OpenCms Days 2011

Workshop Track:

*The OpenCms 8 Content Subscription Engine*

Georg Westenberger,
Alkacon Software GmbH
Agenda

- Introduction
  - Basic concepts
  - Demonstration

- Using subscriptions
  - The <cms:usertracking> tag
  - Using collectors

- Using visits
  - Tracking user visits
  - Handling binary files

- Further topics
  - Configuration
  - Preventing browser caching
  - The Java API
Basic concepts (1)

- The content subscription engine provides two main features:
  - Subscriptions
  - Visit tracking

- How these features are used is not prescribed by the subscription engine
- They are tools used by the template developer to add subscription features to their sites
- Can be used independently from each other
- Subscription/visit data is stored in new tables in the database
• Subscriptions
  – Association between OpenCms users and individual VFS resources
  – Also possible for groups
  – Can be read or written using JSP tags and content collectors

• Use case: Web site personalization
  – Users can subscribe to their favorite contents
  – The user’s subscribed contents are displayed in a side bar

• Use case: Notifications
  – Administrator subscribe users to contents which they should read
Basic concepts (3)

- Visits
  - With the subscription engine, visits of a resource by users can be recorded
  - Only last visit of a user is tracked for each resource
  - Visit data can be accessed via JSP tag and collectors, too

- Use case: Displaying list of resources which have changed since the last visit

- Use case: Displaying different information depending on whether the user is visiting the page for the first time or not
Basic concepts (4)

• The subscription engine can be either used with ADE or with the classic OpenCms template mechanism
• Template/Formatter JSPs need to be adapted to use subscriptions or visits
• It’s up to the template developer to decide what they use subscriptions for
• Subscriptions and visits work on the level of resources, not pages or URLs
Demonstration

- Live Demo
Demonstration
New tag: <cms:usertracking>

- Multi-function tag which can perform different actions
- Used for both subscriptions and user tracking
- Possible operations for subscriptions:
  - Subscribe user to resource
  - Unsubscribe user from resource
  - Check if a resource is subscribed to a user
The `<cms:usertracking>` tag (2)

- Example for subscribing/unsubscribing to files:
  - `<c:if test="${not empty param.action}"">
    `<c:choose>`
    `<c:when test="${param.action == 'subscribe'}">
      `<cms:usertracking action="subscribe"
        file="${param.file}" /`>
    </c:when>`
    `<c:when test="${param.action == 'unsubscribe'}">
      `<cms:usertracking action="unsubscribe"
        file="${param.file}" /`>
    </c:when>`
  </c:choose>`
  `<c:if>`
Subscription for another user:

- `<cms:usertracking file="..." action="subscribe" user="${username}"/>

- Possible use case: Admin wants a specific user to see a certain content in his subscriptions
The `<cms:usertracking>` tag (4)

- **Checking for a subscription:**
  - `<c:set var="subscribed">`
    `<cms:usertracking action="checksubscribed"
      currentuser="true"
      file="${sitepath}"/>
    </c:set>`
  - The tag will be evaluated to ‘true’ or ‘false’

- **Checking for subscription of a specific user:**
  - `<c:set var="subscribed">`
    `<cms:usertracking action="checksubscribed"
      username="${username}"
      file="${sitepath}"/>
    </c:set>`
Using collectors (1)

• The `<cms:usertracking>` tag is used for operations on a single file

• Use the OpenCms content collector mechanism for accessing lists of contents

• New content collector: allSubscribed
  – Used to collect subscribed resources
  – Java class: `org.opencms.file.collectors.CmsSubscriptionCollector`

• Parameters are given as a `|`-separated list of key=value pairs
Using collectors (2)

- Configurable using various parameters, e.g.
  - resource: parent folder
  - includesubfolders: indicates whether subfolders should be searched
  - user: the user for which the subscribed resources should be fetched
  - currentuser: if true, gets subscribed resources for the current user
Example: Collecting all subscribed resources for the current user in the current site

```
<cms:resourceload collector="allSubscribed" param=
"resource=/|currentuser=true|includesubfolders=true|mode=all">
  <cms:resourceaccess var="item" />
  <div>${item.filename}</div>
</cms:resourceload>
```

Parameter “mode” has three possible values:
- all: find subscribed resources
- unvisited: finds subscribed resources unvisited since last change
- visited: finds subscribed resources visited since last change
Using collectors (4)

- `<cms:resourceload collector="allSubscribed" param="resource=/|currentuser=true|includesubfolders=true|mode=all">`<br>  `<cms:resourceaccess var="item" />`<br>  `<div>${item.filename}</div>`<br>  `</cms:resourceload>`

- New tags `<cms:resourceload>`, `<cms:resourceaccess>`
- Like `<cms:contentload>` and `<cms:contentaccess>`, but only load resource information, not contents
- Needed because resources which aren’t normal XML contents can be subscribed
- Faster, too, if you only need e.g. the title property or resource name – does not need to parse XML files
- `${item}` is a Java bean of type
  org.opencms.jsp.util.CmsJspResourceAccessBean
User tracking functionality:
- Marking a page as visited by a given user
- Checking whether a user has visited a page
- Finding visited resources

Done via the `<cms:usertracking>` tag and content collectors
- Special handling for binary files needed
Tracking user visits (2)

- **Example:** Mark a resource as visited by the current user

  `<cms:usertracking action="visit" file="${filepath}"/>

- **Example:** Mark visit by a specific user

  `<cms:usertracking action="visit" user="User" file="${filepath}" />

- **Example:** Check if a user has visited a resource

  `<cms:usertracking action="checkvisited" currentuser="true" file="${filepath}" />

  - Will evaluate to ‘true‘ or ‘false‘
Tracking user visits (3)

• New collector: allVisited
  – Java class:
    org.opencms.file.collectors.CmsSubscriptionCollector

• Collects visited resources, filtered by user, folder and time range

• Visited resources for current user:
  – currentuser=true

• Visited resources for user Username:
  – user=Username
Selecting by time range:
- Parameters: daysfrom, daysto
- Values: Range of days back from the current time for which the visited resources should be returned
- Example: Resources visited by the current user in the last two days
  - \texttt{daysfrom=2|daysto=0|currentuser=true}
• Selecting by folder:
  – resource=/folder/

• Selecting by folder or subfolders:
  – resource=/folder/|includesubfolders=true
Handling binary files (1)

- This works only for structured contents
- Special resource init handler for binary files
- Intercepts direct requests to resources and marks them as visited
- Must be configured in opencms-system.xml:

```
<resourceinit>

    
    ......

    <resourceinithandler class="org.opencms.db.CmsUserTrackingResourceHandler"/>

</resourceinit>
```
Handling binary files (2)

- Will mark certain files as visited by a user if their URL is requested by them

- You also have to set the export property to false so that the files will not be statically exported

- Controlled by the property `usertracking.mark`

- Values:
  - online: opened resources will only be marked as visited in the Online project
  - true: opened resources will always be marked
  - false (or not set): resources will not be marked
Handling binary files (3)

- Why not do this for all files?
  - Rendered pages consist of multiple resources
  - Especially now with Advanced Direct Edit (container pages containing multiple elements)
  - The resource which is directly requested is not necessarily the resource which should be marked as visited
  - The `<cms:usertracking>` tag gives the template developer more control than the resource init handler
Configuration

• Add entry to opencms-system.xml:
  – `<subscriptionmanager enabled="true" poolname="default" maxvisited="500" />
  – Add it as the last sub-element of the <system> element

• enabled: enables or disables the subscription engine

• maxvisited: the maximum number of visited resources stored for a given resource

• poolname: the database pool used to access the subscription information
  – Important when using a cluster, because data should be stored in a central location
Preventing browser caching

- Subscriptions/visits will not work correctly if the user’s browser caches pages on which the `<cms:usertracking>` tag or collectors are used.

- Insert `<cms:nocache>` JSP tag into main template JSP before any output is written:

  `<%@page buffer="none" session="false" taglibs="c,cms,fn" %><cms:nocache/>`

- This will tell the browser to not cache pages.
• Everything you can do with the collectors or tags, you can do with the Java API
• org.opencms.db.CmsSubscriptionManager
• Access via org.opencms.main.OpenCms
• Example: reading all subscribed resources for a user

```java
CmsObject cms = ...;
CmsUser user = ...;
List<CmsResource> resources = OpenCms.getSubscriptionManager().readAllSubscribedResources(cms, user);
```
The Java API (2)

- Also offers some additional functionality

- Example: reading the last visit date

```java
CmsObject cms = ...;
CmsUser user = ...;
CmsResource resource = ...;
long lastVisited = OpenCms.getSubscriptionManager()
          .getDateLastVisitedBy(cms, user, resource);
```
Questions

QUESTIONS?

- Any Questions?

¿Preguntas?

Questiones?

Fragen?
Thank you very much for your attention

Georg Westenberger
Alkacon Software

http://www.alkacon.com
http://www.opencms.org