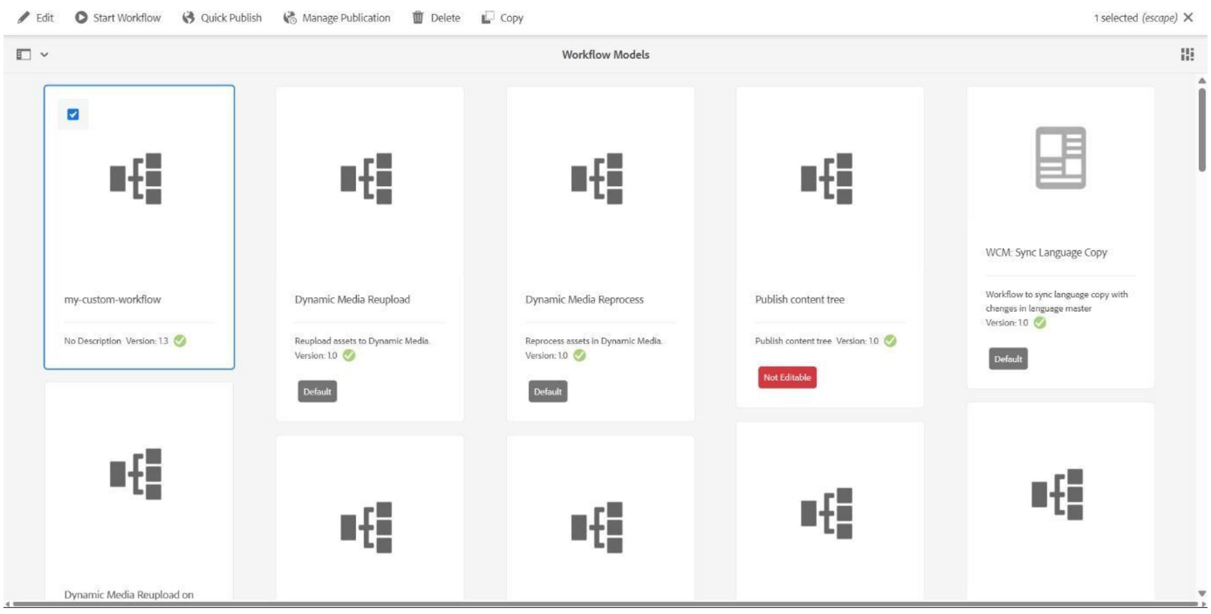
25/03/2025 - TASKS

Step 1: Create a Custom Workflow ("my custom workflow")

1. **Go to AEM → Tools → Workflow → Models.**
2. **Click Create → Create Model, name it "my custom workflow".**
3. **Open the model editor and:**
   - **Drag the "Process Step" component onto the workflow.**
   - **Double-click it and set:**
     - **Process: Select "myCustomWorkflowProcess" (to be created in Step 2).**
4. **Save and activate the workflow.**


Edit
Start Workflow
Quick Publish
Manage Publication
Delete
Copy
1 selected (escape)
Workflow Models
my-custom-workflow
No Description Version: 1.3
Dynamic Media Reupload
Reupload assets to Dynamic Media. Version: 1.0
Default
Dynamic Media Reprocess
Reprocess assets in Dynamic Media. Version: 1.0
Default
Publish content tree
Publish content tree Version: 1.0
Not Editable
WCM: Sync Language Copy
Workflow to sync language copy with changes in language master Version: 1.0
Default
Dynamic Media Reupload on

Step 2: Create a Custom Workflow Process to Print Page Title in Logs

1. **Implement a Java class extending WorkflowProcess.**
2. **Print the page title in logs using workflowSession.getMetaDataMap().**
3. **Deploy the bundle and configure the process step in myCustomWorkflow.**
4. **Apply the workflow to a page and observe logs in AEM.**

Java Code

```
package com.myTraining.core.workflows;

import com.adobe.granite.workflow.WorkflowSession;

import com.adobe.granite.workflow.exec.WorkItem;

import com.adobe.granite.workflow.exec.WorkflowProcess;
import com.adobe.granite.workflow.metadata.MetaDataMap;

import com.day.cq.wcm.api.Page;

import com.day.cq.wcm.api.PageManager;

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

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

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

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;


@Component(
 service = WorkflowProcess.class,
 property = { "process.label=Custom Workflow Process" }
)
public class CustomWorkflowProcess implements WorkflowProcess {
 private static final Logger LOGGER =
LoggerFactory.getLogger(CustomWorkflowProcess.class);
 @Override
 public void execute(WorkItem workItem, WorkflowSession workflowSession,
MetaDataMap metaDataMap) {
 LOGGER.info("[Custom Workflow] Executing workflow process...");
 if (workItem == null || workItem.getWorkflowData() == null) {
 LOGGER.error("[Custom Workflow] WorkItem or WorkflowData is NULL!");
 return;
 }
 String payloadPath = workItem.getWorkflowData().getPayload().toString();
 LOGGER.info("[Custom Workflow] Payload Path: {}", payloadPath);
 ResourceResolver resolver = workflowSession.adaptTo(ResourceResolver.class);
 if (resolver != null) {
 Resource resource = resolver.getResource(payloadPath);
 if (resource != null) {
```

PageManager pageManager = resolver.adaptTo(PageManager.class);

Page page = pageManager.getContainingPage(resource);

if (page != null) {

LOGGER.info("[Custom Workflow] Page Title: {}", page.getTitle());

LOGGER.info("[Custom Workflow] Page Path: {}", page.getPath());

} else {

LOGGER.warn("[Custom Workflow] No Page found for the given resource.");

}

} else {

LOGGER.warn("[Custom Workflow] No Resource found at the given path.");

}

} else {

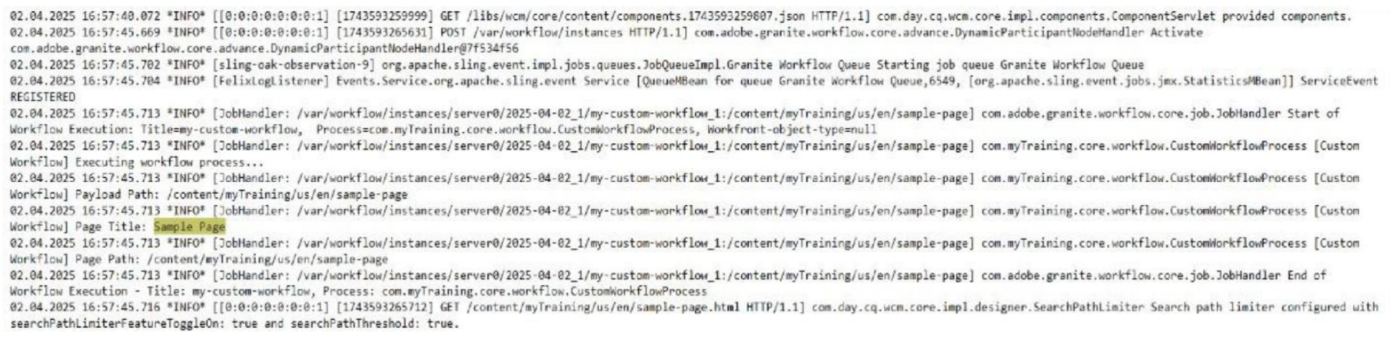
LOGGER.error("[Custom Workflow] Resource Resolver is NULL.");

}

}

}

Step 3: Create an Event Handler to Print Resource Path in Logs



```
02.04.2025 16:57:40.072 *INFO* [[0:0:0:0:0:0:0:1] [1743593259999] GET /libs/wcm/core/content/components.1743593259807.json HTTP/1.1] com.day.cq.wcm.core.impl.components.ComponentServlet provided components.
02.04.2025 16:57:45.669 *INFO* [[0:0:0:0:0:0:0:1] [1743593265631] POST /var/workflow/instances HTTP/1.1] com.adobe.granite.workflow.core.advance.DynamicParticipantNodeHandler Activate
com.adobe.granite.workflow.core.advance.DynamicParticipantNodeHandler@7f534f56
02.04.2025 16:57:45.702 *INFO* [sling-oak-observation-9] org.apache.sling.event.impl.jobs.queues.JobQueueImpl.Granite Workflow Queue Starting job queue Granite Workflow Queue
02.04.2025 16:57:45.704 *INFO* [FelixLogListener] Events.Service.org.apache.sling.event Service [QueueMBean for queue Granite Workflow Queue,6549, [org.apache.sling.event.jobs.jmx.StatisticsMBean]] ServiceEvent
REGISTERED
02.04.2025 16:57:45.713 *INFO* [JobHandler: /var/workflow/instances/server0/2025-04-02_1/my-custom-workflow_1:/content/myTraining/us/en/sample-page] com.adobe.granite.workflow.core.job.JobHandler Start of
Workflow Execution: Title=my-custom-workflow,  Process=com.myTraining.core.workflow.CustomWorkflowProcess, Workfront-object-type=null
02.04.2025 16:57:45.713 *INFO* [JobHandler: /var/workflow/instances/server0/2025-04-02_1/my-custom-workflow_1:/content/myTraining/us/en/sample-page] com.myTraining.core.workflow.CustomWorkflowProcess [Custom
Workflow] Executing workflow process...
02.04.2025 16:57:45.713 *INFO* [JobHandler: /var/workflow/instances/server0/2025-04-02_1/my-custom-workflow_1:/content/myTraining/us/en/sample-page] com.myTraining.core.workflow.CustomWorkflowProcess [Custom
Workflow] Payload Path: /content/myTraining/us/en/sample-page
02.04.2025 16:57:45.713 *INFO* [JobHandler: /var/workflow/instances/server0/2025-04-02_1/my-custom-workflow_1:/content/myTraining/us/en/sample-page] com.myTraining.core.workflow.CustomWorkflowProcess [Custom
Workflow] Page Title: Sample Page
02.04.2025 16:57:45.713 *INFO* [JobHandler: /var/workflow/instances/server0/2025-04-02_1/my-custom-workflow_1:/content/myTraining/us/en/sample-page] com.myTraining.core.workflow.CustomWorkflowProcess [Custom
Workflow] Page Path: /content/myTraining/us/en/sample-page
02.04.2025 16:57:45.713 *INFO* [JobHandler: /var/workflow/instances/server0/2025-04-02_1/my-custom-workflow_1:/content/myTraining/us/en/sample-page] com.adobe.granite.workflow.core.job.JobHandler End of
Workflow Execution - Title: my-custom-workflow, Process: com.myTraining.core.workflow.CustomWorkflowProcess
02.04.2025 16:57:45.716 *INFO* [[0:0:0:0:0:0:0:1] [1743593265712] GET /content/myTraining/us/en/sample-page.html HTTP/1.1] com.day.cq.wcm.core.impl.designer.SearchPathLimiter Search path limiter configured with
searchPathLimiterFeatureToggleOn: true and searchPathThreshold: true.
```

Java Code

package com.myTraining.core.listeners;

import org.apache.sling.api.resource.observation.ResourceChange;

import org.apache.sling.api.resource.observation.ResourceChangeListener;

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

import org.slf4j.Logger;

```
import org.slf4j.LoggerFactory;
import java.util.List;


@Component(
 service = ResourceChangeListener.class,
 property = {
 ResourceChangeListener.PATHS + "=/content/myTraining/us",
 ResourceChangeListener.CHANGES + "=ADDED",
 ResourceChangeListener.CHANGES + "=CHANGED",
 ResourceChangeListener.CHANGES + "=REMOVED"
 }
)
public class ResourceEventHandler implements ResourceChangeListener {
 private static final Logger LOGGER =
LoggerFactory.getLogger(ResourceEventHandler.class);
 @Override
 public void onChange(List<ResourceChange> changes) {
 for (ResourceChange change : changes) {
 LOGGER.info("[Resource Event] Type: {} | Path: {}", change.getType(),
change.getPath());
 }
 }
}
```

Step 4: Create a Sling Job to Print "Hello World" in Logs

Java Code

```
package com.myTraining.core.schedulers;
import org.apache.sling.event.jobs.Job;
import org.apache.sling.event.jobs.consumer.JobConsumer;
import org.osgi.service.component.annotations.Component;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;


@Component(
 service = JobConsumer.class,
 property = { JobConsumer.PROPERTY_TOPICS + "=myTraining/job/helloWorld" }
)
public class HelloWorldJob implements JobConsumer {
 private static final Logger LOGGER =
LoggerFactory.getLogger(HelloWorldJob.class);
 @Override
 public JobResult process(Job job) {
 LOGGER.info("[Sling Job] Executing Hello World Job...");
 LOGGER.info("[Sling Job] Hello World!");
 return JobResult.OK;
 }
}
```

Step 5: Create a Scheduler to Print "Yellow World" Every 5 Minutes

Java Code

```
package com.myTraining.core.schedulers;
import org.osgi.service.component.annotations.Activate;
import org.osgi.service.component.annotations.Component;
import org.osgi.service.component.annotations.Modified;
import org.osgi.service.metatype.annotations.AttributeDefinition;
import org.osgi.service.metatype.annotations.ObjectClassDefinition;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import java.util.concurrent.atomic.AtomicBoolean;


@Component(service = Runnable.class, immediate = true, configurationPid =
"com.myproject.core.schedulers.YellowWorldScheduler")
```

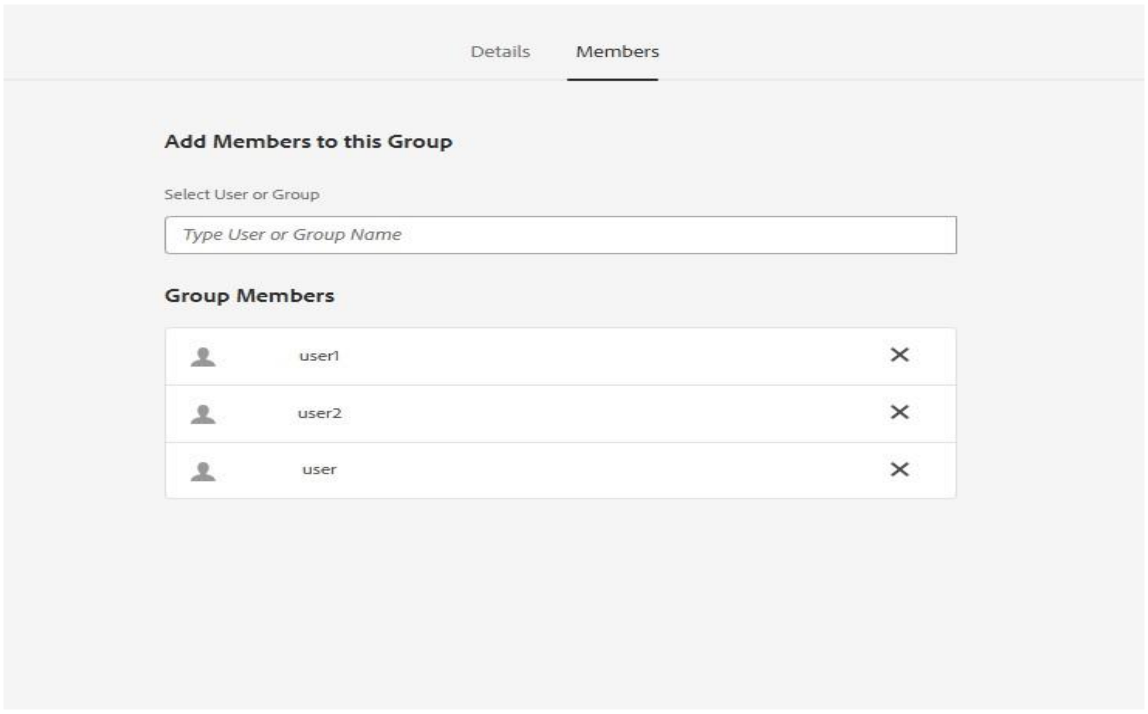
```
public class YellowWorldScheduler implements Runnable {
 private static final Logger LOG =
LoggerFactory.getLogger(YellowWorldScheduler.class);
 @ObjectClassDefinition(name = "Yellow World Scheduler Configuration")
 public @interface Config {
 @AttributeDefinition(name = "Cron Expression")
 String scheduler_expression() default "0 */5 * * * ?";
 }
 private final AtomicBoolean running = new AtomicBoolean(false);
 @Activate
 @Modified
 protected void activate(final Config config) {
 LOG.info("YellowWorldScheduler activated with cron expression: {}",
config.scheduler_expression());
 }
 @Override
 public void run() {
 if (running.compareAndSet(false, true)) {
 try {
 LOG.info("Yellow World");
 } finally {
 running.set(false);
 }
 }
 }
}
```

Step 6: Create Users, Assign to Group, and Set Permissions

1. **Navigate to AEM → Tools → Security → Users.**
2. **Click Create User and add:**
   - **User1: author1**

- **User2: author2**
- **User3: author3**

3. **Navigate to AEM → Tools → Security → Groups, create a new group:**
   - **Group Name: Dev Authors**
4. **Add the 3 users to the "Dev Authors" group.**
5. **Set permissions:**
   - **Go to /content and /dam folders in User Permissions.**
   - **Set Read-Only access.**
   - **Grant Replication permission.**
6. **Save and verify that these users cannot edit but can replicate content.**


Details
Members
Add Members to this Group
Select User or Group
Type User or Group Name
Group Members
user1
user2
user