



DEVELOPERS LIVE

# Get ready for the Cloud! - AEM Cloud Service Migration Best Practices

Andreea Moise | Senior Software Engineer @Adobe

# The Benefits

1

## Always On

Best in class service level

No content freeze or downtime

Protection against cloud disasters and failures

Robust on-demand data storage

2

## Always Current

Continuous access to the latest innovations

Automated rolling product updates

3

## Always at Scale

Ensure performance during peak hours of traffic by visitor and users

Auto-scaling (publish & author)

Faster rendering and processing of 100%-pixel quality assets

Microservices for assets ingestion and processing

4

## Always Learning

Always up-to-date with performance enhancements and security updates

Automated corrective updates

Intelligence and machine learning by default

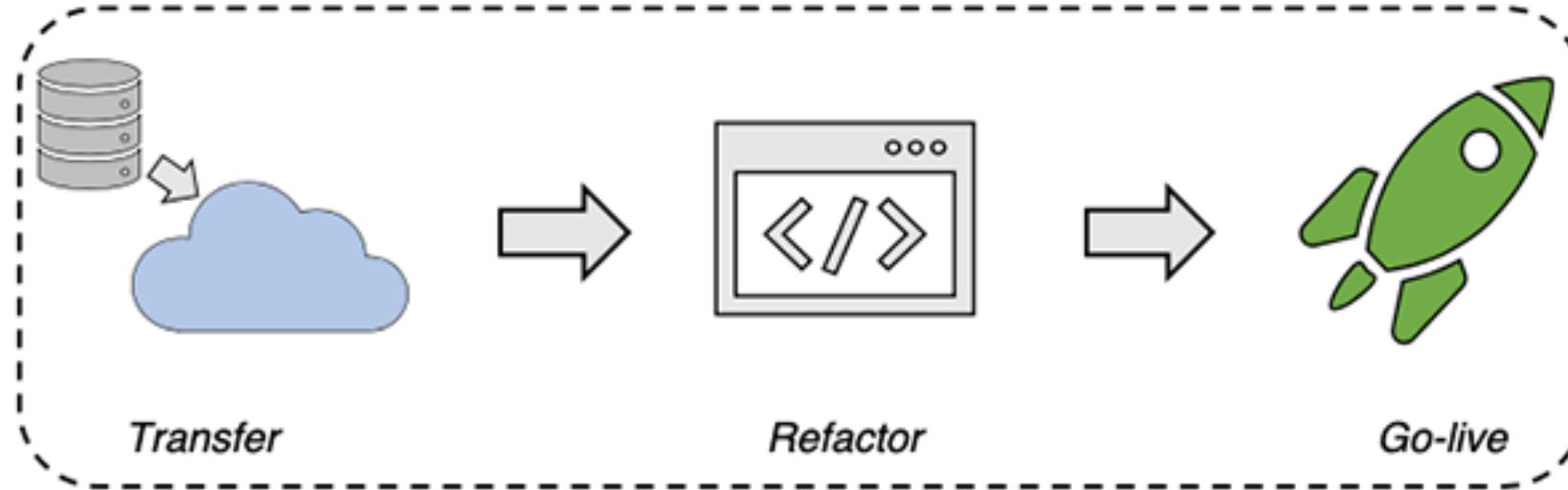
Native access to Sensei intelligence

# The Journey

Planning

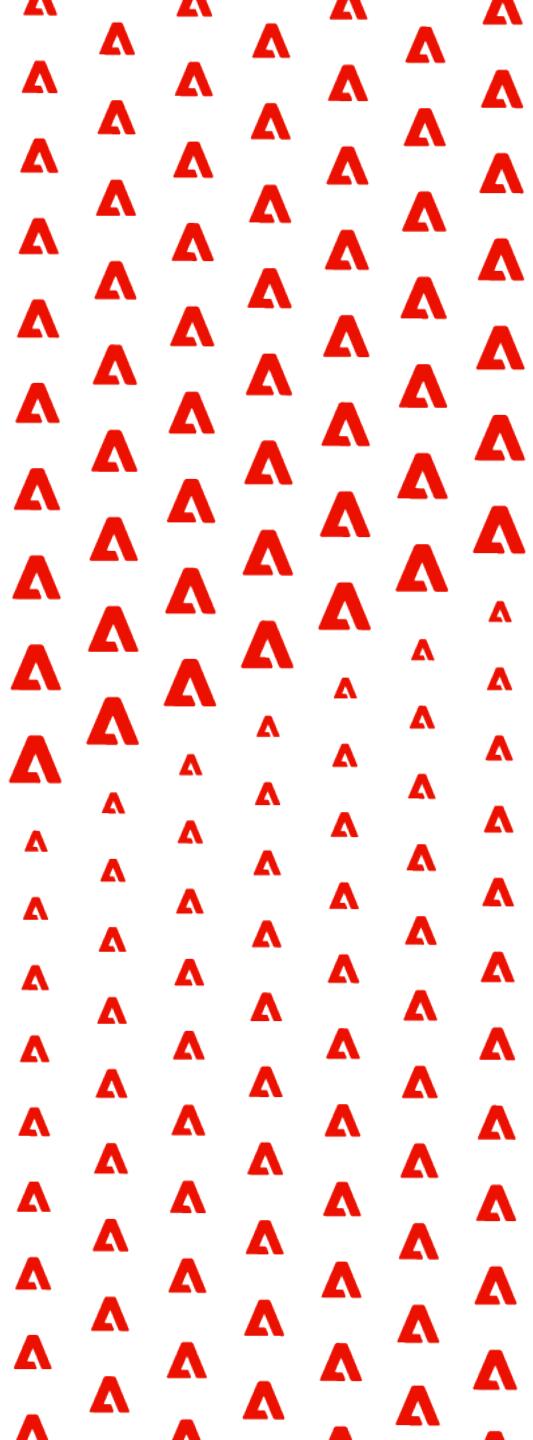


Execution



Post Go-live

# Planning



# Steps

Assess Cloud  
Service Readiness

Review Resource  
Planning

Establish KPIs

# Cloud Service Readiness



**Best Practices Analyzer** - Areas that require refactoring



**Cloud Manager code quality pipeline** - Your current AEM source code against the changes and deprecated features in AEMaaCS

# The Changes in AEMaaCS

CHANGED

- Immutable /apps and /libs
- Repository-based OSGI bundles
- Publish-Side Delivery
- Asset Handling and Delivery

GONE

- Replication Agents
- Classic UI
- Custom Runmodes
- Changes to publish repository

Refresh Report Export to CSV

## Best Practices Analyzer

Help

## Report Overview

Report Time: Thu, 12 Nov 2020 17:03:15 PST

Expiration Time: Thu, 12 Nov 2020 17:03:25 PST

Generation Time Period: 0 hr 00 min 20 sec

Finding Count: 31

This report provides a summary of findings from the Best Practices Analyzer. The following sections display the findings of the analysis, organized by type and presented with a description and the number of findings for each type.

The findings below include general information about the AEM instance and potential deviations from AEM best practices. Please note that some of these findings and severity levels only apply to assessing the readiness to move to AEM as a Cloud Service.

Name	Info	Advisory	Major	Critical
AEM System Overview	5	0	0	0
Legacy User Interface - Custom Component	0	0	14	0
Legacy User Interface - Classic Dialog	0	0	0	2
Legacy User Interface - Coral2 Dialog	0	0	4	0
replication.agent_forward.replication_	0	0	4	0
Replication Agent	0	0	0	1
Upgrade Misconfiguration Issue	0	0	1	0

# Custom Code Quality rules - SonarQube

STAGE DEPLOYMENT

- Validation

Your Pipeline has been verified against a set of pre-defined rules.

 Passed (Finished: July 23, 2020 10:01:49 AM GMT+3)

- Build & Unit Testing

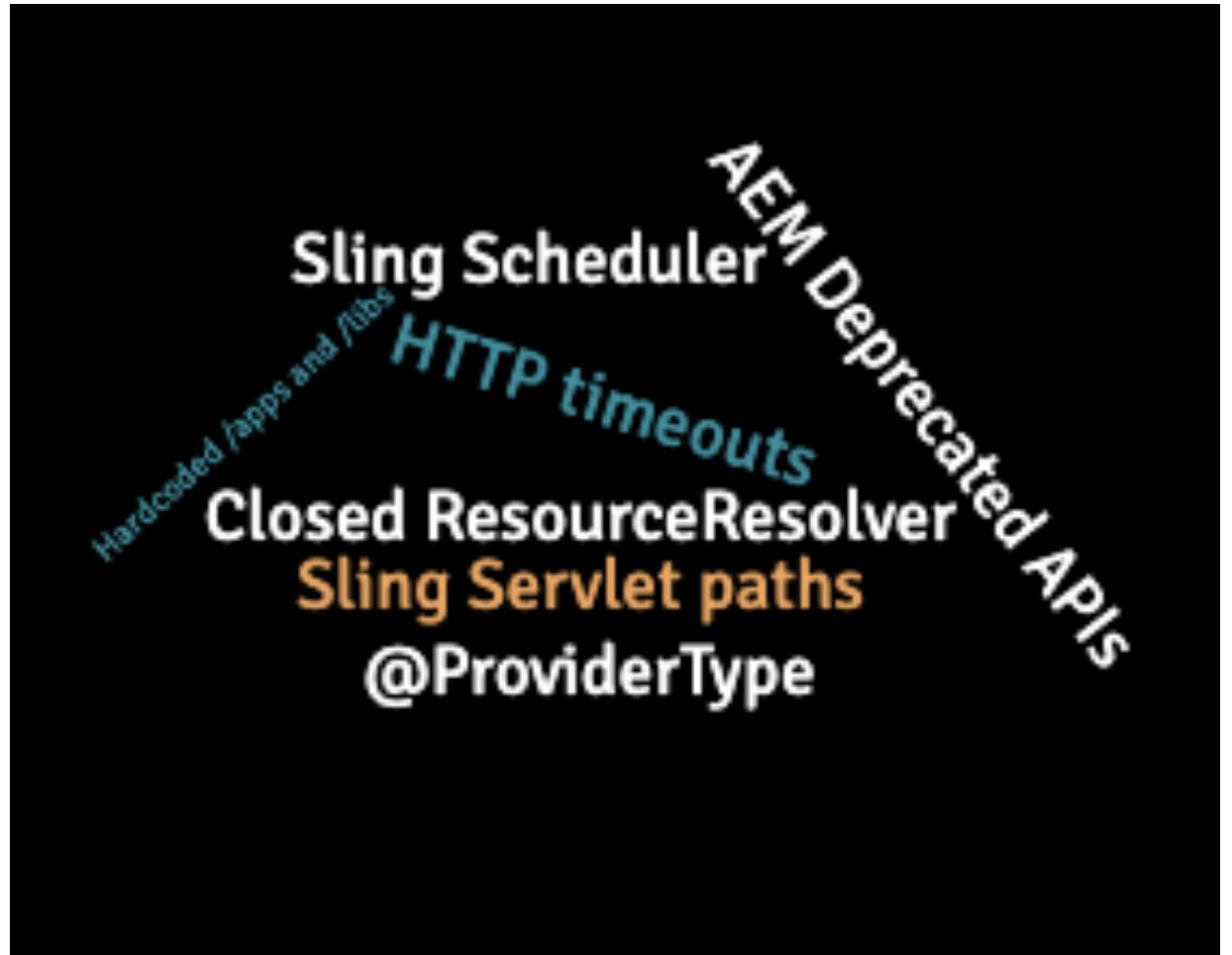
Your application code has been tested for quality and compiled into build artifacts

BRANCH: MASTER

VERSION:2020.526.121154.0000102041

[Download Log](#)

 Succeeded (Finished: July 23, 2020 10:07:50 AM GMT+3)



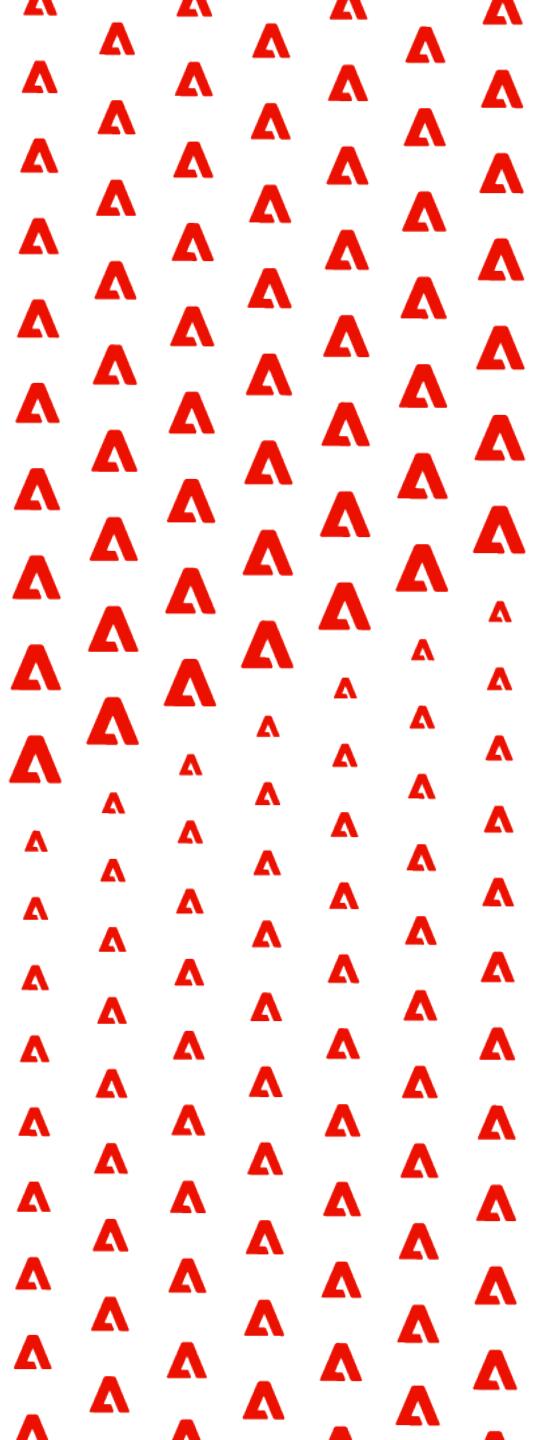
# Custom Code Quality rules - OakPAL



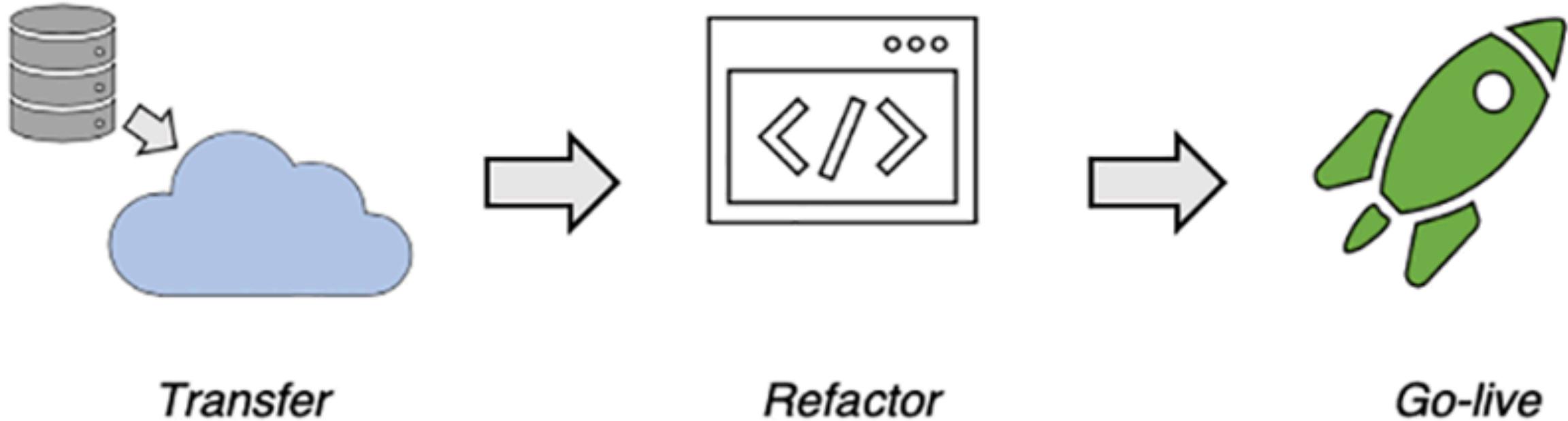
*ClassicUI  
MutableContent  
nibs*

**Duplicate OSGI Configs**  
**Reverse Replication**  
**Touch UI Dialogs**  
**Immutable Content**

# Execution



# Steps

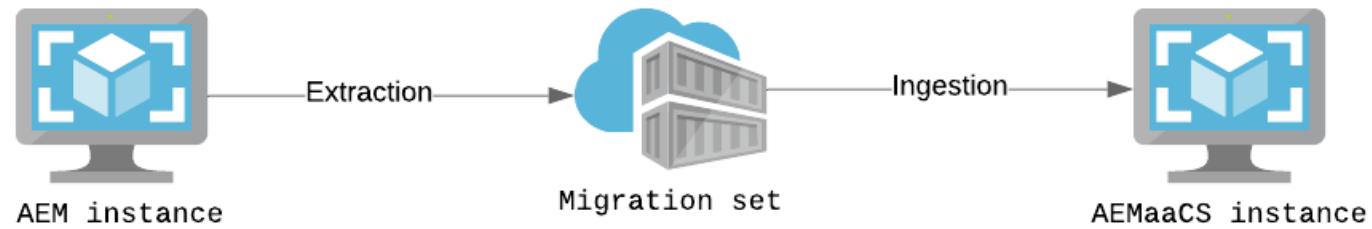


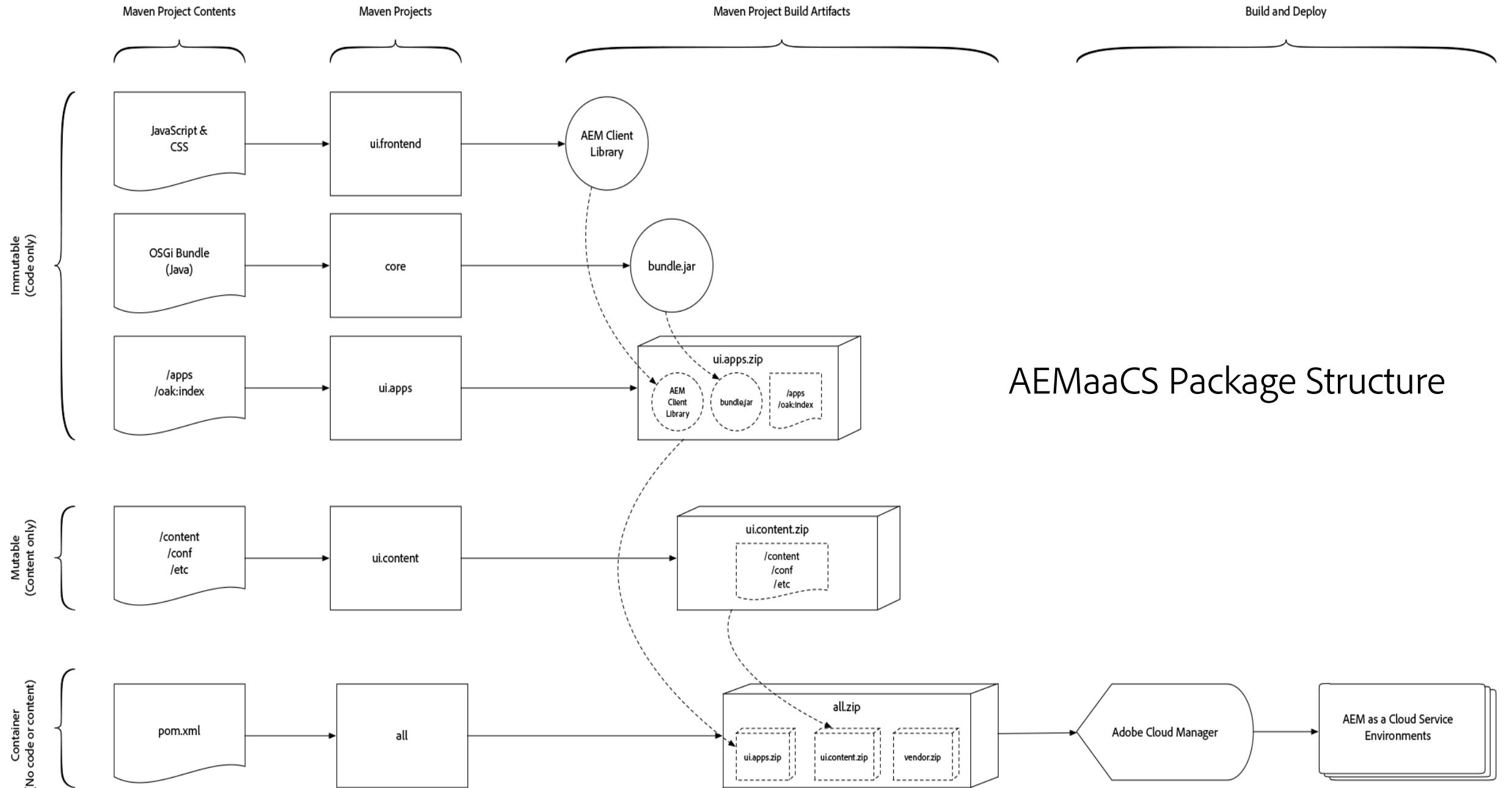
# Content Transfer

- **WHAT?**

Existing content and principals (users or groups) from a source AEM instance (on-premise or AMS) to the target AEMaaCS instance

- **HOW?**





# Code Refactoring - Development Guidelines



Classic UI  
State CRXDE lite  
Background Tasks  
Logs  
Reverse Replication  
Observation  
Long Running Jobs  
Native Binaries  
Replication Agents  
Streaming Binaries  
Outgoing HTTP Connections

# Code Refactoring – Unified Experience Tools

## Unified Experience

- Repository Modernizer
- Index Convertor
- AEM Dispatcher Converter

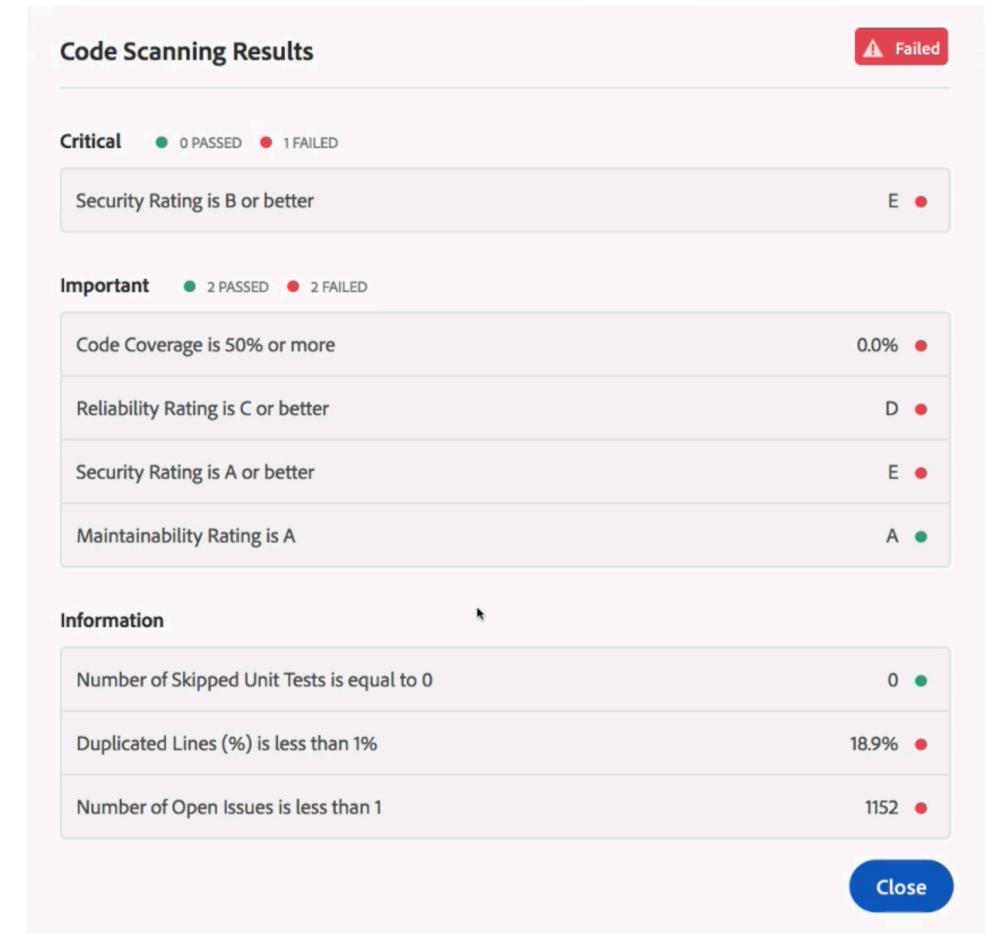
## Asset Workflow Migration

## AEM Modernization Tools

- Static templates to editable templates
- Design configurations to policies
- Foundation Components to Core Components
- Classic UI to Touch-Enabled UI

# Best Practices for Code Deployment and Testing

- Security rating < B
- Reliability rating < C
- Maintainability rating < A
- Coverage < 50%
- Skipped Unit Tests > 1
- Open Issues > 0
- Duplicated Lines > 1%
- Cloud Service compatibility > 0



# Best Practices for Code Deployment and Testing

## Custom Functional Testing

- packaged as a separate JAR file
- class names of the actual tests to be executed must end in IT

### STAGE TESTING

#### Product Functional Testing

Execution of the product functional tests is complete.

[Download Log](#)

 Passed (Finished: July 23, 2020 10:55:26 AM GMT+3)

#### Custom Functional Testing

Execution of the custom functional tests is complete.

[Download Log](#)

 Passed (Finished: July 23, 2020 10:55:36 AM GMT+3)

# Best Practices for Code Deployment and Testing

```
<!-- Create self-contained jar with dependencies -->
<plugin>
  <groupId>org.apache.maven.plugins</groupId>
  <artifactId>maven-assembly-plugin</artifactId>
  <version>3.1.0</version>
  <configuration>
    <descriptorRefs>
      <descriptorRef>jar-with-dependencies</descriptorRef>
    </descriptorRefs>
    <archive>
      <manifestEntries>
        <Cloud-Manager-TestType>integration-test</Cloud-Manager-TestType>
      </manifestEntries>
    </archive>
    ...
  </configuration>
</plugin>
```

# Best Practices for Code Deployment and Testing

## Experience Audit Testing

- Lighthouse checks
- Tests
- Scores and Score Variability
- Guidance

### STAGE TESTING

#### • Product Functional Testing

Execution of the product functional tests is complete.

[Download Log](#)

✓ Passed (Finished: January 27, 2021 9:31:08 PM GMT+2)

#### • Custom Functional Testing

Execution of the custom functional tests is complete.

[Download Log](#)

✓ Passed (Finished: January 27, 2021 9:31:22 PM GMT+2)

#### • Custom UI Testing

Execution of the UI tests is complete.

[Download Log](#)

✓ Passed (Finished: January 27, 2021 9:31:36 PM GMT+2)

#### • Experience Audit

Your site has been audited against a set of rules to validate content quality, performance, and user experience.

No Summary Available

✓ Passed (Finished: January 27, 2021 9:31:48 PM GMT+2)

## Experience Audit

 Passed

### Information

● 4 PASSED ● 0 FAILED

Current

Change

- Best Practices Score is 80 or above for all audited pages 100 ● N/A

- Performance Score is 80 or above for all audited pages 90 ● N/A

● 1 PASSED ● 0 FAILED

Current

Details

Change

<https://publish-p14253-e43686.adobeacmcloud.net/>

90 ●



N/A

- Accessibility Score is 80 or above for all audited pages 86 ● N/A

- SEO Score is 80 or above for all audited pages 89 ● N/A

Generated by Lighthouse 6.1.0

Close

## Experience Audit

[Back](#)[Print](#)

<https://publish-p53603-e106504-cmdev.adobeacmcloud.net/>



Performance



Accessibility



Best Practices



SEO

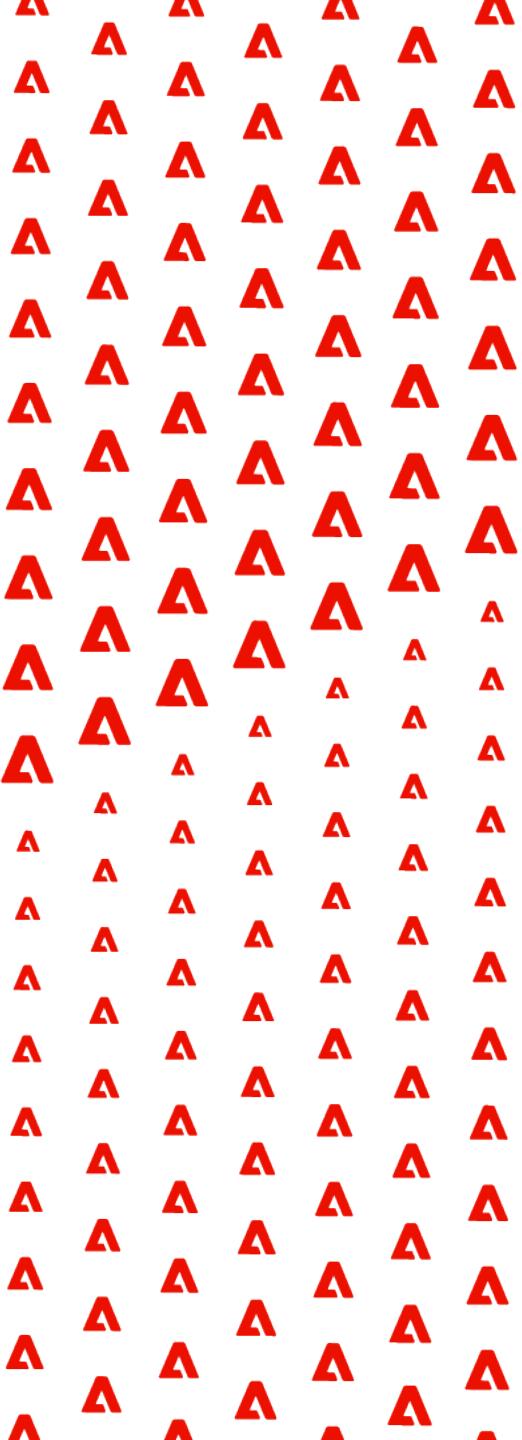


Progressive  
Web App

■ 0–49 ■ 50–89 ■ 90–100



# Go-Live



# Best Practices for Go-Live Preparations

Schedule code and content freeze period

Perform final content top-up

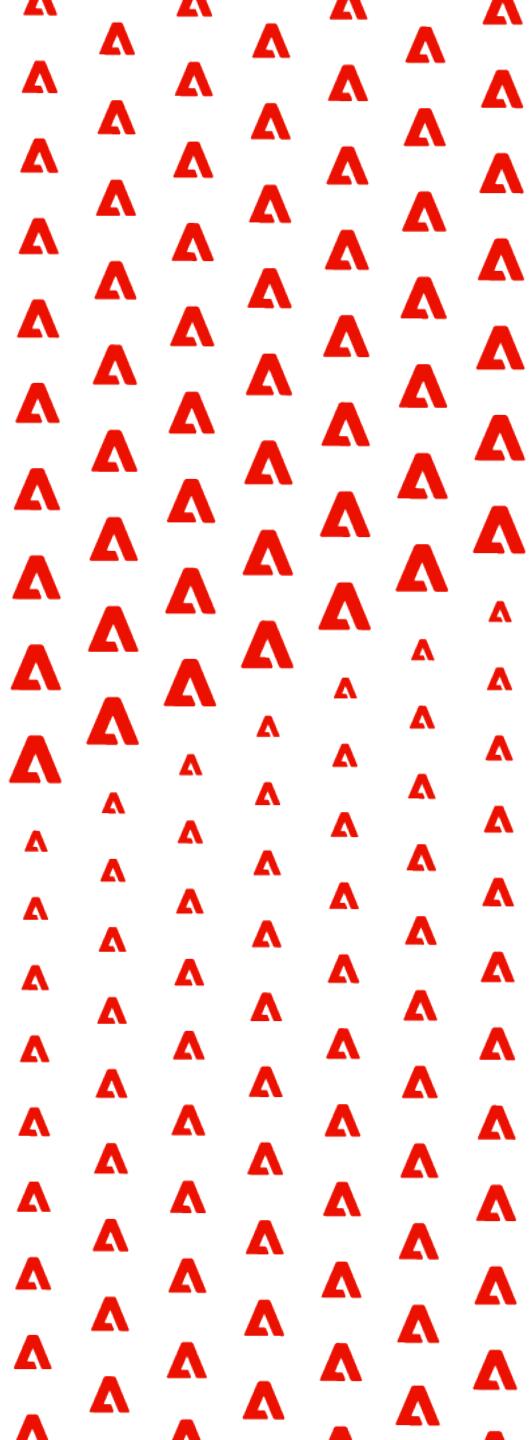
Complete testing iterations

Run performance and security tests

Always create a fallback plan

Cut-Over

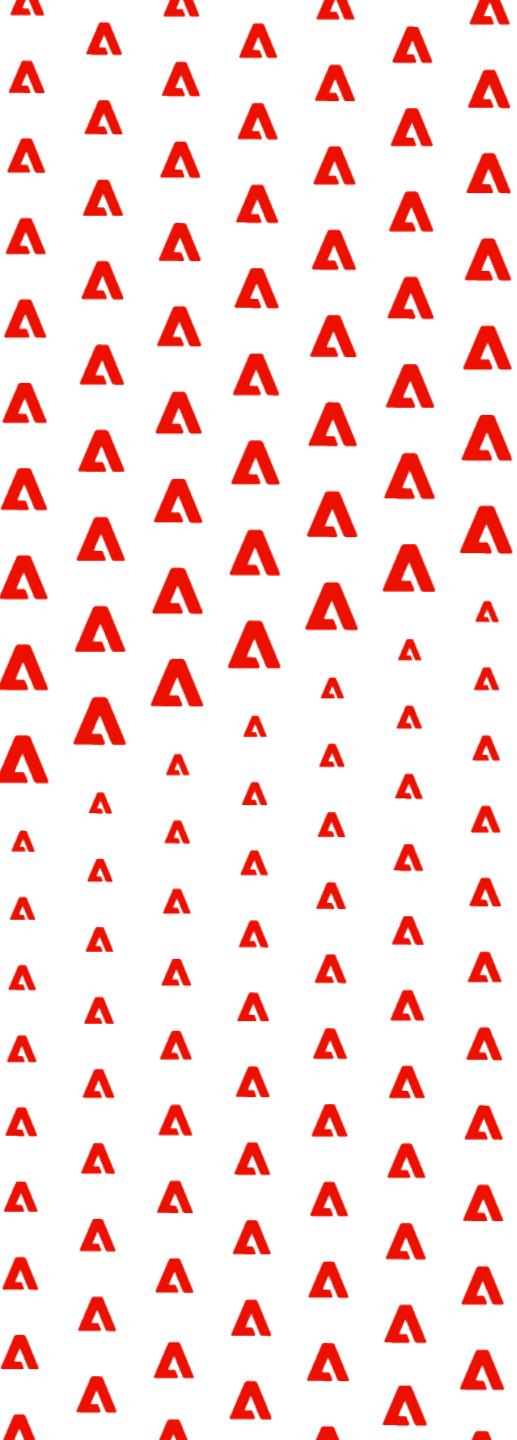
# Summary



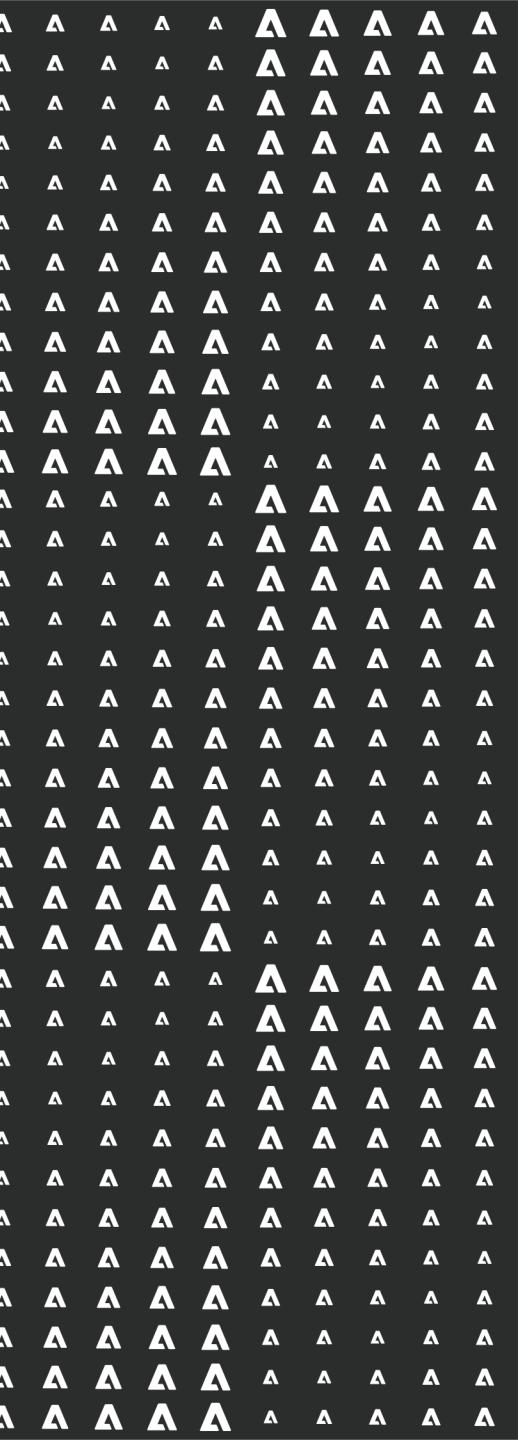
# What we learned?

- How to plan a migration to Cloud Service
- How to execute it by
  - Content transfer
  - Codebase refactoring
- How to have a successful Go-Live

# Q&A



# Appendix



# Custom Code Quality rules - SonarQube

- HTTP requests should always have socket and connect timeouts
- Product APIs annotated with @ProviderType should not be implemented or extended by customers
- ResourceResolver objects should always be closed
- Do not use Sling servlet paths to register servlets
- Logging and exception handling rules
- Avoid Hardcoded /apps and /libs Paths
- Sling Scheduler Should Not Be Used
- AEM Deprecated APIs Should Not Be Used

# Custom Code Quality rules - OakPAL

- Customer Packages Should Not Create or Modify Nodes Under /libs
- Packages Should Not Contain Duplicate OSGi Configurations
- Config and Install Folders Should Only Contain OSGi Nodes
- Packages Should Not Overlap
- Default Authoring Mode Should Not Be Classic UI
- Components With Dialogs Should Have Touch UI Dialogs
- Packages Should Not Mix Mutable and Immutable Content
- Reverse Replication Agents Should Not Be Used

# Code Refactoring - Development Guidelines 1/2

- State in Memory
- State on the Filesystem
- Observation
- Background Tasks and Long Running Jobs
- Outgoing HTTP Connections
- No Classic UI Customizations

# Code Refactoring - Development Guidelines 2/2

- Avoid Native Binaries
- No Streaming Binaries through AEM as a Cloud Service
- No Reverse Replication Agents
- Forward Replication Agents Might Need to be Ported
- AEMaaCS logs available through Cloud Manager
- CRXDE lite available on the development environment but not on stage or production

# Resources

- [AEMaaCS Architecture](#)
- [AEMaaCS Archetype](#)
- [Asset Workflow Migration Tool](#)
- [Unified Experience Tool](#)
- [AEM Modernization Tools](#)
- <https://github.com/adobe/aem-testing-clients>
- <https://github.com/adobe/aem-test-samples>

