

Course code: **CQ**

Course title: **Adobe CQ for Java programmers**

Days: 5

Description:

Course intended for:

The training is addressed to Java programmers (backend), who would like to get familiar with the application development skills in the Adobe Communiqué content management system.

Course objective:

The training objective is to acquire the basic skills necessary to develop Web portals on the basis of CQ. It introduces the general theory and aspects of the technology, necessary to commence programming. After the participants get familiar with the locally launched instances, the trainer will discuss the general issues associated with the fundamental concepts of CQ, which are OSGi and its implementation in the form of Apache Felix, Java Content Repository using the Content Repository Extreme as an example, ReST and the Apache Sling Web framework. Then the participants will go through the application development process, including the Web template with components to be embedded, integrated with the Web interface, which is to be made accessible later for the Web page editors. The basic terms, necessary to work as an author, such as parsys and dialog, are discussed. Some frequently encountered issues are also presented, such as the basics of securing access through the Closed User Group, path mapping and shortening, introduction to dispatchers. Main emphasis, however, is put on getting the participants familiar with the API provided and with the basic methods of access to objects and data in the rendered page requests.

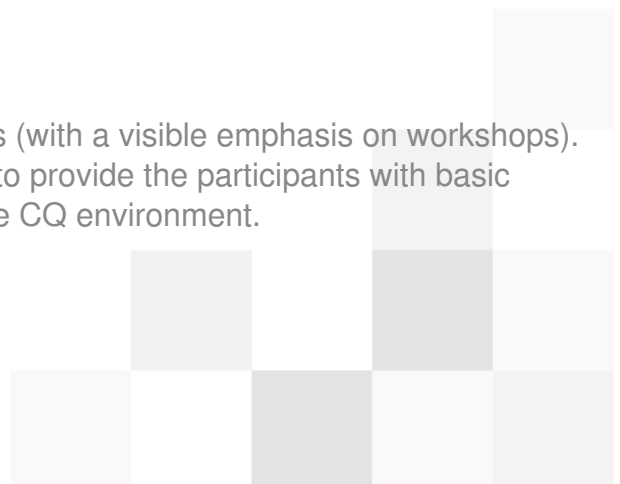
Requirements:

The training content assumes the practical skill of Java language programming, basic HTML and familiarity with the JSP technology.

Course parameters:

5*8 hours (net 5*7 hours) of lectures and workshops (with a visible emphasis on workshops). During the workshops, exercises will be conducted to provide the participants with basic programming skills and a thorough knowledge of the CQ environment.

Group size: no more than 8 participants.



Course curriculum:

1. Introduction to CQ
 - I. Launching – command line parameters
 - i. Author mode
 - ii. Publish mode
 - II. The system interface
 - III. Content management
 - i. Activation
 - ii. Pages
 - A. Structure
 - B. New page creation
 - a. Title vs. name
 - b. Template
 - C. Switching to edition
 - D. Operations
 - iii. Media (digital assets)
 - A. Upload
 - B. Renderings
 - C. Editing properties
 - D. Operations
 - iv. Tags
 - A. Operations
 - B. Properties
 - IV. Administration
 - i. Users
 - A. Operations
 - B. Groups
 - C. Properties
 - D. Access rights
 - ii. Tools
 - A. Structure
 - B. Designs
 - C. ClientLibs
 - D. Workflows
 - iii. Basic package management - Package Manager
 - A. General information
 - B. Upload
 - C. Installation
 - iv. Replication
 - A. General information
 - B. Agents
 - C. Tree activation
 - V. Tools associated with CRX repository
 - i. CRXDE Lite
 - A. Interface



- B. Tree structure
- C. Properties of nodes
- D. jcr:content
- E. Node edition and operations
- F. Node types
- G. Inheritance
- ii. Packages
 - A. Development
 - B. Properties
 - C. Filters
 - D. ZIP file structure
 - a. Conventions
 - b. jcr_root
 - c. .content.xml
 - d. The content of META-INF/vault
 - e. Package manual edition
 - 1. What needs particular care
 - 2. Structure in subfolders vs. structure in xml
- VI. General system architecture
- 2. Page edition
 - I. Interface
 - II. Content Finder
 - III. Override #cf
 - IV. Sidekick
 - V. Components
 - i. Grouping
 - ii. Parsys
 - iii. Adding
 - iv. Dialog
 - VI. Operations menu
 - VII. Page properties
 - VIII. Preview mode
 - IX. Design mode
 - X. Parameters in URL
 - i. ?debug=layout
 - ii. ?wcmmode=(edit|preview|design|disabled)
 - iii. ?debugClientLibs=true
 - XI. Quick preview of the node structure thanks to extensions
 - i. .html
 - ii. .xml
 - iii. .json
 - XII. Best practices
- 3. ReST
 - I. Definition
 - II. HTTP
 - III. Resource-oriented



- IV. Addressable resources
- V. Uniform, constrained
- VI. Representation oriented
- VII. Stateless
- 4. JCR + CRX
 - I. Hierarchy
 - II. Transactionability
 - III. Referential constraint
 - IV. Versioning
 - V. Workspace
 - VI. Name space
 - VII. Paths
 - VIII. Path transformation to URL
 - IX. Global ID
 - X. Full text search
 - XI. Nodes
 - i. Attributes
 - A. Value types
 - a. string
 - b. uri
 - c. boolean
 - d. date
 - e. long
 - f. double
 - g. decimal
 - h. path
 - i. name
 - j. binary
 - k. reference
 - l. weakreference
 - B. Distinguished attributes
 - a. jcr:primaryType
 - b. sling:resourceType
 - c. sling:resourceSuperType
 - d. jcr:mixinTypes
 - e. jcr:title
 - f. cq:template
 - g. jcr:uuid
 - h. cq:renderer
 - ii. Types (jcr:primaryType)
 - A. nt:unstructured
 - B. nt:file
 - C. nt:Folder
 - D. nt:OrderedFolder
 - E. dam:Asset
 - F. cq:Page



- G. cq:PageContent
- H. nt:resource
 - I. cq:Component
 - J. cq>EditConfig
 - K. cq:Dialog
 - L. cq:Widget
 - M. cq:WidgetCollection
 - N. cq:Tag
- iii. Inheritance
- XII. What it is like in practice
 - i. Node copying
 - ii. Issues requiring care
- 5. Apache Felix OSGi
 - I. Modularity
 - II. Review/system/console
 - i. Bundles
 - ii. Components
 - iii. Configuration
 - iv. Services
 - v. Sling Resource Resolver
 - III. Bundle vs jar
 - i. MANIFEST.MF
 - ii. Bundle-SymbolicName
 - iii. Import-Package
 - iv. Export-Package
 - v. Require-Bundle
 - vi. Creating bundles from jar files
 - IV. Bundle lifecycle
 - V. Selected API components
 - i. org.osgi.framework.BundleActivator
 - ii. org.apache.felix.scr.annotations.Service
 - iii. org.apache.felix.scr.annotations.Component
 - iv. org.apache.felix.scr.annotations.Properties
 - VI. Exemplary service
 - i. Implementation
 - ii. jar - > bundle
 - iii. Installation
 - iv. Use
- 6. Apache Sling
 - I. ReST + JCR + OSGi
 - i. Scripts – languages
 - II. Resources
 - III. URL decomposition
 - i. Content path
 - ii. Selectors
 - iii. Extension



- iv. Suffix
- IV. Script file location on the basis of URL
 - i. libs, apps
 - ii. GET, POST
 - iii. Priorities
- V. Mappings for Resource Resolution
 - i. sling:match
 - ii. sling:redirect
 - iii. sling:internalRedirect
 - iv. sling:alias
 - v. sling:status
- 7. Out of the Box
 - I. /libs
 - i. /libs/foundation/global.jsp
 - ii. components
 - A. parsys
 - B. sitemap
 - C. text
 - D. image
 - E. breadcrumb
 - II. Geometrix - a model application
 - i. application structure
 - ii. content structure
 - iii. components
 - iv. renderers
- 8. Development
 - I. Tools
 - i. CRXDE Lite based development
 - ii. FileVault
 - iii. Maven
 - iv. Eclipse
 - II. Java API overview
 - i. CQ taglib
 - A. include
 - B. includeClientLib
 - C. defineObjects
 - D. requestURL
 - ii. JSP objects
 - A. componentContext
 - B. component
 - C. currentDesign
 - D. currentPage
 - E. currentStyle
 - F. designer
 - G. editContext
 - H. pageManager



- I. pageProperties
- J. properties
- K. resourceDesign
- L. resourcePage
- M. resource
- N. slingRequest
- O. resourceResolver
- iii. Classes, interfaces
 - A. org.apache.sling.api.adapter.Adaptable
 - B. org.apache.sling.api.resource.Resource
 - C. org.apache.sling.api.resource.ResourceResolver
 - D. org.apache.sling.api.scripting.SlingScriptHelper
 - E. org.apache.sling.api.request.RequestPathInfo
 - F. com.day.cq.wcm.api.Page
 - G. com.day.cq.wcm.api.PageManager
 - H. com.day.cq.dam.api.Asset
- III. We develop
 - i. An application
 - ii. A template
 - iii. A renderer
 - iv. Page structure
 - v. Basic page content
 - vi. Several rendering scripts
 - vii. Modularization
 - viii. We initialize WCM
 - ix. Design
 - x. Navigation
 - xi. Title
 - xii. Logo
 - xiii. We use drag and drop
 - xiv. Bundle
 - xv. Dialog
 - A. panel
 - B. textfield
 - C. textarea
 - D. selection
 - E. options
 - F. tabpanel
 - G. multifield
 - H. richtext
 - I. image
 - J. dialogfieldset
- IV. Debugging
- V. Security
 - i. CUG
 - ii. Dispatcher



