Aufgabe 2 17.04.2011

Softwaretechnikpraktikum SS 2011

swp11-2

Projektleiter : Matthias Haeßner | Dokument erstellt von David Müller

Recherchebericht

1. Begriffe

1.1. Spring

Spring ist ein eigens für die Java-Plattform entwickeltes, quelloffenes Framework, dessen Ziel die vereinfachte Entwicklung von Java/Java EE Applikationen und die Förderung guter Programmierpraktiken ist. Hierfür wird ein konsistentes Programmier- und Konfigurationsmodell bereit gestellt, welches die Geschäftslogiken und Objekte der Anwendung vor deren komplexer Verwaltung isoliert. Bei all dieser Hilfe für den Entwickler, sind die im Framework entstehenden Programme nicht abhängig von den Spring-APIs.

1.2. JSP

Java Server Pages (JSP) ermöglichen die einfache Erzeugung von dynamischen HTML- und XML-Antworten eines Webservers. Im Gegensatz zu den Servlets ist der Java-Code jedoch direkt in die HTML-Seite eingebettet, wo durch Logik und Visualisierung getrennt werden. Beim Einsatz im Webserver werden JSPs zuvor in Servlets übersetzt.

1.3. Tomcat

Tomcat ist eine quelloffene Implementierung der Java Servlet und JavaServer Page Technologien, welche bei Apache innerhalb des Jakarta Projekts entwickelt wird. Genauer handelt es sich hierbei um einen Servlet-Container, der die Ausführung von Java-Servlets innerhalb eines Webservers ermöglicht. Mit Hilfe des JSP Kompilierers Jasper können Java Server Pages in Servlets übersetzt und ausgeführt werden. Hinzu kommt noch ein HTTP Server, der jedoch hautsächlich zur Entwicklung eingesetzt wird.

1.4. Java EE

Die Java Platform Enterprise Edition (Java EE) ist die Spezifikation einer Softwarearchitektur für transaktionsbasierte Ausführung für speziell in Java geschriebene Web-Anwendungen. Die Spezifikation stellt einen Rahmen zur Verfügung auf dessen Basis aus modularen Komponenten mehrschichtige und verteile Anwendung entwickelt werden können, wobei klar definierte Schnittstellen zwischen den Komponenten und Containern dafür sorgen das eine Interoperabilität

zwischen Komponenten verschiedener Hersteller gewährleistet wird. Als Laufzeitumgebung benötigen Java EE Komponenten einen Java EE Application Server.

1.5. Extension

OLAT bietet einen vorteilhaften Erweiterungsmechanismus, der es den Entwicklern erlaubt, OLAT zu verbessern ohne das der ursprüngliche Quellcode bearbeitet werden muss. Das Konzept funktioniert folgendermaßen: Um OLAT zu erweitern, muss man lediglich eine Zeile in die Datei „olat_extentions-.xml“ einfügen, die auf den neu geschriebenen Quellcode zeigt. Abschließend wird die jar- datei mit den neuen Programmteil in das Verzeichnis „Web-inf/lib“ abgelegt und die Erweiterung ist abgeschlossen.

1.6. OLAT als LMS

OLAT (Online Learning and Training) ist ein Open Source Learning Management System (LMS), welches seit 1999 von der Universität Zürich entwickelt wird. Es stellt verschiedene Formen von webbasiertem Lehren und Lernen als Webapplikation zur Verfügung und umfasst viele für E-Learning-Plattformen typische Bausteine wie z.B. Content Management, Wikis, Test und Selbsttest uvm.

1.7. Maven

Maven ist ein Softwaretool welches beim Build- und Projectmanagement zum Einsatz kommt. Dabei wird es hauptsächlich für die Java-Programmierung eingesetzt, kann aber auch für andere Sprachen wie z.B. C#, Ruby oder Scala benutzt werden.

1.8. Brasato (OLAT-Core)

OLAT-Core folgt zwei Entwurfsmustern – dem MVC-Pattern und dem Composite-Pattern. In OLAT-Core sind die Schichten Model und View miteinander verwoben in der abstrakten Klasse org.olat.core.-gui.components.Component. Über deren Subklassen und Dateien kann dann wieder eine Trennung erreicht werden.

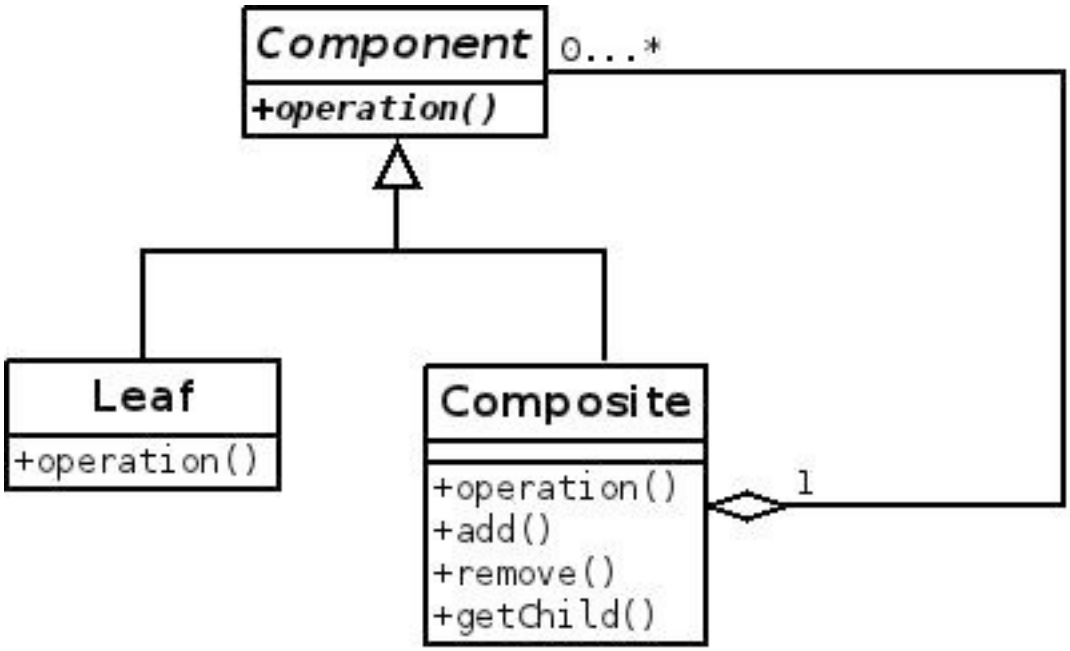
Aufgabe 2 17.04.2011

Softwaretechnikpraktikum SS 2011

swp11-2

Projektleiter : Matthias Haeßner | Dokument erstellt von David Müller

Das Composite-Pattern:



Component ist eine abstrakte Klasse und definiert die Schnittstelle für die Komponenten. In org.olat.-core.gui.components.Component sind dies u.a.

+ addListener(controller : Controller) - hinzufügen eines Controllers

+ getHTMLRendererSingleton() - Rückgabe eines org.olat.core.gui.components.ComponentRenderer zur Darstellung der Komposita

+ dispatchRequest(request : UserRequest) – Behandlung der Nutzeranfrage; wird an die Komposita delegiert.

Der Controller org.olat.core.gui.control.Controller definiert eine einheitliche Schnittstelle. Implementierung folgt dem Strategy-Pattern.

1.9. Mercurial

Mercurial ist ein in Python entwickeltes plattformunabhängiges, verteiltes Versionskontrollsystem welches in der Softwareentwicklung zum Einsatz kommt. Mercurial wird hauptsächlich über die Kommandozeilen bedient, allerdings stehen mit TortoiseHG für Windows und MacHG für Mac OS X auch grafische Oberflächen zur Verfügung. Entwicklungsumgebung wie Netbeans oder Eclipse bieten außerdem die Möglichkeit mit einem Plugin zu integrieren.

Projektleiter : Matthias Haeßner | Dokument erstellt von David Müller

1.10. aspektorientierte Programmierung

Aspektorientierte Programmierung (AOP) ist ein Programmierparadigma der Objektorientierten Programmierung, dessen Motivation es ist logisch unabhängige Belange sogenannter Cross-Cutting-Concerns (miteinander verwobene Anforderungen) auch physisch zu trennen. Das Ziel ist es auch für schwierig trennbare Prozesse wie z.B. Logging sauberen und wiederverwendbaren Code zu schreiben

2. Konzepte

2.1. Open Source

Open Source Software, wie z.B. OLAT, zeichnet sich dadurch aus dass der Quelltext offen liegt und frei verfügbar ist. OLAT darf damit kostenlos benutzt, kopiert und weiterverbreitet werden und kann zudem leicht weiterentwickelt oder verändert werden und die entstehende veränderte Version benutzt oder wiederveröffentlich werden.

Allgemein entfallen bei Open Source Software die sonst in der Softwarebranche üblichen Lizenzkosten. Die Software wird unter freien Lizenzen wie z.B. der general public license (GPL) veröffentlich

2.2. OLAT Schichtenmodell

OLAT ist mit drei Schichten(Tiers) aufgebaut, dem User Tier, dem Business Tier und dem Data Tier.

Das User Tier enthält die sichtbare Applikation und läuft auf dem Client im Webbrowser. Der User greift hier auf die Funktionen zu die auf der Benutzeroberfläche(GUI) als Seiten dargestellt sind.

Im Business Tier wird die allgemeine Ablaufsteuerung geregelt. Die Funktionen der Validierung (validiert Benutzereingaben), Businesslogik(steuert die Ablauflogik), Persistierung (gibt Daten an den Data Tier weiter) und die Aufarbeitung und Bereitstellung der HTML-Antwortseite an das User Tier werden durch weitere Schichtungen getrennt.

Das Data Tier enthält die Datenbank und das Filesystem auf die der Zugriff mittels Hibernate (Datenbank) und dem Virtual Filesystem VFS (Filesystem) ermöglicht wird.

Business und Data Tier laufen hierbei auf einem oder mehreren Servern.

swp11-2

Projektleiter : Matthias Haeßner | Dokument erstellt von David Müller

2.3. Hibernate

Die Hauptaufgabe von Hibernate ist das Object-Relational-Mapping(ORM) womit es möglich ist gewöhnlich Objekte mit Attributen und Methoden in einer relationalen Datenbank zu speichern und umgekehrt aus entsprechenden Daten einer solchen Datenbank wiederrum Objekte zu erzeugen. Die Speicherung der Beziehungen zwischen Objekten erfolgt mittels Abbildung auf die entsprechenden Datenbankrelationen.

Als Anfragesprache nutzt Hibernate dazu SQL Statements, die SQL ähnliche Abfragesprache Hibernate Query Language (HQL) oder eine objektorientierte Variante mittels der Hibernate Criteria-API. Die Abfrage werden mit einem JDBC-Treiber übersetzt, was Hibernate unabhängig von der verwendeten Datenbank macht solange selbige einen solchen JDBC Treiber besitzt.

2.4 Modularer Aufbau

Der modulare Aufbau einer Software bringt vor allem den Vorteil der Wiederverwendbarkeit mit sich. Zusätzlich erleichtert ein solcher Aufbau die Entwicklung da sich die Module als Ganzes austauschen lassen und somit eine bessere Arbeitsaufteilung auf Gruppen ermöglicht. Ebenso lassen sich die Module sehr einfach testen.

2.5. Model-View-Controller (MVC)

MVC ist ein Architekturkonzept in der Softwareentwicklung welches die Anzeige(View), Eingabe (Controller) und Verarbeitung(Model) trennt.

Hierbei dient das Model der Datenverarbeitung und Datenmanipulation und enthält zudem alle internen Daten. Der Controller ist für die Interaktion mit dem Benutzer zuständig, steuert zudem welche Daten im Model berechnet werden müssen und verwaltet die Darstellung im View. Der View dient der Darstellung der vom Model berechneten Daten.

2.6. Client-Server-Modell

Das Client-Server-Modell beschreibt eine Programmarchitektur welche ermöglicht Aufgaben innerhalb eines Netzwerkes zu verteilen. Hierbei wird ein Programm in zwei Teile gespalten, den Client und den Server. Der Client beinhaltet meisten nur die Benutzeroberfläche, der Server die Geschäftslogik.

swp11-2

Projektleiter : Matthias Haeßner | Dokument erstellt von David Müller

2.7. Servlets

Der Begriff Servlet setzt sich aus den beiden Begriffen Server und Applet zusammen und beschreibt also ein serverseitiges Applet (genauer gesagt eine Java-Klasse, deren Instanz auf einem Java-Webserver Clientanfragen entgegennimmt und beantwortet). Servlets werden dabei im Gegensatz zu Applets, die auf dem Rechner des Endbenutzers ausgeführt werden, auf dem Server ausgeführt. Instanzen der Servlets werden dabei bei Bedarf von einem Web-Container (z.B. Tomcat) erzeugt und von ihm aus angesprochen, wobei der Web-Container seinerseits mit dem Server kommuniziert oder selber integraler Bestandteil eines Java-Webservers ist. Der Inhalt der Antwort eines Servlets muss dabei nicht statisch bereits auf dem Webserver bereit liegen sondern kann dynamisch zur Anfragezeit neu erstellt werden.

Der Aufruf eines Servlets funktioniert nun so dass der Endbenutzer eine URL anfordert, wobei der Webserver so konfiguriert ist das er Aufrufe an ein bestimmtes Unterverzeichnis in Aufrufe an die Servlet-Engine umwandelt. Nach Weiterleitung intrepretiert das Servlet den Schlussteil der URL als Aufrufparameter und aktiviert einen damit verbundenen Dienst.

2.8 Inversion of Control und Depency Injection

Unter Inversion of Control (IoC, deutsch "Umkehrung der Steuerung") versteht man ein Umsetzungsparadigma, welches Anwendung im Bereich der objektorientierten Programmierung findet. Das Paradigma beschreibt die Arbeitsweise eines Frameworks (in unserem Fall Spring) folgendermaßen: Man registriert eine Funktion der Anwendung bei der Standardbiliothek, damit diese die Funktion zu einem späterem Zeitpunkt aufrufen kann. Anstatt der Anwendung die Steuerung des Kontrollflusses und den Aufruf der Standardfunktionen zu überlassen, wird die Aufgabe der Steuerung einiger Unterprogramme an das Framework abgegeben. Ein simples Beispiel für die Umsetzung, neben den bereits beschriebenen Servlets, sind Listener, wie man sie beim Observer-Muster vorfindet.

Anwendung findet IoC beim Dependency Injection (DI) Entwurfsmuster. Dieses dient in objektorientierten Systemen dazu, Abhängigkeiten zwischen Objekten zu senken. DI bezieht sich hierbei ausschließlich auf die Erzeugung von Objekten und kann somit als Verallgemeinerung der Fabrikmethode ausgelegt werden.

Im Gegensatz zu klassischen OO-Anwendungen, bei denen jedes Objekt selbständig für seine Erzeugung und die Erzeugung und Verwaltung seiner Ressourcen und Objekte zuständig ist, überträgt DI diese Verantwortung an ein externes Framework. Dadurch wird das Objekt nicht mehr dazu genötigt, Kenntnisse seiner Umgebung mitzubringen, welche es für seine eigentlichen Aufgaben gar nicht benötigt. Das Objekt wird also unabhängig von den konkreten Umsetzungen der Klassen die es nutzt, wodurch unnötige Abhängigkeiten beim Kompilieren vermieden werden.

swp11-2

Projektleiter : Matthias Haeßner | Dokument erstellt von David Müller

3. Beschreibung der zu studierenden Applikation

3.1 Zusammenspielt zwischen Tomcat, Spring und Brasato

Als Servlet-Container für OLAT wird standardmäßig Apache Tomcat verwendet. Der Client ruft über einen Webbrowser einen Link auf, dessen Ende von Tomcat als Parameter interpretiert und an das entsprechende Servlet weitergegeben wird. OLAT nutzt dabei folgende Servlets :

- OLATServlet
- SecureWebdavServlet
- Staticsservlet
- RSSServlet
- OlatDevServlet

Das OLATServlet stellt dabei den Hauptverarbeitungspunkt für eingehende Anfrage dar. Die Anfragen werden mittels Dispatcher (DMZDispatcher, AuthenticatedDispatcher, Shibbolethdispatcher) weiterverarbeitet, also an die vorher als zuständig ermittelte Komponente weitergegeben.

Zur Datenverarbeitung nutzt OLAT eine eigens entwickelte MVC-Architektur, das Brasato Framework(OLAT-Core). In diesem Framework sind die Schichten View und Model in einer abstrakten Klasse miteinander verwoben und werden mittels des Composite-Patterns in Subklassen wieder getrennt. Die abstrakte Basisklasse enthält dabei alle zu präsentierende Objekte. Der Controller der die jeweilige Komponente erstellt hat und dort als Listener angefügt ist erhält nun den Request als Event über die Controller.Event() Methode und bearbeitet ihn entsprechend. Wenn die komplette Businesslogik abgearbeitet ist wird mit Hilfe eines Renderes aus allen Container und Komponenten eine HTML-Antwortseite zusammengestellt

Aufgabe 2 17.04.2011

Softwaretechnikpraktikum SS 2011

swp11-2

Projektleiter : Matthias Haeßner | Dokument erstellt von David Müller

3.2 Konfigurationsdateien

Eine Konfigurationsdatei ist eine Datei , in der bestimmte Einstellungen (die Konfiguration) von Programmen gespeichert sind. OLAT verwendet unter WEB-INF/classes/serviceconfig folgende Konfigurationsdateien.

./olat.properties Hauptkonfigurationsdatei für OLAT

Diese Datei dient der Bestimmung ob OLAT mit voreingestellten Einstellungen startet oder nicht. In dieser Datei werden folgende Ordnerpfade angegeben: der Ordner, in dem während der Laufzeit erzeugte Daten generiert werden; die Logfile, in der die Aktionen der Webapplication protokolliert wird; das Wurzelverzeichnisses des Repositorys, welches zur Speicherung und Organisation der generierten Dateien dient.

Weiter hin werden in dieser Datei die Maximalwerte für Upload und Quotas in MB, sowie der zur Laufzeit verfügbaren Arbeitsspeicher angegeben. Außerdem wird die voreingestellte Sprache sowie die Sprache, die verwendet wird falls es zu Problemen mit der Ausgabe in der voreingestellten Sprache kommt bestimmt. In diesem Zusammenhang werden auch die Sprachen angegeben, die in der GUI(Benutzeroberfläche) als Ausgabesprache gewählt werden können. Des weiteren wird der zu verwendende Zeichensatz bestimmt(wodurch Ausgaben unter Verwendung bestimmter Alphabete ermöglicht oder unterbunden werden können).

Die Werte die dem E-Mail-Service betreffen und hier gesetzt werden können sind die Benennung des Local-Host (bei keiner Eingabe oder 'disable' wird der E-mail-suport deaktiviert), Angabe von Nutzernamen und Passwort, Bestimmung der Admin- sowie der Support- mailadresse und der maximalen größe von Anhängen an E-mails in MB.

Im Zusammenhang mit der Registration von neuen Usern werden ebenfalls verschiedene Werte gesetzt. Zu diesen Werten gehören folgende Angaben: Selbregistration erlauben oder verbieten, E-mailadresse an die Nachrichten gesandt werden, wenn sich ein neuer User registriert(sowie [de]aktivierung der Benachrichtigung bei der Zusendung einer Nachricht in oben genannten Fall), desweiteren kann die Anzeige, Auswahlmöglichkeit der Zustimmung, sowie die Möglichkeit zu einem Download eines Haftungsausschluss gesetzt werden. Es kann ausserdem bestimmt werden ob der Username vorbestimmt werden soll (und auf welche Art dies geschehen soll), ob Gastlogins erlaubt sind oder nicht und ob die User-bezogen Daten zu Mail und Login gespeichert bleiben sollen wenn der User gelöscht wird. Des weiteren werden die Verfügbaren Intervalle bestimmt werden, in denen der User von OLAT Benachrichtigungen erhält

Zudem ist die Auswahl von Onxy als Assesmentplugin zum Testen oder Olat für die tatsächlich Ausführung möglich sowie die Modifikation der technischen Einstellungen von OLAT (unter anderem kann der OLAT-Instanz eine eindeutig ID zugeordnet , ein Layouttheme gewählt, das Debuging ermöglicht, sowie Erweiterungen angegeben werden)

Einstellungen bezüglich Tomcat, Maven, Eclipse sowie der Verwendeten Datenbank, Instant-Messaging, automatisierter Übersetzung, Voltextsuche und Sicherheitsbestimmungen werden ebenfalls in

Aufgabe 2 17.04.2011

Softwaretechnikpraktikum SS 2011

swp11-2

Projektleiter : Matthias Haeßner | Dokument erstellt von David Müller
diesem Dokument verankert, ausserdem werden Variablen bezüglich des Clusterings und J-Unit gesetzt

Konfigurationsdateien für Einstellungen im Zusammenhang mit Spring:

./brasatoconfig.xml enthält die wichtigsten Variablen für Brasato

./org/olat/_spring/ Enthält insgesamt 6 Xml-Dateien für Einstellungen im Zusammenhang mit Spring

- portalContext.xml enthält die Variablen bezüglich der Portlets
- sitedefContext.xml enthält die Variablen bezüglich der Definition
- des Seitenlayouts
- olatextconfig.xml enthält die Variablen bezüglich der Controller
  1. und des Workflows
- brasatoconfigpart.xml enthält die Variablen, die im Zusammenhang
  1. mit Webanwendungen und Webpages stehen
- extensionContext.xml enthält Variablen im Zusammenhang mit der
  1. Anbindung von Extensions
- webdavContext.xml Modifizierung der Ordnerverwaltung

./org/olat/ommons/coordinate/cluster/_spring/olatdefaultconfig.xml

Standartkonfiguration für olat

./org/olat/core/_spring/olatcoreconfig.xml

Konfiguration für den OLAT-Kern

./org/olat/core/commons/scheduler/_spring/olattextconfig.xml

Ermöglicht das Erstellen von Jobs die in regelmäßigen Abständen wiederholt werden

./org/olat/notifications/_spring/olatdefaultconfig.xml

Verantwortlich für notifications

./org/olat/user/_spring/olatdefaultconfig.xml

Grundstruktur aller Config-XML-Dateien

Aufgabe 2 17.04.2011

Softwaretechnikpraktikum SS 2011

swp11-2

Projektleiter : Matthias Haeßner | Dokument erstellt von David Müller

Um die gewünschten Einstelungen in der OLAT-Instanz vor zunehmen müssen folgende Werte verändet werden

in ./olat.properties erfolgen folgende Änderungen:

login.enableGuestLoginLinks	wird auf false gesetzt und unterbindet so den Gastzugang
registration.enableSelfRegistration	wird ebenfalls auf false gesetzt und unterbindet so die Möglichkeit der Selbstregistration

`guidemo.enabled und portlet.macartney.enabled werden ebenfalls auf false gesetzt um somit die gui-demo und das macartney-portlet zu deaktivieren`