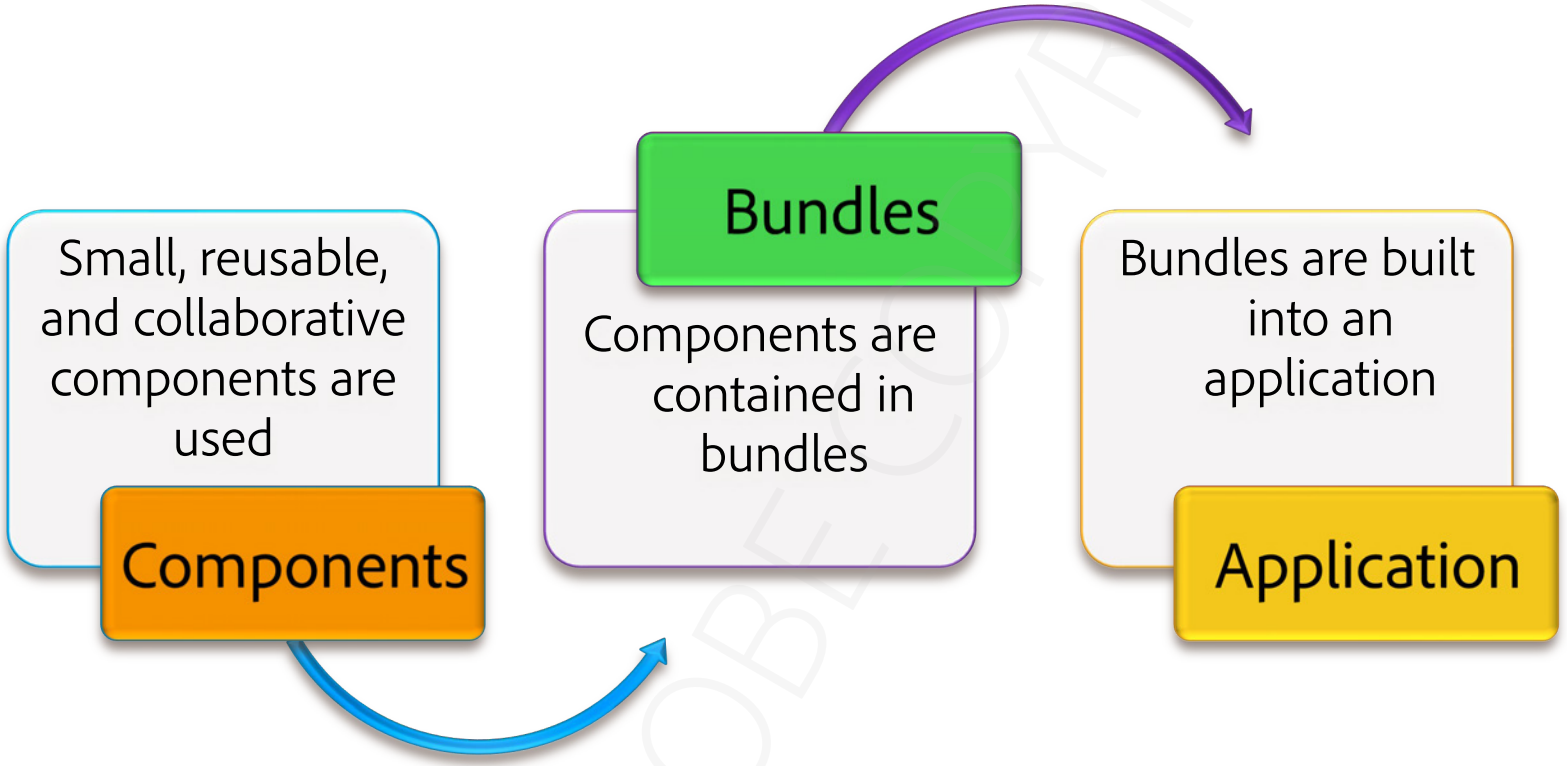
Configure AEM Cloud Service

Agenda:

- OSGi configurations
- Most commonly used OSGi configurations
- Creating an OSGi configuration
- Role of System Users
- Repository Initializer Service and its language

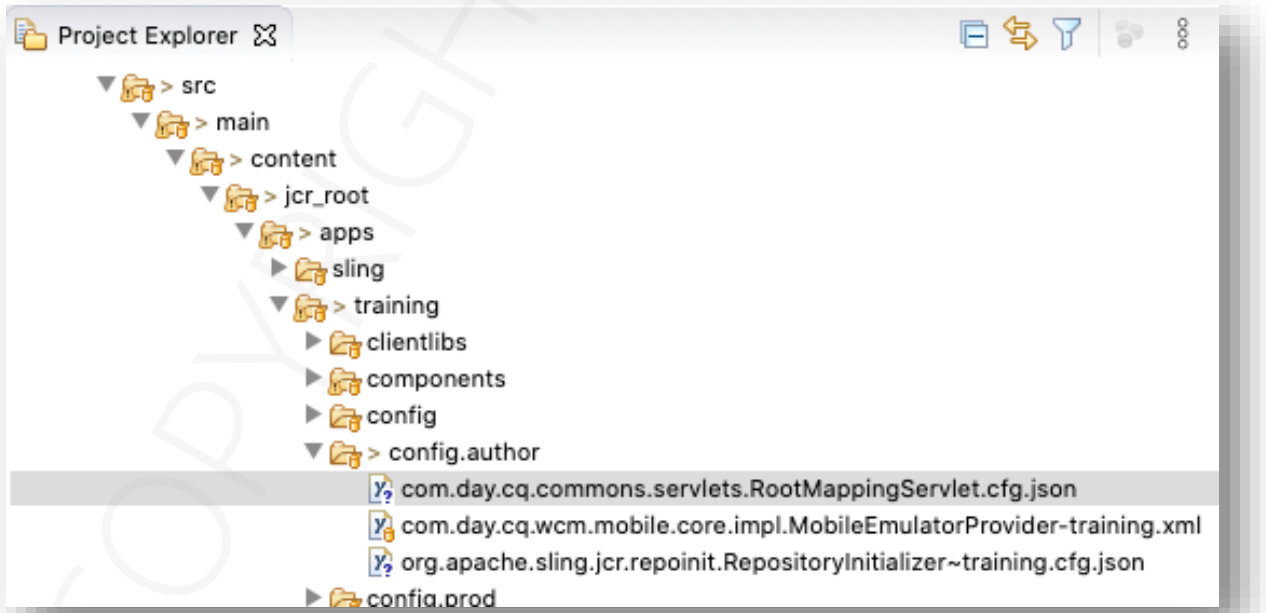
OSGi

Fundamental element in the technology stack of Adobe Experience Manager (AEM)



Creating OSGi configurations

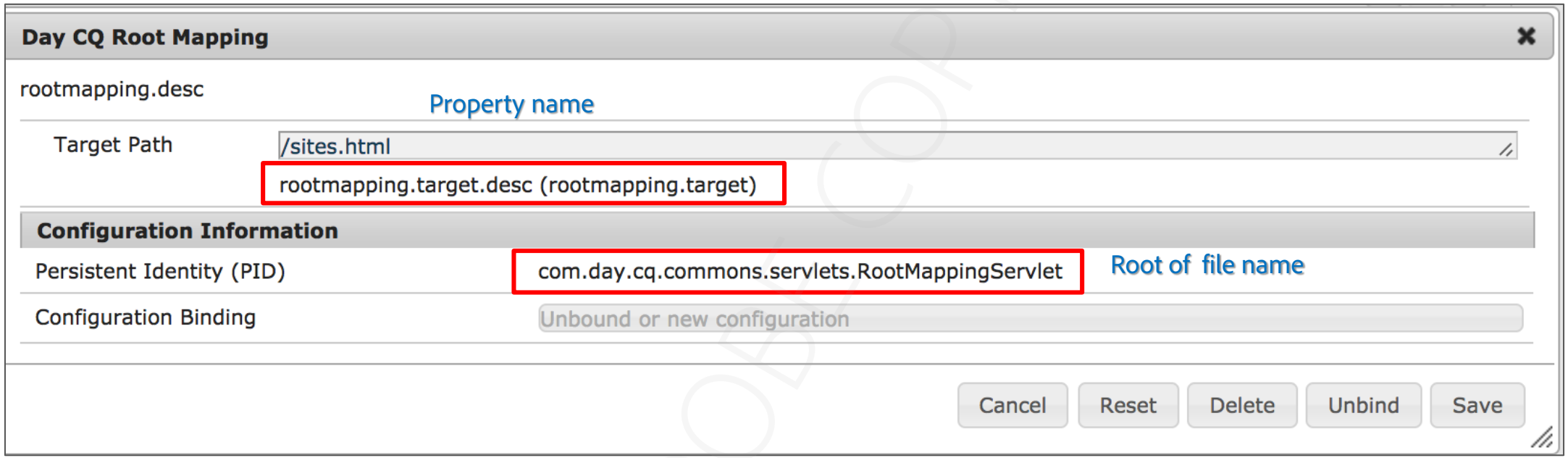
- Configuration files are created in AEM Maven Project
- Configurations are considered code
  - Enables configurations to be pushed into Cloud Manager
  - All AEM services will contain these configurations
  - Like code, OSGi configs can only be updated through a CICD pipeline



Determining property name and file name to use

From the Web Console > OSGi > Configurations

- Find the desired configuration
- Obtain the Property name and Node name-PID to use in your configuration node.

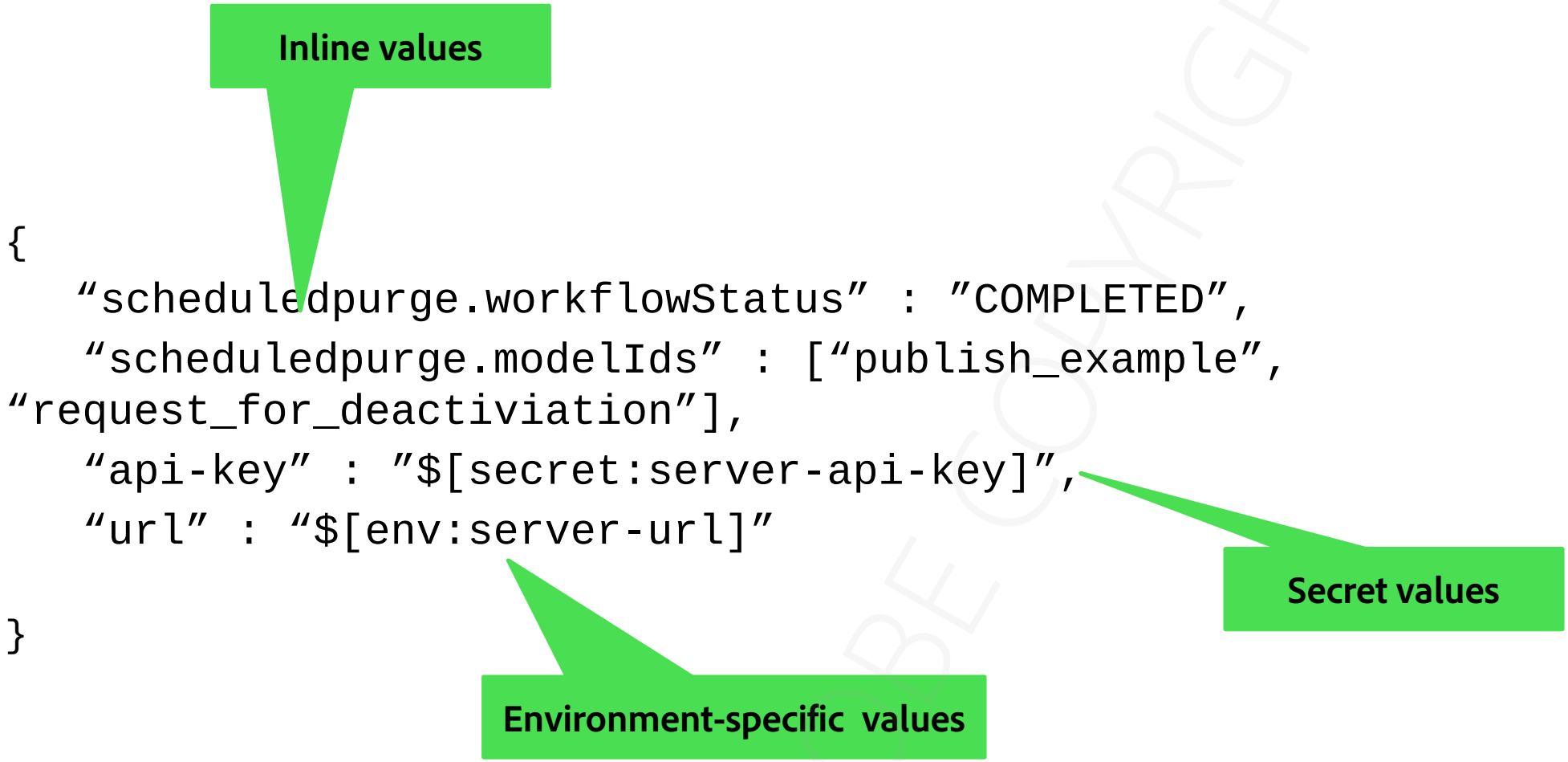


OSGi configuration file contents

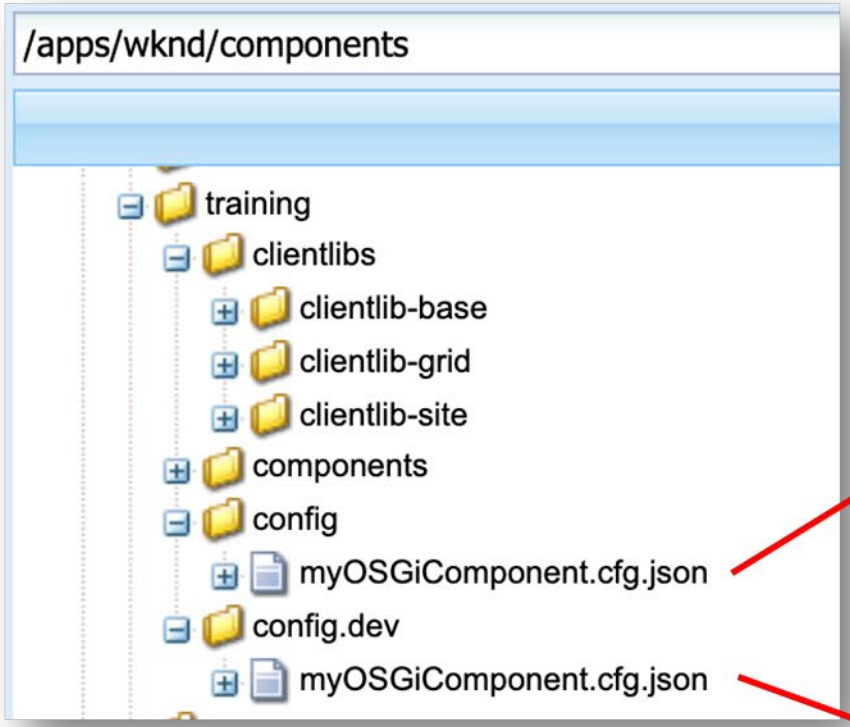
/apps/<my-app>/config/com.day.cq.commons.servlets.RootMappingServlet.cfg.json

```
{

   "rootmapping.target" : "/sites.html"

}
```

OSGi configuration values



Environment-specific configurations



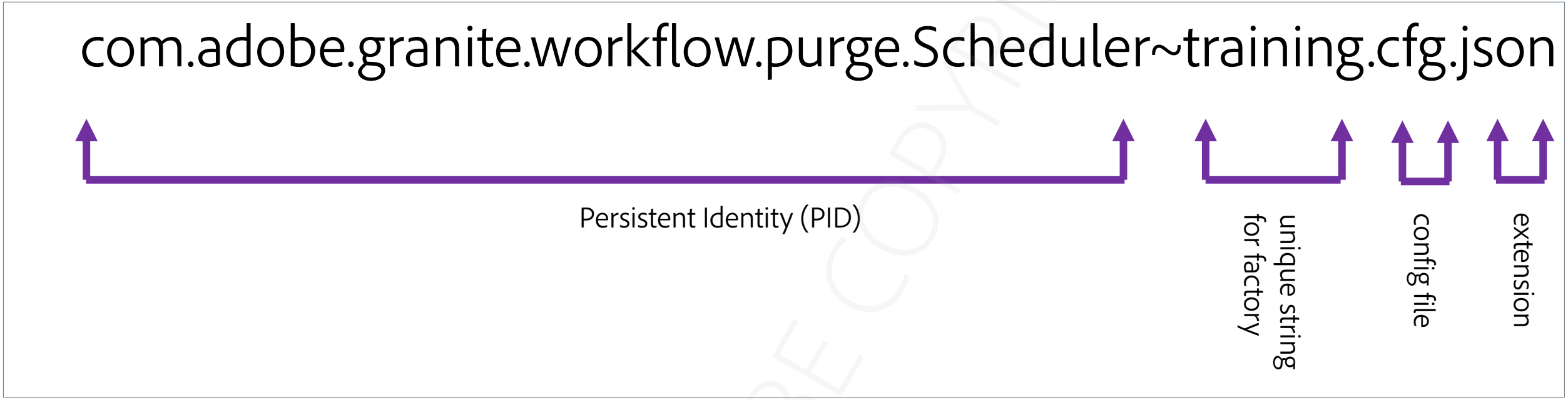
Home | myOSGiComponent.cfg.j

```
{
    "my_var1" : "blue",
    "my_var2" : "green",
    "my_var3" : "yellow"
}
```

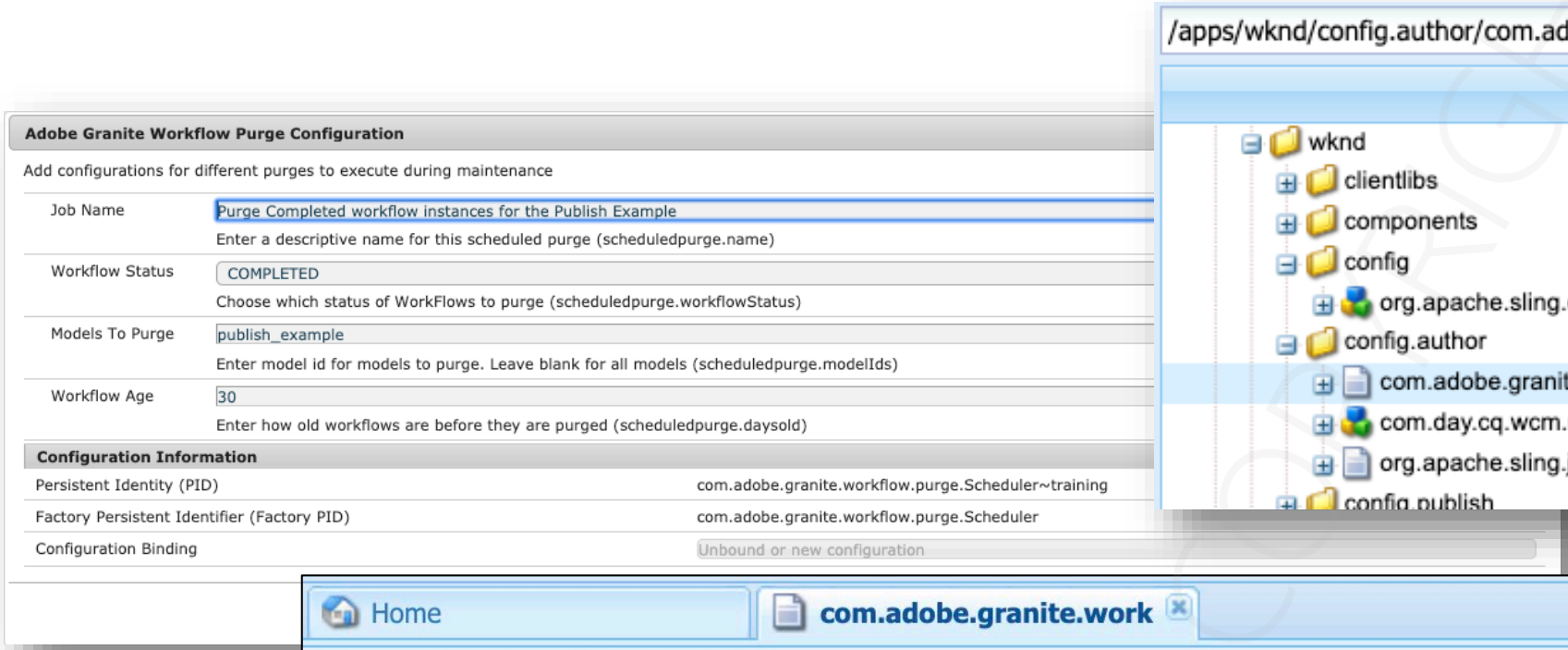
Home | myOSGiComponent.cfg.j

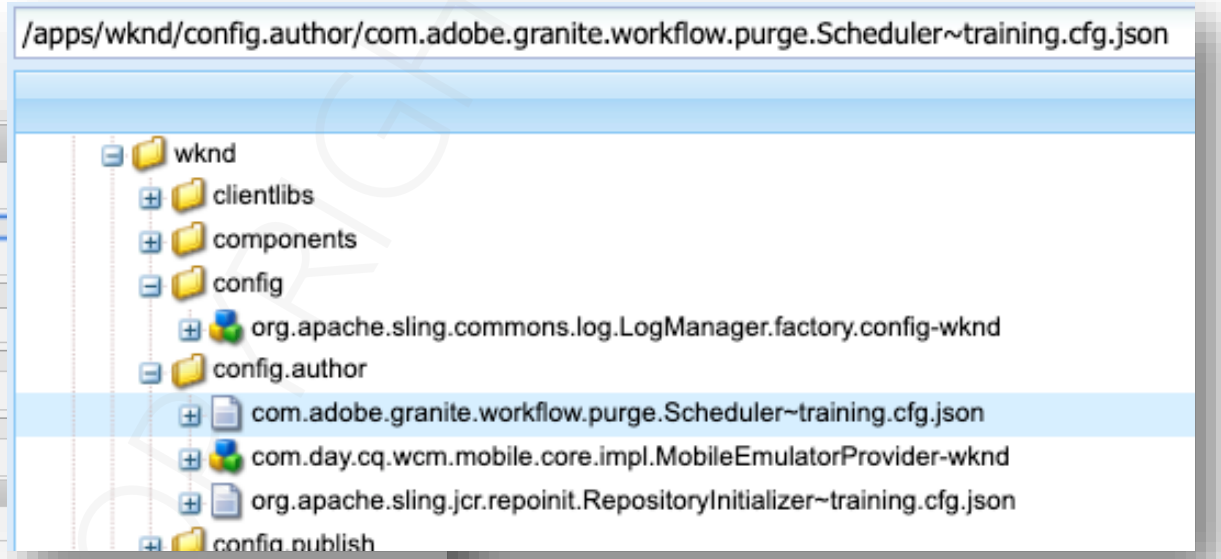
```
{
    "my_var1" : "$[env:myvar_1]",
    "my_var2" : "green",
    "my_var3" : "purple"
}
```

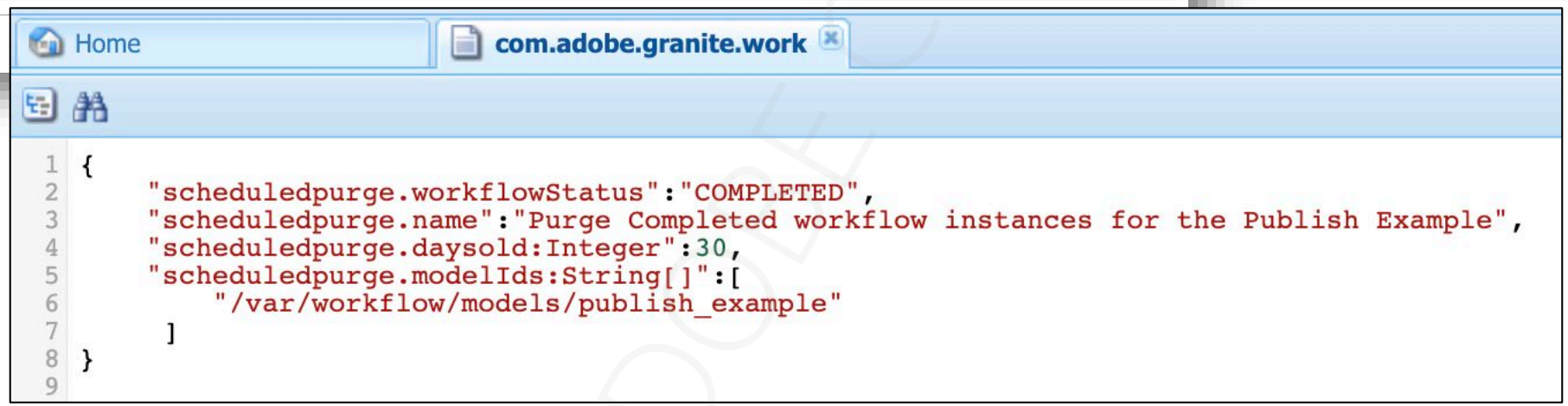
OSGi configuration files for factory services



Workflow purge maintenance task





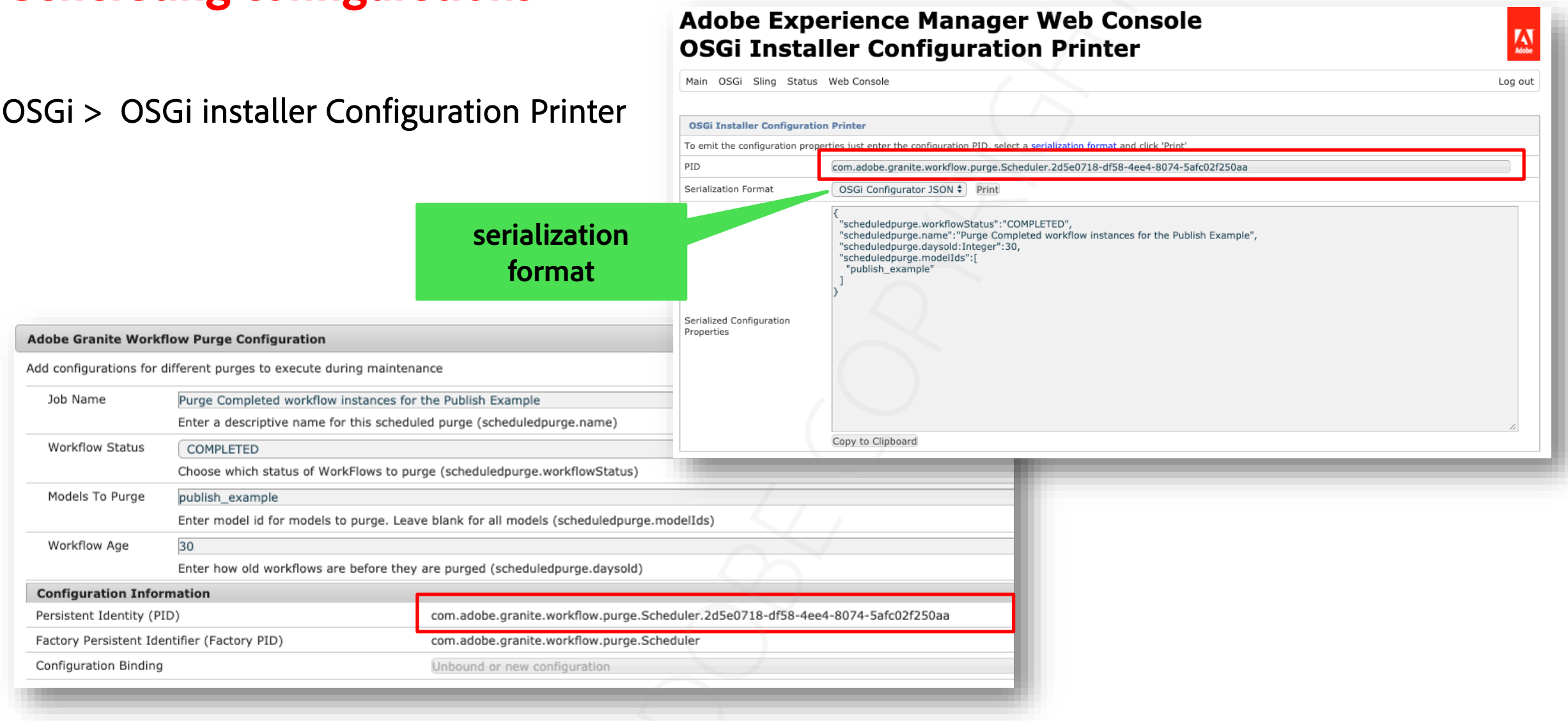


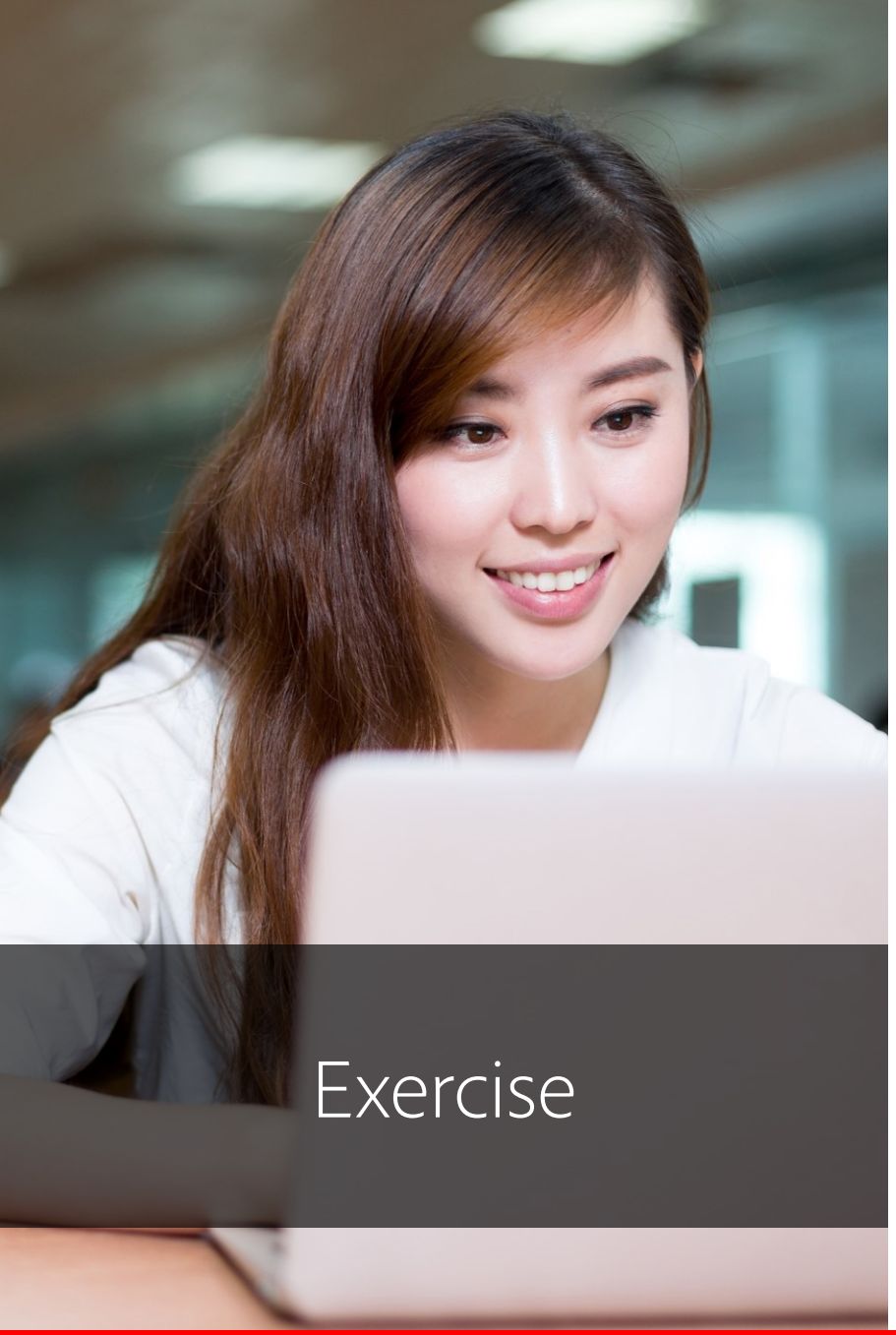
Common OSGi configurations

- Day CQ WCM Undo
- Day CQ Root Mapping
- Maintenance Tasks
    - Workflow Purge
    - Ad-hoc Task Purge
    - Project Purge

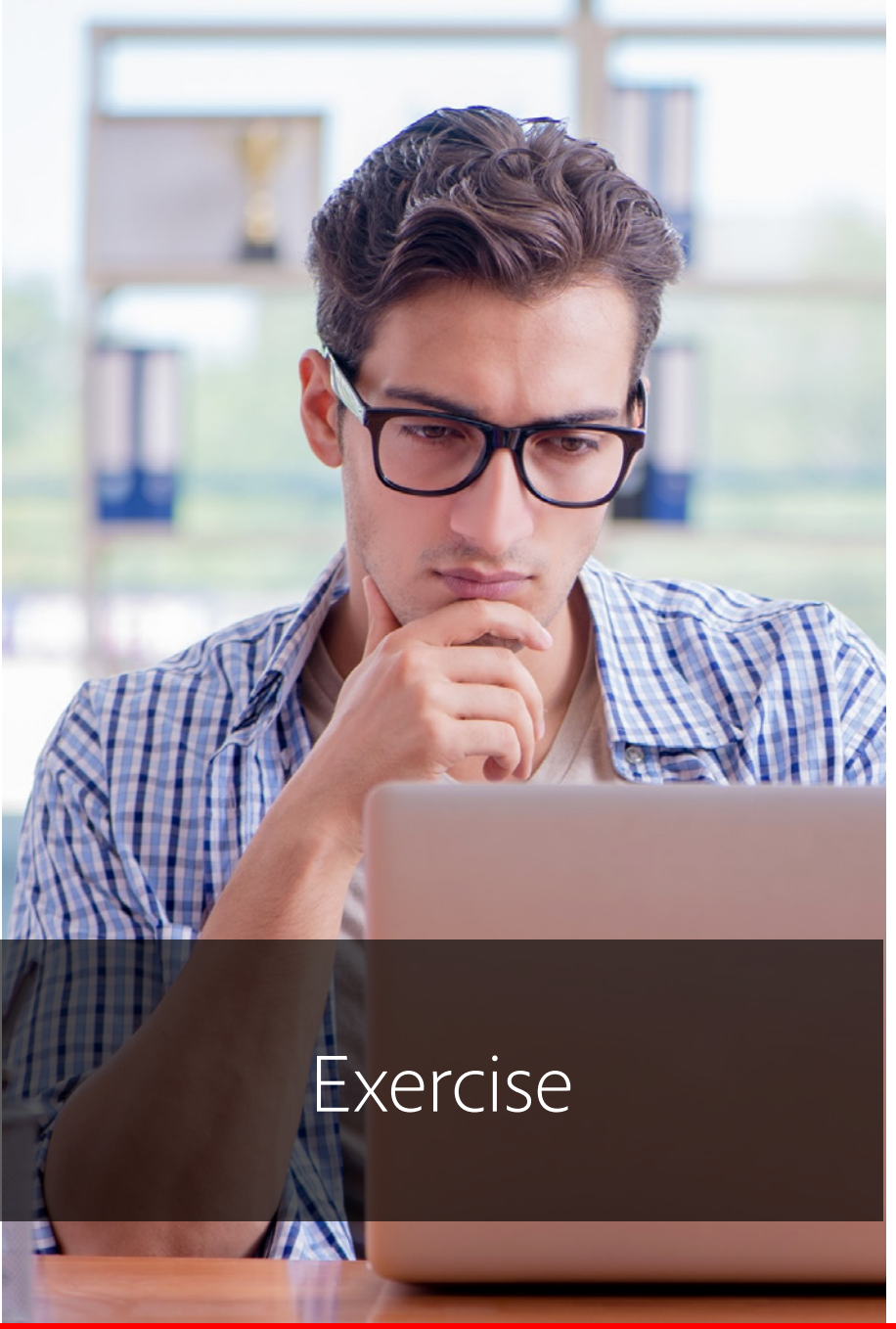
Generating configurations

OSGi > OSGi installer Configuration Printer





Exercise 1: Use the OSGi Installer Configuration Printer



Exercise 2: Configure AEM

Permission service | Sling service

- Piece or collection of functionality
  - Examples of services might include
    - Sling queuing system
    - Tenant Administration
    - Event Handlers
    - Custom Workflow processes
- Identified by a unique service name
- Implemented as a component in an OSGi bundle
- Named by the bundles providing them

Permission service | Service permission context

Adobe recommends:

- Run a service within the calling user's permission context when possible

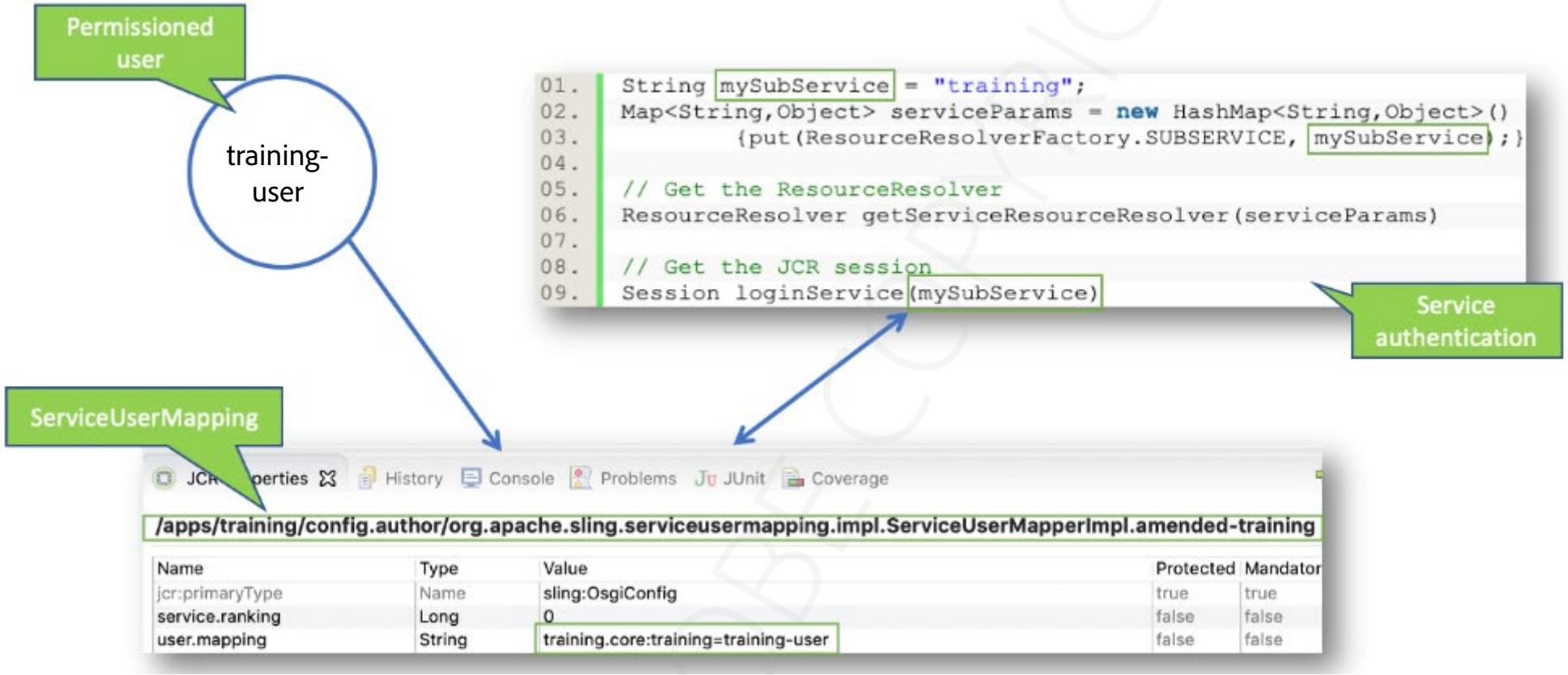
Background tasks

- Typically not called.
- Not associated with a specific user

Apache Sling provides a three-part mechanism for service authentication and authorization:

- Service user
- Service ID
- ServiceUserMapper

Permission service | Service to user mapping



Repository initialization

Some elements of an application need to be loaded first to install the application successfully

- Create/delete/disable service users
- Create/delete users and groups
- Add access control lists (ACLs)
- Add path (for example, for root folder structures)
- Add CNDs (nodetype definitions)
- Add repository namespaces

SlingRepositoryInitializer runs the code before the **SlingRepository** service is registered

- Enables initialization
- Controlled by use of repoinit script
- Useful for initialization and content migration purposes

Repository Initialization Language (repoinit)

Example language

```
create service user user1, u-ser_2
set ACL on /libs,/apps
    allow jcr:read for user1,u-ser_2

    deny jcr:write for u-ser_2
    deny jcr:lockManagement for user1
    remove jcr:understand,some:other for u3
end

create service user bob_the_service

set ACL on /tmp
    allow some:otherPrivilege for bob_the_service
end
```

```
create group since124_A
create group since124_B with path /path_B
delete group since124_C
```

```
add user1,user2 to group grpA
remove user3,user5 from group grpB
```

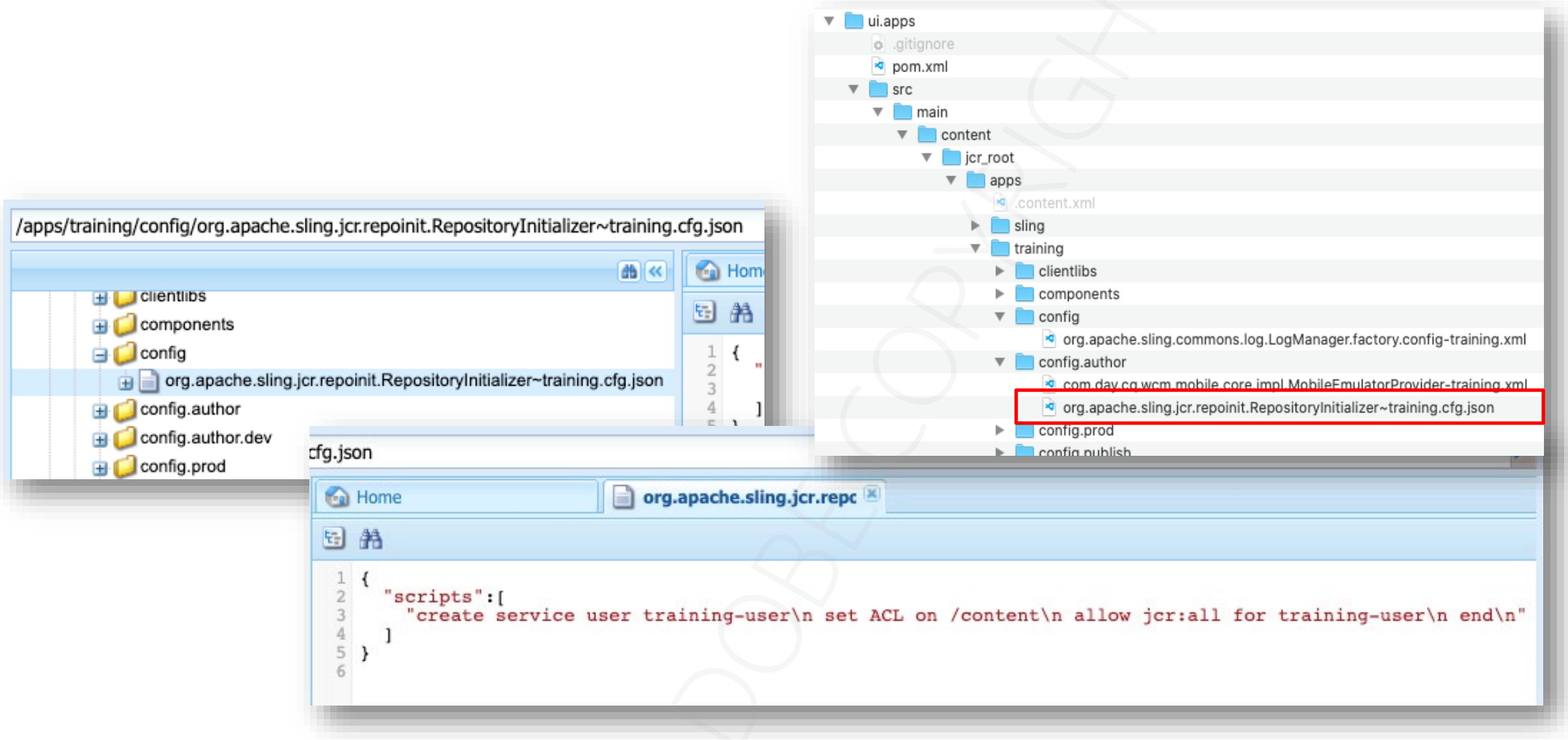
```
set ACL on home(alice)
  allow jcr:read for alice, bob, carol
end

set ACL for bob
  allow jcr:read on home(alice), /another/path, home(larry)
end
```

Repoinit versus mutable content package

repoinit	Mutable Content Package
Creates resources at startup	Resources are created after startup
Safe instruction set: with few exceptions, supported operations are additive	Removal of something is explicit
Fast and atomic operations	Performance highly dependent on the structures covered by a filter

During application deployment, *repoinit* statements are executed independent of the installation of any content packages

Repoinit configuration



Caution: Syntax

Apache Sling Repository Initializer Factory

Initializes the JCR content repository using repoinit statements.

References

References to the source text that provides repoinit statements. format is a raw URL. (references)

Scripts

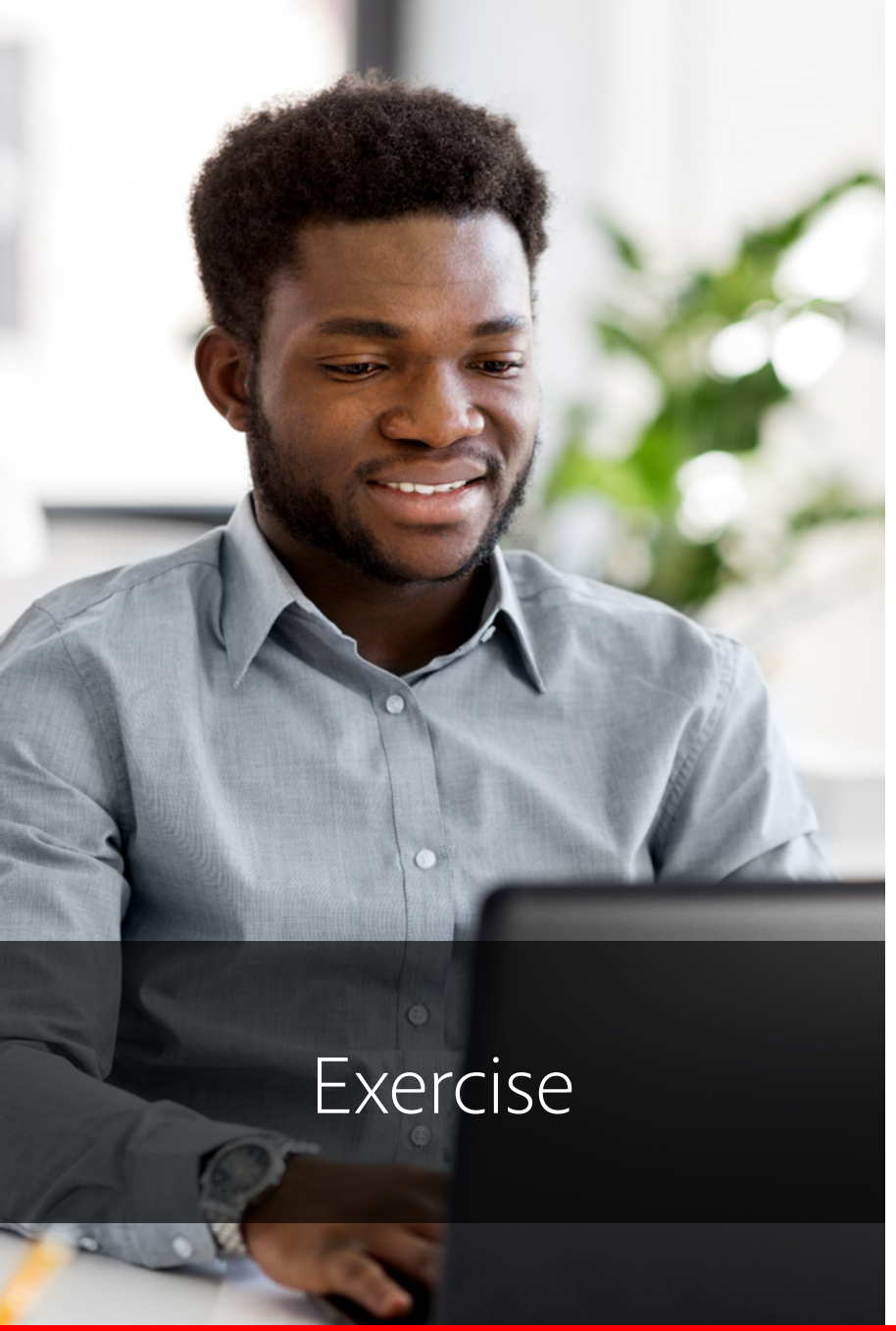
```
create service user test-user-1
Set ACL on /content
allow jcr:read for test-user-1
end
create path /content/example.com(sling:Folder)
create path /content/example.com(sling:Folder mixin mix:referenceable,mix:shareable)
```

Contents of a repo init script. (scripts)

Configuration Information

Persistent Identity (PID)	[Temporary PID replaced by real PID upon save]
Factory Persistent Identifier (Factory PID)	org.apache.sling.jcr.repoinit.RepositoryInitializer
Configuration Bindin	


```
{
 "scripts":[
  "create service user test-user-1\r\nSet ACL on /content\r\nallow jcr:read for test-user-1\r\nend\r\ncreate path
/content/example.com(sling:Folder)\r\ncreate path /content/example.com(sling:Folder mixin mix:referenceable,mix:shareable)"
 ],
 "references":[
  ""
 ]
}
```



Exercise 3: Use repoinit statements to define service user ‘training-user’

Key takeaways

- OSGi
  - Fundamental element in the technology stack of AEM
- Common OSGi configurations
  - Day CQ WCM Undo
  - Day CQ Root Mapping
  - Maintenance Tasks
- Sling Service
  - Piece or collection of functionality
- repoinit
  - Creates resources at startup
  - performs fast and atomic operations