

## 25/03/2025 - TASKS

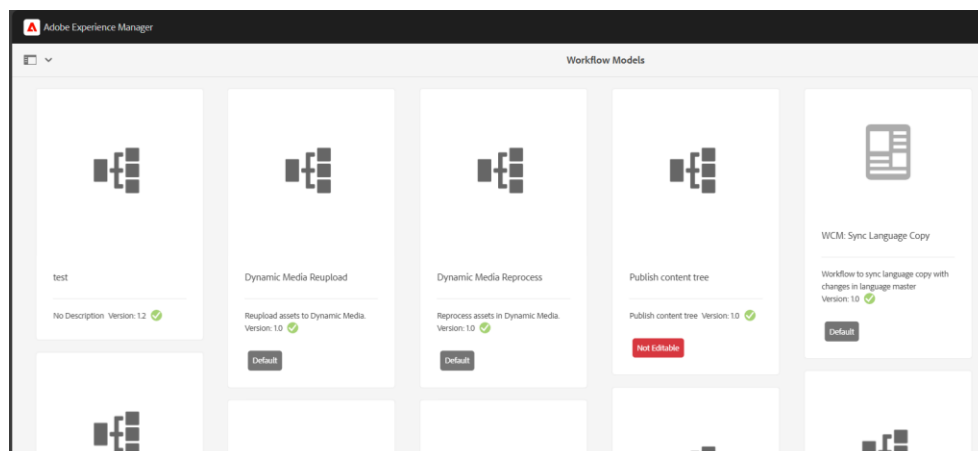
### Table of Contents

1. Create Custom Workflow Model
  2. Create Custom Workflow Process
  3. Create Event Handler
  4. Create Sling Job
  5. Create Scheduler
  6. Create Users & Group with Permissions
  7. Test and Troubleshoot Custom Workflow, Event Handler, Sling Job, and Scheduler
  8. Best Practices for Workflow and Event Handling in AEM
- 

### 1. Create Custom Workflow Model

#### Steps:

- Navigate to the AEM Workflow Console.
- Click "Create" and define a new workflow model.
- Drag workflow components into the model.
- Configure transitions and conditions for each step.
- Save, activate, and validate functionality.



## 2. Create Custom Workflow Process

### Steps:

- Develop a Java class implementing WorkflowProcess.
- Write logic inside the execute method.
- Register the class as an OSGi component.
- Deploy and test the workflow process.

### Example Code:

```
@Component(service = WorkflowProcess.class, property = {"process.label=Custom Workflow"})

public class CustomWorkflowProcess implements WorkflowProcess {

    @Override

    public void execute(WorkItem item, WorkflowSession session, MetadataMap metaData)
throws WorkflowException {

        String payload = (String) item.getWorkflowData().getPayload();

        System.out.println("Processing: " + payload);

    }

}
```

---

## 3. Create Event Handler

### Steps:

- Implement EventHandler interface in Java.
- Register event topics in the OSGi component.
- Define logic inside the handleEvent method.
- Deploy and validate event handling.

### Example Code:

```
@Component(immediate = true, service = EventHandler.class, property = {"event.topics=org/apache/sling/api/resource/Resource/ADDED"})
```

```
public class CustomEventHandler implements EventHandler {  
    @Override  
    public void handleEvent(Event event) {  
        System.out.println("Event received: " + event.getTopic());  
    }  
}
```

---

#### 4. Create Sling Job

##### Steps:

- Implement JobConsumer to process Sling jobs.
- Register the job in the OSGi service properties.
- Deploy and monitor execution.

##### Example Code:

```
@Component(service = JobConsumer.class, property = {JobConsumer.PROPERTY_TOPICS +  
"=custom/job"})
```

```
public class CustomJobConsumer implements JobConsumer {  
    @Override  
    public JobResult process(Job job) {  
        System.out.println("Processing job: " + job.getTopic());  
        return JobResult.OK;  
    }  
}
```

---

#### 5. Create Scheduler

##### Steps:

- Develop a Runnable OSGi component.

- Configure scheduling parameters.
- Deploy and monitor task execution.

Example Code:

```
@Designate(ocd = ScheduledTask.Config.class)
@Component(service = Runnable.class)
public class ScheduledTask implements Runnable {
    @Override
    public void run() {
        System.out.println("Scheduled task executed");
    }
}
```

---

## 6. Create Users & Group with Permissions

Steps:

- Open AEM User Management.
- Create new users and groups.
- Assign permissions to roles.

- Validate access control settings.

The screenshot shows the AEM Access Control configuration interface. It includes a 'Path' field with the value '/content', a 'Privileges' section with a search bar containing 'Type to add privileges' and a list of privileges including 'jcr:read', and a 'Permission Type' section with 'Deny' and 'Allow' radio buttons. Below these are 'Restrictions' fields with a 'Select Type' dropdown and a 'Restriction Value' input. At the bottom, there is an information icon and a paragraph explaining that privileges define access rights at a path in the repository, and restrictions help refine the effect of an Access Control Entry. A link to the 'documentation page' is provided.

---

## 7. Test and Troubleshoot Custom Workflow, Event Handler, Sling Job, and Scheduler

### Testing Steps:

- Trigger workflows manually.
- Monitor system logs.
- Debug issues and refine logic.
- Ensure expected execution of all components.

---

## 8. Best Practices for Workflow and Event Handling in AEM

### Guidelines:

- Use asynchronous processing for performance.
- Secure event listeners and job execution.
- Optimize workflows for efficiency.
- Follow AEM coding standards and best practices.

