

Kod szkolenia: **CQ**

Tytuł szkolenia: **Adobe CQ dla programisty Java**

Dni: 5

Opis:

Adresaci Szkolenia:

Szkolenie adresowane jest do programistów Java (backend), którzy chcą poznać podstawy umiejętności tworzenia aplikacji w systemie zarządzania treścią Adobe Communicé.

Cel szkolenia:

Celem szkolenia jest zdobycie od podstaw umiejętności niezbędnych do tworzenia portali w oparciu o CQ. Wprowadza podstawową teorię i aspekty technologii, niezbędne do rozpoczęcia samodzielnego programowania. Po wstępnym zapoznaniu uczestników z uruchomionymi lokalnie instancjami omawiane zostają ogólne zagadnienia związane z głównymi osiami fundamentalnymi CQ jakimi są OSGi i jego implementacja w postaci Apache Felix, Java Content Repository na przykładzie Content Repository Extreme, ReST oraz webowy framework Apache Sling. Następnie przeprowadzony zostaje proces stworzenia aplikacji, a w niej szablonu strony wraz z komponentami do osadzenia na niej, zintegrowanymi z webowym interfejsem dostępnym później dla edytorów stron. Pojawiają się podstawowe pojęcia dialogu i parsysa, niezbędne do pracy autora. Przedstawione zostają również pewne często używane zagadnienia, między innymi podstawy zabezpieczania dostępu poprzez Closed User Group, mapowanie i skracanie ścieżek, wstęp do dispatchera. Główny jednak nacisk kładziony jest na zapoznanie uczestników z dostarczonym API i podstawowych metodach dostępu do obiektów i danych w wywołaniach renderowanych stron.

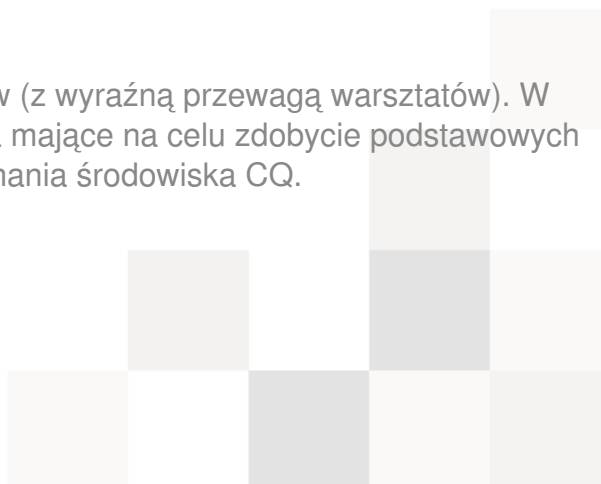
Wymagania:

W treści szkolenia zakłada się praktyczną umiejętność programowania w języku Java, podstawy HTML i znajomość technologii JSP.

Parametry szkolenia:

5*8 godzin (5*7 godzin netto) wykładów i warsztatów (z wyraźną przewagą warsztatów). W trakcie warsztatów przeprowadzane będą ćwiczenia mające na celu zdobycie podstawowych umiejętności programistycznych i gruntownego poznania środowiska CQ.

Wielkość grupy: max. 8 osób.



Program szkolenia:

1. Wprowadzenie do CQ
 - I. Uruchamianie - parametry linii poleceń
 - i. Author mode
 - ii. Publish mode
 - II. Interfejs systemu
 - III. Zarządzanie treścią
 - i. Aktywacja
 - ii. Strony
 - A. Struktura
 - B. Tworzenie nowej strony
 - a. Tytuł vs nazwa
 - b. Szablon
 - C. Przejście do edycji
 - D. Operacje
 - iii. Media (digital assets)
 - A. Upload
 - B. Rendycje (renderings)
 - C. Edycja właściwości
 - D. Operacje
 - iv. Tagi
 - A. Operacje
 - B. Właściwości
 - IV. Administracja
 - i. Użytkownicy
 - A. Operacje
 - B. Grupy
 - C. Właściwości
 - D. Uprawnienia dostępu
 - ii. Narzędzia
 - A. Struktura
 - B. Designs
 - C. ClientLibs
 - D. Workflows
 - iii. Podstawowe zarządzanie pakietami - Package Manager
 - A. Ogólne informacje
 - B. Upload
 - C. Instalacja
 - iv. Replikacja
 - A. Ogólne informacje
 - B. Agents
 - C. Aktywacja drzewa
 - V. Narzędzia związane z repozytorium CRX
 - i. CRXDE Lite
 - A. Interfejs



- B. Struktura drzewa
- C. Właściwości węzłów
- D. jcr:content
- E. Edycja i operacje na węzłach
- F. Typy węzłów
- G. Dziedziczenie
- ii. Pakiety
 - A. Tworzenie
 - B. Właściwości
 - C. Filtry
 - D. Struktura pliku ZIP
 - a. Konwencje
 - b. jcr_root
 - c. .content.xml
 - d. Zawartość META-INF\vault
 - e. Manualna edycja pakietu
 - 1. Na co trzeba uważać
 - 2. Struktura w podfolderach vs. struktura w xml
- VI. Architektura systemu ogólnie
- 2. Edycja strony
 - I. Interfejs
 - II. Content Finder
 - III. Pominięcie #cf
 - IV. Sidekick
 - V. Komponenty
 - i. Grupowanie
 - ii. Parsys
 - iii. Dodawanie
 - iv. Dialog
 - VI. Menu operacji
 - VII. Właściwości strony
 - VIII. Tryb podglądu
 - IX. Tryb projektowania
 - X. Parametry w URL
 - i. ?debug=layout
 - ii. ?wcmmode=(edit|preview|design|disabled)
 - iii. ?debugClientLibs=true
 - XI. Szybki podgląd struktury węzła dzięki rozszerzeniom
 - i. .html
 - ii. .xml
 - iii. .json
 - XII. Dobre praktyki
- 3. ReST
 - I. Definicja
 - II. HTTP
 - III. Resource-oriented



- IV. Addressable resources
- V. Uniform, constrained
- VI. Representation oriented
- VII. Stateless
- 4. JCR + CRX
 - I. Hierarchiczność
 - II. Transakcyjność
 - III. Spójność referencyjna
 - IV. Wersjonowanie
 - V. Workspace
 - VI. Przestrzenie nazw
 - VII. Ścieżki
 - VIII. Transformacja ścieżki do URL
 - IX. Globalne ID
 - X. Full text search
 - XI. Węzły
 - i. Atrybuty
 - A. Typy wartości
 - 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. Wyróżnione atrybuty
 - 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. Typy (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. Dziedziczenie
- XII. Jak to wygląda w praktyce
 - i. Kopiowanie węzłów
 - ii. Na co trzeba uważać
- 5. Apache Felix OSGi
 - I. Modularność
 - II. Przegląd /system/console
 - i. Bundles
 - ii. Komponenty
 - iii. Konfiguracja
 - iv. Serwisy
 - v. Sling Resource Resolver
 - III. Bundle vs jar
 - i. MANIFEST.MF
 - ii. Bundle-SymbolicName
 - iii. Import-Package
 - iv. Export-Package
 - v. Require-Bundle
 - vi. Tworzenie bundle z pliku jar
 - IV. Cykl życia bundle
 - V. Wybrane elementy API
 - 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. Przykładowy serwis
 - i. Implementacja
 - ii. jar -> bundle
 - iii. Instalacja
 - iv. Wykorzystanie
- 6. Apache Sling
 - I. ReST + JCR + OSGi
 - i. Skrypty - języki
 - II. Resources
 - III. Dekompozycja URL
 - i. Content path
 - ii. Selectors
 - iii. Extension



- iv. Suffix
- IV. Lokalizowanie skryptu na podstawie URL
 - i. libs, apps
 - ii. GET, POST
 - iii. Priorytety
- 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. komponenty
 - A. parsys
 - B. sitemap
 - C. text
 - D. image
 - E. breadcrumb
 - II. Geometrix - wzorcowa aplikacja
 - i. struktura aplikacji
 - ii. struktura content'u
 - iii. komponenty
 - iv. renderery
- 8. Development
 - I. Narzędzia
 - 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. Obiekty JSP
 - 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. Klasy, interfejsy
 - 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. Tworzymy
 - i. Aplikację
 - ii. Szablon
 - iii. Renderer
 - iv. Strukturę stron
 - v. Podstawową zawartość strony
 - vi. Kilka skryptów renderujących
 - vii. Modularyzację
 - viii. Inicjalizujemy WCM
 - ix. Design
 - x. Nawigację
 - xi. Tytuł
 - xii. Logo
 - xiii. Używamy 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. Debugowanie
- V. Zabezpieczenia
 - i. CUG
 - ii. Dispatcher



