



**DEVELOPERS LIVE**

# **Custom Functional Tests for safer CI/CD pipelines**

**Andres Bott | AEM Cloud Software Reliability Engineer**



# Cloud Service

AEM Cloud service allows quick updates to minimize the time to market.

We need confidence that changes won't break the user-experience.

Cloud service has many quality gates that need to pass before code is deployed to production

Custom functional tests allows you to write quality gates for your application.

This applies to both:

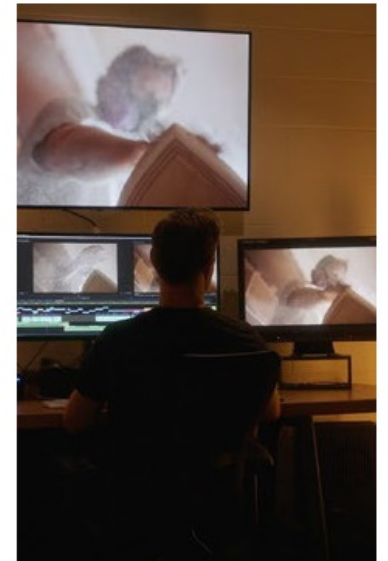
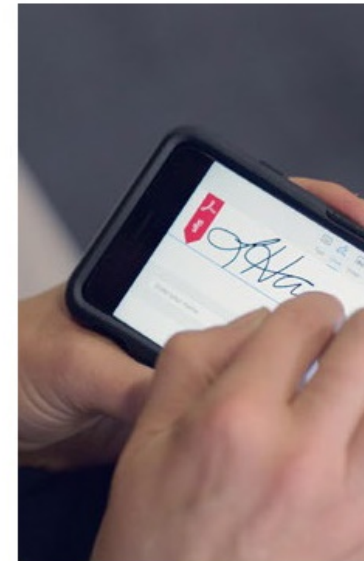
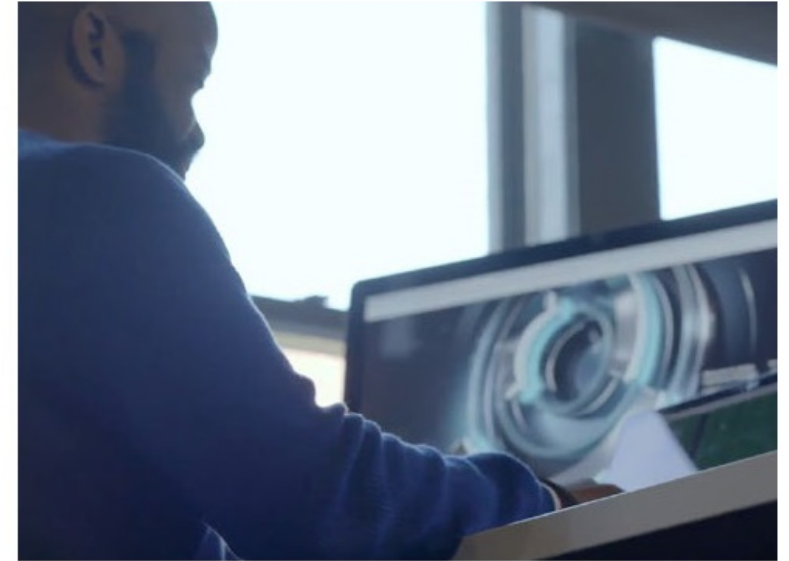
- Application deployment
- Product push updates

The screenshot displays the Adobe Cloud Manager interface for a 'Production Pipeline' that is 'Finished'. The pipeline consists of several stages:

- Stage deployment** (highlighted in the left sidebar):
  - Stage testing
  - Production deployment
- Build Images**: Your built artifacts will be combined with the base AEM 2020.8.4081.20200820T070745Z to produce an image runnable in the cloud. AEM RELEASE: 2020.8.4081.20200820T070745Z. Status: **Completed** (Finished: August 25, 2020 6:25:49 PM GMT+2).
- Deploy to Stage**: Deployment complete. Status: **Completed** (Finished: August 25, 2020 6:44:43 PM GMT+2).
- STAGE TESTING**:
  - Product Functional Testing**: Execution of the product functional tests is complete. Status: **Passed** (Finished: August 25, 2020 6:48:14 PM GMT+2).
  - Custom Functional Testing**: Execution of the custom functional tests is complete. Status: **Passed** (Finished: August 25, 2020 6:50:33 PM GMT+2).
- Experience Audit**: Your site has been audited against a set of rules to validate content quality, performance, and user experience. Status: **Passed** (Finished: August 25, 2020 6:53:15 PM GMT+2).
- PRODUCTION DEPLOYMENT**:
  - Deploy to Production**: Deployment complete. Status: **Completed** (Finished: August 25, 2020 7:11:13 PM GMT+2).

## About product functional tests

- Same technologies as the Custom Functional tests.
- Tests written and maintained by Adobe
- Prevent application code changes to break core functionality.
- Open-sourced and build following the best practices.
  - [aem-test-samples](#)



# Custom functional tests

- Part of your application code.
- Test the interaction of your application within the whole AEM Cloud stack:
  - This includes load balancer and dispatcher (publish)
- Executed as part of production deployment pipeline
- Block deployments if they don't pass.
- ... But you can run them locally.



## Custom functional tests (2)

- Tests are written in Java using use AEM testing clients and Junit rules.
- Http requests that allow remote controlling AEM from java code.
- Custom functional tests are NOT:
  - unit tests.
  - Selenium UI tests



# Where to start



# AEM Archetype

- Starting with the AEM project archetype

```
mvn -B archetype:generate
-D archetypeGroupId=com.adobe.aem
-D archetypeArtifactId=aem-project-archetype
-D archetypeVersion=25
-D appTitle="My Site"
-D appId="mysite"
-D groupId="com.mysite"
```

```
ls -l mysite/it.tests/src/main/java/com/mysite/it/tests/
```

It contains two basic IT tests: CreatePageIT and GetPageIT (demo)

- They are created as build-artifacts that can be executed outside of the maven build process
- All test class names must end with IT to be picked up. (defined in pom.xml)

# Running tests locally (demo)

Http request against a local instance or a cloud instance.

On cloud instances an admin-like user needs to be created.

```
AUTHOR_URL="https://author-pXXXX-eXXXX.adobeaemcloud.com/"
```

```
# (alternative) AUTHOR_URL="http://localhost:4502"
```

```
...
```

```
mvn clean verify \
```

```
-Plocal \
```

```
-Dit.author.url="${AUTHOR_URL}" \
```

```
-Dit.author.user="${AUTHOR_USER}" \
```

```
-Dit.author.password="${AUTHOR_PASSWORD}" \
```

```
-Dit.publish.url="${PUBLISH_URL}" \
```

```
-Dit.publish.user="${PUBLISH_USER}" \
```

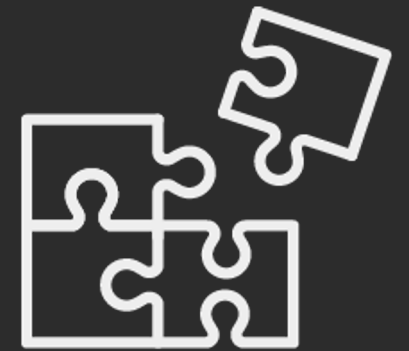
```
-Dit.publish.password="${PUBLISH_PASSWORD}"
```

```
[INFO] --- maven-resources-plugin:3.0.2:testResources (default-testResources) @
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] skip non existing resourceDirectory /home/bott/.datos/sync/edit/Developer
[INFO] --- maven-compiler-plugin:3.8.0:testCompile (default-testCompile) @ aem-g
[INFO] No sources to compile
[INFO] --- maven-surefire-plugin:2.21.0:test (default-test) @ aem-guides-wknd.it
[INFO] Tests are skipped.
[INFO] --- maven-jar-plugin:3.0.2:jar (default-jar) @ aem-guides-wknd.it.tests -
[INFO] Building jar: /home/bott/.datos/sync/edit/DevelopersConference/Developers
[INFO] --- maven-assembly-plugin:3.1.0:single (make-assembly) @ aem-guides-wknd.
[INFO] Building jar: /home/bott/.datos/sync/edit/DevelopersConference/Developers
th-dependencies.jar
[INFO] --- maven-failsafe-plugin:2.21.0:integration-test (default-integration-te
[INFO] -----
[INFO] T E S T S
[INFO] -----
[INFO] Running com.adobe.aem.guides.wknd.it.tests.CreatePageIT
[main] INFO com.adobe.cq.testing.junit.rules.ConfigurableInstance - Using LoginT
[main] INFO org.apache.sling.testing.junit.rules.instance.util.ConfigurationPool
[main] INFO org.apache.sling.testing.junit.rules.instance.util.ConfigurationPool
[main] INFO org.apache.sling.testing.junit.rules.instance.ExistingInstanceStatem
thor) found for test com.adobe.aem.guides.wknd.it.tests.CreatePageIT
[main] INFO com.adobe.cq.testing.junit.rules.Page - Created test template in /co
[main] INFO com.adobe.cq.testing.junit.rules.Page - Created test site /content/t
[INFO] Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 4.768 s -
[INFO] Running com.adobe.aem.guides.wknd.it.tests.GetPageIT
[main] INFO com.adobe.cq.testing.junit.rules.ConfigurableInstance - Using LoginT
[main] INFO com.adobe.cq.testing.junit.rules.ConfigurableInstance - Using LoginT
[main] INFO org.apache.sling.testing.junit.rules.instance.ExistingInstanceStatem
thor) found for test com.adobe.aem.guides.wknd.it.tests.GetPageIT
[main] INFO org.apache.sling.testing.junit.rules.instance.ExistingInstanceStatem
ublish) found for test com.adobe.aem.guides.wknd.it.tests.GetPageIT
[INFO] Tests run: 4, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 5.841 s -
[INFO] Results:
[INFO]
[INFO] Tests run: 5, Failures: 0, Errors: 0, Skipped: 0
[INFO]
[INFO] --- maven-failsafe-plugin:2.21.0:verify (default-verify) @ aem-guides-wkn
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 18.100 s
[INFO] Finished at: 2021-01-29T17:49:08+01:00
[INFO] -----
```



# AEM Testing Clients

- Function testing framework for AEM
- Developed and maintained by Adobe
- Open source
- Takes care actions like authentication, session handling, etc.
- Contains multiple clients to perform common actions on AEM,
- Examples:
  - *CQClient* to interact with pages.
  - *ReplicationClient* to perform replication actions.
  - *AbstractComponent* building block for other components.



# Custom test (demo)

The demo test verifies wknd.site Byline component.

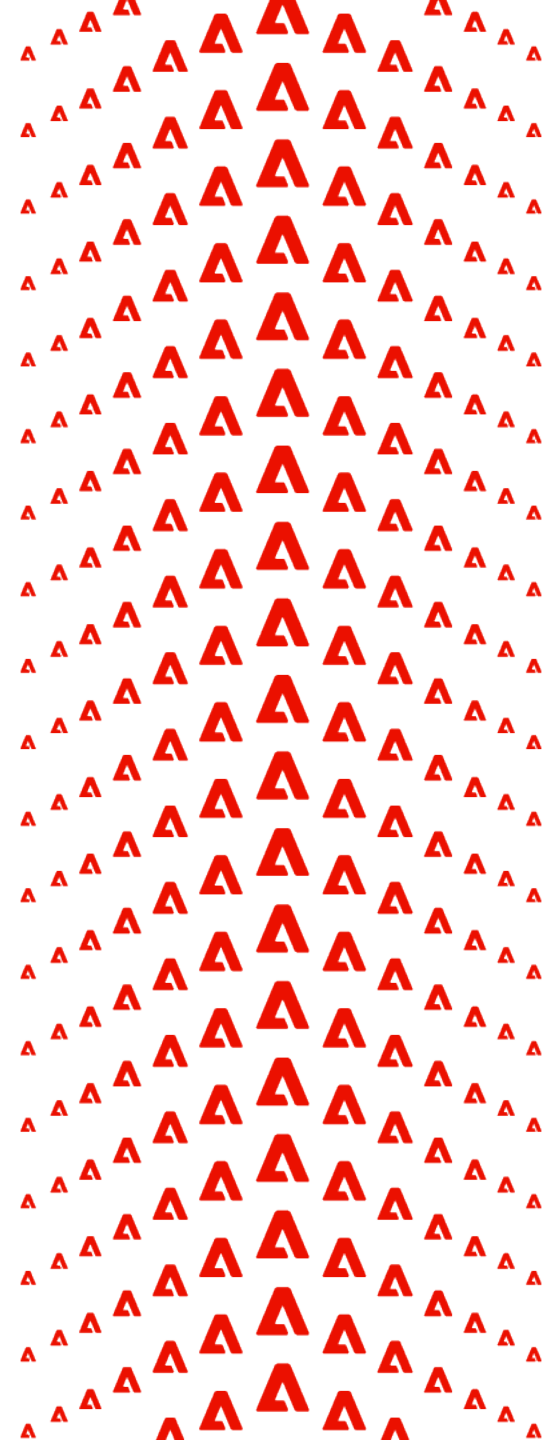
The component only renders content if all properties are set, otherwise it renders empty html.

The tests:

1. Creates a new page,
2. Uploads an image (to be used in the component),
3. Creates a "Byline" component
4. Requests the rendered component
5. Asserts the values are rendered correctly.



# Best practices



# Best practices

- Tests should not rely on existent content, like sample pages
- If the test needs content it needs to be created by the test.
- Idempotent: if the test modifies the state of the AEM instance ( like generate content) it needs to be removed at the end of the test.
- No external dependencies, everything needs to be packed in "jar-with-dependencies"
- All publish instances are deployed behind a dispatcher, make sure the resources are exposed by the dispatcher.
- The tests don't have access to:
  - Osgi console, CRX/DE
  - JCR or Sling API to create nodes

# About: Selenium UI tests



## Custom UI Tests

- Selenium UI tests is a new feature that will be made available soon.
- They will be part of the cloud pipeline.
- UI tests don't replace but extend functional tests.
- Focus on the visual experience for the end user.
- Slower than functional tests.
- Already present in the archetype



# Running UI tests locally

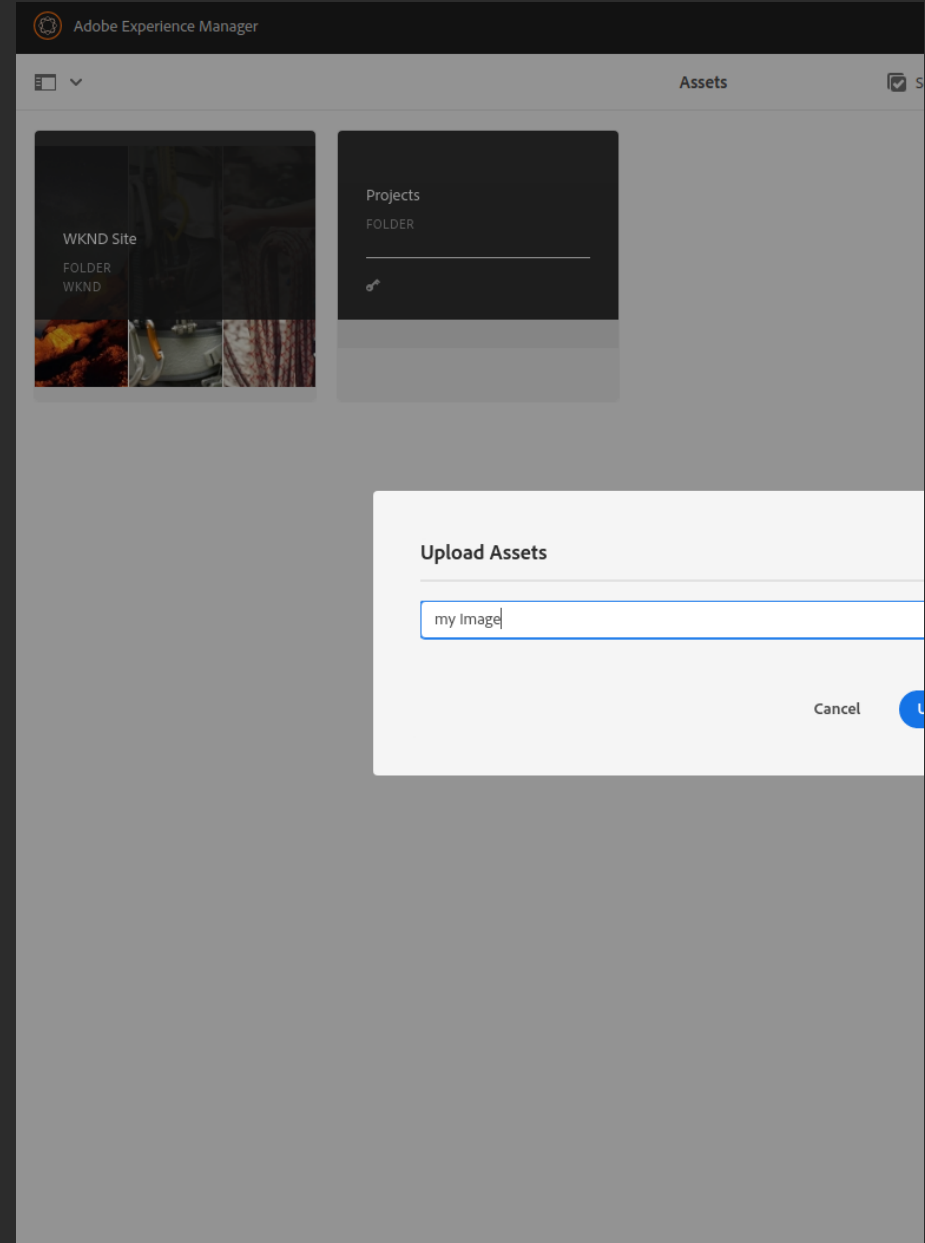
On cloud instances an admin-like user needs to be created.

```
AUTHOR_URL="https://author-pXXXX-eXXXX.adobecloud.com/"
```

```
# (alternative) AUTHOR_URL="http://localhost:4502"
```

```
...
```

```
mvn clean verify \
  -Pui-tests-local-execution \
  -DHEADLESS_BROWSER=false \
  -DSELENIUM_BROWSER=chrome \
  -DAEM_AUTHOR_URL="${AUTHOR_URL}" \
  -DAEM_AUTHOR_USERNAME="${AUTHOR_USER}" \
  -DAEM_AUTHOR_PASSWORD="${AUTHOR_PASSWORD}" \
  -DAEM_PUBLISH_URL="${PUBLISH_URL}" \
  -DAEM_PUBLISH_USERNAME="${PUBLISH_USER}" \
  -DAEM_PUBLISH_PASSWORD="${PUBLISH_PASSWORD}"
```



# References

- Functional testing documentation:  
<https://experienceleague.adobe.com/docs/experience-manager-cloud-service/implementing/using-cloud-manager/test-results/functional-testing.html?lang=en#using-cloud-manager>
- AEM Archetype:  
<https://github.com/adobe/aem-project-archetype>
- AEM testing clients:  
<https://github.com/adobe/aem-testing-clients>
- AEM product tests:  
<https://github.com/adobe/aem-test-samples/tree/aem-cloud/smoke>
- Wknd sample page tutorial:  
<https://experienceleague.adobe.com/docs/experience-manager-learn/getting-started-wknd-tutorial-develop/overview.html?lang=en>
- Wknd Byline component:  
<https://github.com/adobe/aem-guides-wknd/blob/master/core/src/main/java/com/adobe/aem/guides/wknd/core/models/impl/BylineImpl.java>



## Q&A

- Do tests run against an author or an publish instance ?
- Do tests run on dev instances?
- (Your questions here)

