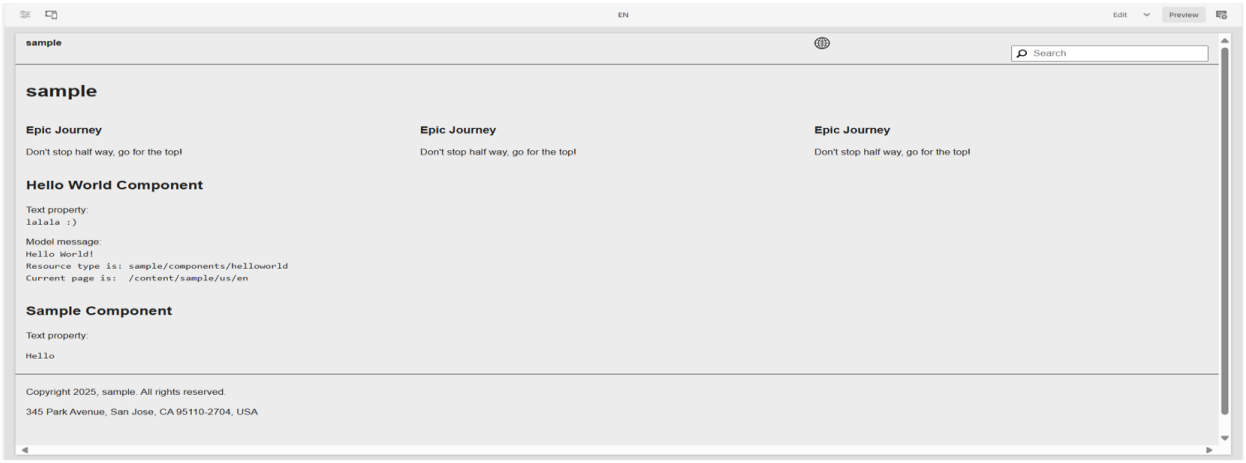
- Knowledge of Java and AEM development

- AEM Project setup using Maven

- Access to CRXDE and AEM's OSGi configuration

3. Creating SampleServlet

This servlet extends `SlingAllMethodsServlet` and is registered using `resourceType`.



3.1 Code Implementation

```
@Component(
        service = Servlet.class,
        property = {
                "sling.servlet.resourceTypes=myproject/components/sample",
                "sling.servlet.methods=GET"
```

```
    }
)
public class SampleServlet extends SlingAllMethodsServlet {
   @Override
   protected void doGet(SlingHttpServletRequest request, SlingHttpServletResponse
response)
         throws ServletException, IOException {
      response.getWriter().write("Sample Servlet Executed");
   }
}
```

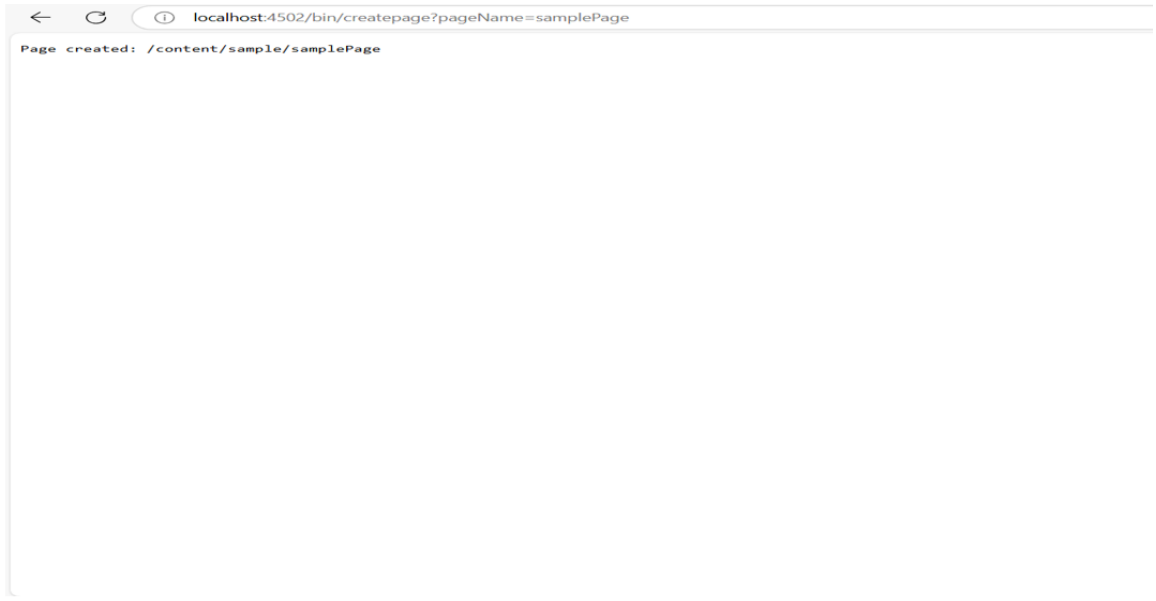
4. Creating CreatePageServlet

This servlet extends `SlingSafeMethodsServlet` and creates pages dynamically.

4.1 Code Implementation

```
@Component(
      service = Servlet.class,
      property = {
            "sling.servlet.paths=/bin/createpage",
            "sling.servlet.methods=GET"
      }
)
public class CreatePageServlet extends SlingSafeMethodsServlet {
   @Override
   protected void doGet(SlingHttpServletRequest request, SlingHttpServletResponse
response)
         throws ServletException, IOException {
      String pageName = request.getParameter("name");
      if (pageName == null || pageName.isEmpty()) {
         response.getWriter().write("Invalid page name");
         return;
      }
      PageManager pageManager =
request.getResourceResolver().adaptTo(PageManager.class);
      if (pageManager != null) {
         try {
            Page newPage = pageManager.create("/content/myproject", pageName,
"myproject/templates/basic", pageName);
            response.getWriter().write("Page Created: " + newPage.getPath());
```

```
            } catch (WCMException e) {
                response.getWriter().write("Error creating page: " + e.getMessage());
            }
        }
    }
}
```



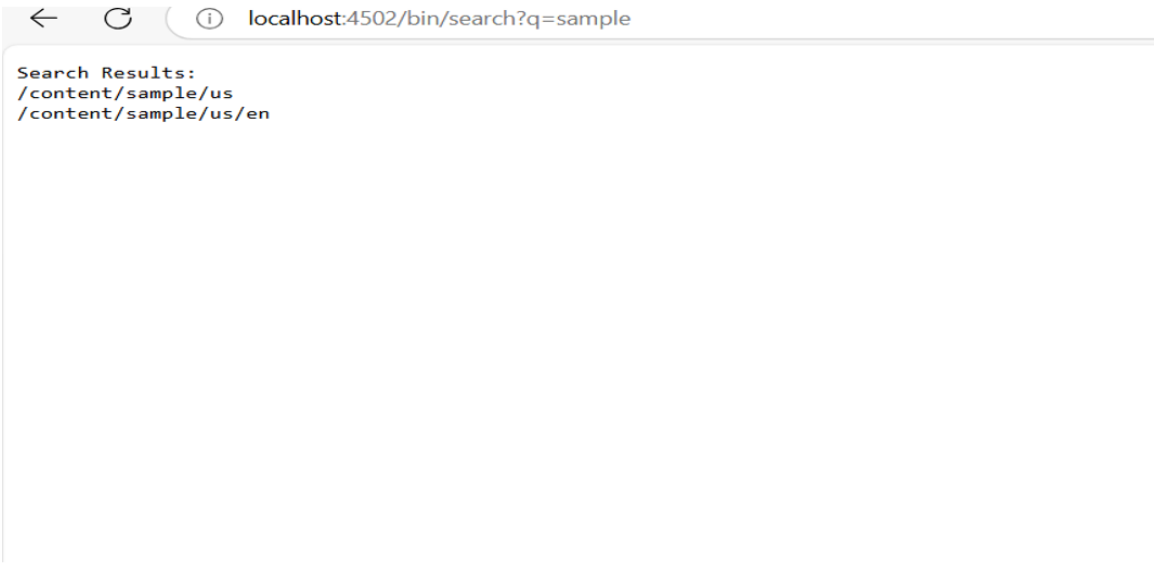
5. Creating SearchServlet

This servlet uses `PredicateMap` to search content.

5.1 Code Implementation

```
@Component(
        service = Servlet.class,
        property = {
                "sling.servlet.paths=/bin/searchcontent",
                "sling.servlet.methods=GET"
        }
)
public class SearchServlet extends SlingSafeMethodsServlet {
    @Reference
    private QueryBuilder queryBuilder;

    @Override
    protected void doGet(SlingHttpServletRequest request, SlingHttpServletResponse
response) throws IOException {
        String searchText = request.getParameter("query");
        if (searchText == null || searchText.isEmpty()) {
            response.getWriter().write("Invalid search query");
            return;
        }
        Map<String, String> predicates = new HashMap<>();
        predicates.put("path", "/content");
        predicates.put("type", "cq:Page");
        predicates.put("fulltext", searchText);
        Query query = queryBuilder.createQuery(PredicateGroup.create(predicates),
request.getResourceResolver().adaptTo(Session.class));
        Result result = query.getResult();
        response.getWriter().write("Found " + result.getHits().size() + " results");
    }
}
```


localhost:4502/bin/search?q=sample
Search Results:
/content/sample/us
/content/sample/us/en