1. Begriffe

- **CSS** (Cascading Style Sheets) ist eine deklarative StyleSheet-Sprache, die die Darstellung eines besonders ausgezeichneten Inhalts bestimmt. Eingesetzt mit HTML oder XML legt CSS fest, wie die Abschnitte, die gleich aussehen sollen, dargestellt werden. Im entsprechenden HTML-oder XML-Dokument wird nur die Bedeutung der Abschnitte beschrieben. Dazu ist es wichtig, dass die gleich ausgezeichneten Abschnitte als eine Klasse erkannt werden. Man trennt auf diese Weise den Inhalt von dem optischen Design.

- **CMS** (Content Management System) ist ein Anwendungssystem, das die Organisation und Bearbeitung von Inhalten wie Text- und Multimedia-Dokumente ermöglicht. Die zu erstellenden Informationen, meist für das Web, werden als Content (Inhalt) bezeichnet. Zum Bedienen des Systems sind keine Programmierkentnisse erforderlich.

- **Extension** steht für Erweiterung. Ohne den vorhandenen Quellcode zu verändern, können Erweiterungen bzw. zusätzliche Funktionalitäten der Rahmenapplikation OLAT hinzugefügt werden. Dafür werden die sogennanten Extension Points, Erweiterungspunkte, verwendet, die als Schnittstellen zu Verfügung stehen.

- **HTML** (Hypertext Markup Language) ist eine Hypertext-Auszeichnungssprache zur Beschreibung und Strukturierung von Inhalten in Dokumenten, die die Basis für das World Wide Web sind. HTML ist keine Programmiersprache, man spricht hier nicht von "Programmieren" sondern von "Notieren".

- **JSP** (Java Server Pages) bezeichnet die von Sun Microsystems entwickelte Technologie zur dynamischen Generierung von Dokumenten. Der deklarative Stil von JSP ermöglicht dabei eine bessere Trennung von Design und Anwendungslogik. Grundsätzlich lassen sich JSP als HTML-oder XML-Seite mit zusätzlichen JSP-Tags und Java-Code beschreiben. Mit der Programmiersprache Java kann ausschließlich die Logik implementiert werden und das Design kann dagegen weiterhin in HTML durchgeführt werden.

Als Webanwendung basiert OLAT auf Servlets und JSPs, deren Begriffserklärungen deswegen notwendig sind.

- **OLAT** (Online Learning and Training) ist ein von der Züricher Universität entwickelte Open Source Learning Management System mit einer sehr modularen, erweiterbaren und anpassbaren Architektur. Seit Version 3.0 wird das System von komponentenbasierter Java-Anwendung unterstützt. Ziele sind, Vorlesungen und Übungen über das Internet anbieten zu können, wobei der Aufbau und die Durchführung von Lehrveranstaltungen im Vordergrund stehen.
- **SCORM** (Sharable Content Object Reference Model) repräsentiert eine Anzahl von Standards und Spezifikationen aus diversen Quellen und ermöglicht die Austauschbarkeit und die Wiederverwendbarkeit in verschiedenen Umgebungen von E-Learning-Inhalten. Mit diesem Referenzmodell können Lernressourcen in OLAT erzeugt werden.
- **Servlets** bezeichnen Java-Klassen, die als Servererweiterungen die Funktionalität des Servers in sofern ergänzen, dass sie auf ganz bestimmte Anfragen eines Clients mit beliebig dynamisch erzeugten Inhalt antworten können. Sie sind gut zum Erzeugen von Binärdaten geeignet wie z.B. von Bildern, Archiven usw. Alle Servlets müssen die Java-Klasse javax.servlet.Servlet oder eine davon abgeleitete Klasse implementieren, damit sie innerhalb eines J2EE (Java-2-Plattform Enterprise Edition) Applicationserver laufen können.
- **Tomcat** bezeichnet eine Laufzeitumgebung, ein in Java geschriebener Open Source Servlet-Container. Tomcat wurde im Rahmen des Jakarta-Projekts von Apache entwickelt und wurde speziell für die JavaServer Pages geplant. Neben einem kompletten HTTP-Server stellt Tomcat für die Kommunikation mit anderen Servern den Coyote-Connector zu Verfügung. Mit Hilfe des JSP-Compilers Jasper können auch JavaServer Pages in Servlets übersetzt und ausgeführt werden.
- **XML** (Extensible Markup Language) bezeichnet eine Auszeichnungssprache zur Darstellung hierarchisch strukturierter Daten in Textdatei-Form und wird beim Austausch von Daten zwischen unterschiedlichen IT-Systemen eingesetzt. Im OLAT-Projekt wird XML auch für die Konfiguration und die Anpassung der Komponenten sowie für die Kommunikation angewendet.

2. Konzepte

2.1 MVC: MVC ist ein Architekturmuster, das Softwaresysteme in drei Einheiten gliedert: Datenmodell (Model), Präsentation(View) und Programmiersteuerung (Controller). Die strikte Trennung zwischen Businesslogik, Ablauflogik, Darstellung und Datenhaltung erleubt eine schnelle und übersichtliche Entwicklung, die Wiederverwendbarkeit der Komponenten und eine leichte Erweiterbarkeit.

Model: Das Modell enthält die Businesslogik und beschreibt die Kernfunktionalitäten und Anwendungsdaten, unabhängig von Präsentations- und Ablauflogik.

View: Die Präsentation, ist für die Darstellung der Daten aus dem Modell verantwortlich, realisiert bzw. bildet die Benutzerinteraktionen auf dem Bildschirm ab. Die View muss das Modell überwachen, um sich nach Änderungen von Daten im Modell zu aktualisieren.

Controller: Der Controller steuert den Ablauf der Programme. Der Controller verwaltet die Views und bearbeitet die von View gelieferten Benutzeraktionen, wertet diese aus und verändert die entsprechenden Daten im Modell.

2.2 JEE: JEE(Java Platform, Enterprise Edition) spezifiziert eine Softwarearchitektur, die zur transaktionsbasierten Ausführung von Java-basierten Anwendungen dient. Sie definiert Softwarekomponenten und Dienste, auf deren Basis aus modularen Komponenten verteilte, mehrschichtige Anwendungen entwickelt verden können.

2.3 GUI-Framework: Das Olat-Web-GUI-Framework ist ein vollständig in Java geschriebenes, Komponenten-basiertes, MVC-orientiertes Framework, das eine effizientere Entwicklung und Pflege von Webapplikationen ermöglicht. Olat kann sowohl mit als auch ohne AJAX-Unterstützung betrieben werden.

2.4 Hibernate: Hibernate ist ein Open-Source-Persistenz-Framework für Java. Es ermöglicht die Speicherung von Objekt-Daten in einer relationalen Datenbank und aus entsprechenden Datensätzen wiederum Objekte zu erzeugen. Dadurch kann auf die Programmierung von SQL-Abfragen verzichtet werden und die Anwendung ist unabhängig vom SQL-Dialekt der verwendeten Datenbank.

2.5 Spring: Spring ist ein Open-Source-Framework für die Java-Plattform, das die Entwicklung von Java/JEE-Anwendungen vereinfacht. Das Framework basiert auf folgenden Prinzipien:

Dependency Injection: Den Objekten werden die abhängigen Objekte bzw. Ressourcen zugewiesen. Sie müssen sie nicht selbst suchen.

Aspektorientierte Programmierung: Unter aspektorientierter Programmierung versteht man ein Programmierparadigma, das anstrebt, verschiedene logische Aspekte einer Anwendung getrennt voneinander zu entwerfen, entwickeln und zu testen.

Templates: Templates dienen dazu, die Arbeit mit einigen APIs zu vereinfachen, indem Ressourcen automatisch aufgeräumt werden und Fehlersituationen einheitlich behandelt werden.

2.6 Servlet-Architektur: Als Servlets bezeichnet man Java-Klassen, deren Instanzen innerhalb eines JEE-Applikationsservers Anfragen von Clients entgegen nehmen und beantworten. Der Inhalt der Antworten kann dynamisch, also zur Zeit der Anfrage, erstellt werden und muss nicht bereits statisch für den Webserver verfügbar sein.

3. Applikationsbeschreibung

OLAT (Online Learning and Training) ist ein webbasiertes Learning Management System (LMS), das seit 1999 an der Züricher Universität entwickelt wird. Mit Beachtung der Apache 2.0 Open Source Lizenz ist die Software gebührenfrei erweiterbar und verwendbar.
Das LMS OLAT baut auf modernen Technologien auf und ist zu 100% in der Programmiersprache Java geschrieben. Der Betrieb unter diversen Betriebssystemen wie Linux, Windows, MacOSX, BSD, Unix oder Solaris benötigt keine zusätzlichen Anpassungen. Zur Persistierung der Daten können ebenfalls verschiedene Datenbank-Managementsysteme wie MySQL, Postgres oder Oracle eingesetzt werden.
OLAT beruht auf der Java 2 Enterprise Edition (J2EE) und verwendet eine Servlet-basierte Architektur. Mit dem MVC Framework wird eine moderne, fehlerreduzierte und schnelle Entwicklung erlaubt, wobei eine strikte Trennung zwischen Darstellungs-, Ablauf-, Businesslogik und Datenhaltung besteht.

Mit zunehmenden Anforderungen an das Campusmanagement wächst OLAT zu einer modularen, leicht anpassbaren und modifizierbaren Architektur. Unterstützt werden dabei die gängigen E-Learning-Standards wie IMS Content Packaging, IMS QTI oder SCORM.

Funktionen

Zu den wichtigsten Funktionen von OLAT zählen die Organisation eines flexiblen Kurssystems, die allgemeine Verwaltung von Lernressourcen inklusive Katalogisierung und die Bereitstellung von Editorenwerkzeugen für Tests, Fragebögen oder Kurse.

Weitere Funktionalitäten und Eigenschaften von OLAT, die einem unternehmensinternen (privaten), einem kooperativen (kollaborativen) und einem öffentlichen Bereich zu entnehmen sind, werden im Folgenden aufgezählt:

- Privater (Nutzer-)Bereich
  - Profil, Einstellungen und Passwort anpassen
  - Portal planen und organisieren
  - Kalender verwalten und Notizen führen
  - Kurse bookmarken, Persönliche Ordner
  - Persönliche Leistungen im besuchten Kurs anschauen
  - Kurse starten, Einschreibungen in Veranstaltungen
- Kollaborativer Bereich und kooperative Aktivitäten
  - Gruppenkalender
  - Mitgliederlisten anschauen
  - Benachrichtigungsservice via E-Mail oder RSS, Chat, Wiki, Forum für Gruppenkommunikation
  - Dateien in Ordnern austauschen
- öffentlicher Bereich
  - Administration
  - Durch Gruppenmanagement Kurse verwalten, Kurskonfiguration Kurse einstellen
  - Durch Rechtemanagement den Zugang zu Werkzeugen verteilen

- Datenarchivierung
- Durch Bewertungswerkzeug Punkte von Kursteilnehmern anschauen und editieren

Rechte- und Rollenkonzept

OLAT ist rechtebasierend und hat einen hierarchischen Aufbau, der sich an das Policy-Konzept von Java anlehnt. Es gibt Identities (User), Gruppen, die Identities haben (sog. Securitygroups, nicht zu verwechseln mit Businessgroups), Rechte, Ressourcen und Policies.

User bilden eine oder mehrere Gruppen und haben Rechte (z.B. „lesen“) auf eine gewisse Ressource. Die entsprechende Gruppe wird von einer Policy (Kombination von Gruppe, Recht und Ressource) referenziert und somit hat die Policy einen Fremdschlüssel auf die Gruppe, einen auf die Ressource und einen Text für das Recht.

BusinessGroup: ist ein Konzept für eine Gruppe, die etwas gemeinsames macht und kollaborativ aktiv ist. BusinessGroup hat einen Namen und Beschreibung und implementiert eine Anzahl von Funktionen, die nützlich für eine Zusammenarbeit sein können.

SecurityGroup: Mit der Erzeugung von einer Lernressource, z.B. einem Kurs, wird automatisch eine SecurityGroup für diese Ressource erzeugt, der Erzeuger wird in die Gruppe hinzugefügt. Jeder in der Gruppe kann dann die Ressource verwalten.

Erweiterbarkeit

OLAT hat einen Mechanismus zur Erweiterung, ähnlich wie von Eclipse -Plugin. Damit kann man ohne Zugriff auf den Quellcode OLAT modifizieren. Die sog. „Extension Points“ ermöglichen es, OLAT nach kundenspezifischen Bedürfnissen zu erweitern, ohne dabei änderungen am Grundsystem vornehmen zu müssen. Somit wird die Installation von Updates erleichtert. Beispiele für solche Erweiterungen wären ein zusätzlicher Navigationspunkt in der Hauptnavigation oder ein neuer Kursbaustein.
Um OLAT zu erweitern, wird ein Pfad in die Datei olat_extensions.xml hinzugefügt und die eigene Datei extension.jar in WEB-INF/lib Directory. Das Paket ch.goodsolutions.demoextension zeigt, wie man SitesCreator benutzen kann.
Folgende Extension Points werden definiert:

Extension interface	Extension point	Klasse benutzt die Erweiterung	Beschreibung
org.olat.extensions.globalmapper MapperProvider	org.olat.dispatcher.DispatcherAction	org.olat.dispatcher.DispatcherAction	Extension kann aus einem Mapper bestehen und sie kennt den Pfad, der mit dem Mapper assoziiert.
org.olat.extensions.action.ActionExtension	org.olat.home.HomeMainController	…	Extension kann Link mit dem Text und der einsprechenden Beschreibung besorgen. Sie definieren die Aktion, was passiert, wenn der Link geklickt wird.
org.olat.extensions.css.CSSIncluder	org.olat.gui.components.Window	org.olat.gui.css.CSSGenerator	Extension kennt den Pfad, wo sich die Stylesheets in OLAT befinden. Alle CSS - Definitionen aus allen Erweiterungen teilen einen Namensraum. Es ist nützlich, kurze Namen von Extension vor die CSS- Klasse zu setzen. Die Definition der CSS Klasse soll nicht mit „o_“ beginnen, da es für OLAT Basissystem benutzt wird.
org.olat.extensions.hibernate.HibernateConfigurator	org.olat.persistence.DB	org.olat.persistence.DB	Extension kann zusätzlich Hibernate-Darstellung hinzu fügen. Die Erzeugung einer Tabelle hier ist sehr schwer. Diese braucht ein separates SQL-Script in der Datenbasis Dialekt, den Sie benutzen.
org.olat.extensions.sitescreator.SitesCreator	org.olat.gui.control.generic.dtabs.DTabs	org.olat.FullChiefController	Wenn Extension Sites wie „Home“, „Group“, „Learning Resources“ hinzufügen will, dann soll SitesCreator geöffnet werden. Eine Liste von Site-definition-Objekten muss zurückgegeben werden und jedes von ihnen kann eine Site-Instanz erzeugen.

OLAT-Verzeichnisse

Beschreibung der OLAT-Verzeichnisse:

```
/INSTALL_DIR
|
+-/olatdata : OLAT Laufzeitdaten
| |
| +-/bcroot
| | |
| | +-/course : Kurse
| | +-/cts : Gruppenordner
| | +-/forum : Dateinanhänge zu Forumseinträgen
| | +-/homepages : Benutzer Homepages
| | +-/homes : Benutzerordner
| | +-/repository : Lernressourcen
| | +-/scorm : SCORM Laufzeitdaten
| | +-/search_index : Der Index der Volltextsuche
| | +-/tmp : Temporäre Daten
| |
| +-/calendars : Kalenderdateien
| +-/logs : Log Dateien
| +-/monitoring : Statistikdaten der Systemerwachung
| +-/recording : Daten des Recording/Test Frameworks
| +-/system : Installierte Upgrades
| +-/tmp : Temporäre Daten
+-/webapp : Ordner für den Betrieb von OLAT (Applikationsdateien)
| |
| +-/examples : Demo Kurs
| +-/help : Hilfe und Support
| +-/static : statische Elemente des Portals wie Cascading Style Sheets
oder Bilder
| +-/WEB-INF : Hauptordner einer Servlet-Anwendung(enthält Konfigurations-
dateien von OLAT, z.B. olat_extensions.xml)-erforderlich für
Entwicklungsarbeiten
| | |
| | +-/classes : compilierte Quelldateien
| | +-/lib : zusätzliche Java-Bibliotheken
| | +-/patchesSrc : enthält Pfade zur Bibliotheken
| | +-/src : Quelldateien
```

4. Quellen

http://www.olat.org/docu/dev/olat_dev_docu_one_page.html#d0e1983
http://www.olat.org/website/en/download/olat_install_admin_docu_one_page.htmlhttp://www.olat.org/website/en/html/unit_development.html
http://www.olat.org/website/en/html/download_documentation.html
http://www.frentix.com/de/resources/docu/OLAT-Systemdokumentation.pdf
http://pcai042.informatik.uni-leipzig.de/swp/Material/KP-Allgemein.pdf