



FirstSpirit™

Unlock Your Content

FirstSpirit™ Dynamic Personalization FirstSpirit™ Version 5.0

Version	2.42
Status	RELEASED
Datum	2013-09-02
Abteilung	FS-Core
Copyright	2013 e-Spirit AG

Dateiname
DPER50DE_FirstSpirit_DynamicPersonalization

e-Spirit AG

Barcelonaweg 14
44269 Dortmund | Germany

T +49 231 . 477 77-0
F +49 231 . 477 77-499

info@e-spirit.com
www.e-spirit.com

e-Spirit

Inhaltsverzeichnis

1	Einleitung	5
1.1	Übersicht über die Funktionen	5
1.2	Thema dieser Dokumentation	7
1.3	Begriffe und Konzepte	7
1.3.1	Konzept Single Sign-On	7
1.3.2	FirstSpirit PermissionService	8
1.3.3	Login-Package	10
1.3.4	Login-Modul	12
1.3.5	Authentifizierungs-Modul	12
1.3.6	Gruppen-Modul	12
1.3.7	Attribut-Modul	13
2	Installation und Konfiguration	14
2.1	Installation des Moduls FirstSpirit DynamicPersonalization	14
2.2	Installieren der Webanwendung im Projekt	16
2.3	Konfiguration der Webanwendung	17
2.4	Konfiguration eines Login-Packages	20
2.5	Weitere Konfigurationsmöglichkeiten (web.xml)	21
2.5.1	Filter: NtlmPreAuthenticationFilter	21
3	Module	22
3.1	Login-Module	22
3.1.1	Login-Modul "Kerberos Login"	23
3.1.2	Login-Modul "NTLM Login"	25
3.1.3	Login-Modul "Anonymous NTLM Login"	26



- 3.1.4 Login-Modul "Request Parameter Login" 27
- 3.1.5 Login-Modul "Request Header Login" 27
- 3.1.6 Login-Modul "Portlet Login" 28
- 3.1.7 Login-Modul "SAP Portlet Login" 28
- 3.1.8 Login-Modul "FS SSO" 28
- 3.2 Authentifizierungs-Module 29
 - 3.2.1 Authentifizierungs-Modul „JDBC“ 30
 - 3.2.2 Authentifizierungs-Modul "LDAP" 32
 - 3.2.3 Authentifizierungs-Modul "ORMapper" 35
 - 3.2.4 Authentifizierungs-Modul "Property File" 36
 - 3.2.5 Authentifizierungs-Modul "Permission Service" 37
- 3.3 Gruppen-Module 38
 - 3.3.1 Gruppen-Modul „JDBC“ 39
 - 3.3.2 Gruppen-Modul "LDAP Group" 40
 - 3.3.3 Gruppen-Modul "LDAP Group Iterate" 41
 - 3.3.4 Gruppen-Modul "ORMapper" 43
 - 3.3.5 Gruppen-Modul "Portlet" 43
 - 3.3.6 Gruppen-Modul "SAP Portlet" 44
 - 3.3.7 Gruppen-Modul "FS SSO Groups" 44
 - 3.3.8 Gruppen-Modul "Permission Service" 44
- 3.4 Attribut-Module 45
 - 3.4.1 Attribut-Modul "LDAP" 45
 - 3.4.2 Attribut-Modul "ORMapper" 48
- 4 Servlets 49**
 - 4.1 "Login-Servlet" 49
 - 4.2 "Logout-Servlet" 51



5	Tags	52
5.1	Präfix	53
5.2	<fsp:loginRequired>	53
5.3	<fsp:authorize>	54
5.4	<fsp:userInfo>.....	55
5.5	<fsp:userGroups>.....	55
5.6	<fsp:userAttributes>	56
5.7	<fsp:isAuthorized>	57
5.8	<fsp:isNotAuthorized>.....	61
5.9	<fsp:logout>	61
6	Erweiterung der Funktionalität	62
6.1	Session-Variablen	62
6.2	Interface "User"	62
6.2.1	Methode "getLogin()"	63
6.2.2	Methode "getGroups()"	64
6.2.3	Methode "addGroup(String)".....	64
6.2.4	Methode "setGroups(List<String>)"	64
6.2.5	Methode "isInGroup(String)"	65
6.2.6	Methode "isInGroups(String)"	65
6.2.7	Methode "getAttributes()"	65
6.2.8	Methode "getAttribute(String)"	66
6.2.9	Methode "addAttributes(Map<String, String>)"	66
6.2.10	Methode "setAttribute(String, String)"	66
6.2.11	Methode "clearAttributes()"	67
7	Konfiguration mit JOSSO (Java Open Single Sign-On)	68
7.1	Konfiguration der Personalisierung.....	68



7.2 Konfiguration des FirstSpirit-Projekts..... 68

 7.2.1 Über Strukturvariablen oder Projekteinstellungen..... 68

 7.2.2 Konfiguration der personalisierten Seiten (Inhalte-Verwaltung).... 69

8 Rechtliche Hinweise 71



1 Einleitung

Über das Modul FirstSpirit™ DynamicPersonalization kann die Darstellung von FirstSpirit-Inhalten personalisiert werden. Das Modul stellt dazu unterschiedliche Möglichkeiten für Anmeldung (Login), Authentifizierung und das Auslesen benutzerspezifischer Informationen (Gruppenzugehörigkeit und weitere benutzerspezifische Attribute, z. B. Email, Telefonnummer, u.a.) bereit, die beliebig miteinander kombiniert werden können. Die Benutzerinformationen (z. B. Login-Name und Gruppenzugehörigkeit) können aus der FirstSpirit-Umgebung (FirstSpirit™ SSO) oder aus einem Fremdsystem (u.a. NTML, LDAP, etc.) stammen.

Die Darstellung der personalisierten Inhalte kann über eine spezielle Java-Tag-Library (die FirstSpirit-Personalization-Tags) konfiguriert werden. Abhängig von der Auswahl und der Konfiguration der Tags werden die Inhalte für bestimmte Benutzer oder Gruppen angezeigt oder ausgeblendet.

1.1 Übersicht über die Funktionen

Das Modul FirstSpirit™ DynamicPersonalization unterstützt die folgenden Aspekte:

- **Login und Authentifizierung über Fremdsysteme:** Das Modul FirstSpirit™ DynamicPersonalization unterstützt den Login und die Authentifizierung über Fremdsysteme. FirstSpirit™ DynamicPersonalization stellt dazu unterschiedliche Gruppen-, Login- und Authentifizierungs-Module bereit, die es dem Benutzer von FirstSpirit ermöglichen, sich über Fremdsysteme, wie LDAP, Kerberos, NTLM oder aus einer Portal-Umgebung heraus anzumelden. Die Authentifizierung erfolgt dabei üblicherweise über einen Cookie, ein Ticket oder ein HTML-Formular. Über die dort enthaltenen Anmelde-Informationen wird versucht, den Benutzer am FirstSpirit-Server zu authentifizieren.
- **Login und Authentifizierung über externe oder interne Datenquellen:** Neben der Authentifizierung über ein Fremdsystem (siehe oben) können die Authentifizierungs-Informationen auch über die Anbindung einer externen oder internen Datenquelle (an ein FirstSpirit-Projekt) zur Verfügung gestellt werden. Dabei werden über die FirstSpirit-Datenquellen-Verwaltung des Projekts Anmelde-Informationen in einer Datenbank hinterlegt, die als Grundlage für die Authentifizierung des Benutzers dienen (bei Anbindung einer internen Datenquelle). Bei Anbindung einer externen Datenquelle können die Anmelde-Informationen beispielsweise über eine JSP-Seite gepflegt werden. (Weitere Informationen zur Anbindung von Datenquellen siehe "FirstSpirit Handbuch für



Administratoren", Kapitel 4.8).

- **SSO:** Die ermittelten Login-Informationen können innerhalb der FirstSpirit-Umgebungen weiter verwendet werden (siehe Kap. 1.3.1 Seite 7). Für die Authentifizierung über FirstSpirit DynamicPersonalization bedeutet SSO, dass die Anmeldeinformationen des aktuellen Benutzers aus FirstSpirit oder aus einem Fremdsystem (z. B. LDAP) übernommen werden und an die Stelle von Login und Passwort in FirstSpirit treten. So kann aus jeder Umgebung heraus die Redaktionsumgebung oder der WebClient gestartet werden und der angemeldete Benutzer wird ohne erneuten Anmeldedialog auf dem FirstSpirit-Server authentifiziert.
- **Verwendung von Attributen und Gruppenzuordnungen aus Fremdsystemen:** Neben der reinen Authentifizierung eines Benutzers über ein Fremdsystem (z. B. LDAP), können auch die für einen Benutzer hinterlegten Informationen, z. B. Attribute und Gruppenzuordnungen, übernommen werden. Dazu stellt FirstSpirit DynamicPersonalization Gruppen- und Attribut-Module zur Verfügung (siehe Kapitel 1.3.5 f.).
- **Verknüpfung mit den FirstSpirit PermissionService:** FirstSpirit stellt ein Modul für die Vergabe und Auswertung von Benutzerrechten für die generierten Inhalte zur Verfügung. Benutzerrechte werden für den "Besucher" der generierten Site vergeben und sind damit immer mit dem eingesetzten Personalisierungssystem verknüpft (im Gegensatz zu den Redaktionsrechten). Wird als Personalisierungssystem FirstSpirit DynamicPersonalization eingesetzt (was nicht zwangsläufig der Fall sein muss), so kann ein sehr enger Zusammenhang hergestellt werden (siehe Kapitel 1.3.2.3 Seite 9). (Weitere Informationen zur Konfiguration siehe "FirstSpirit Handbuch für Administratoren" Kap. 12)
- **Personalisierter Zugriff auf Seiten:** Über FirstSpirit DynamicPersonalization kann ein personalisierter Zugriff auf bestimmte Navigationsbereiche bzw. Seiten realisiert werden. Die personalisierten Inhalte können dabei direkt für einen bestimmten Benutzer oder über die Zugehörigkeit des Benutzers zu einer bestimmten Gruppe konfiguriert werden. Auf diese Weise kann beispielsweise der personalisierte Zugriff eines Benutzers auf Kundenbereiche oder Mitarbeiterbereiche realisiert werden.
- **Personalisierter Zugriff für einzelne Inhaltsbereiche:** Neben dem personalisierten Zugriff auf ganze Navigationsbereiche bzw. Seiten, können innerhalb einer Seite auch einzelne Inhalte personalisiert werden. So kann beispielsweise ein Link innerhalb einer Seite ("Seite zum Bearbeiten im JavaClient öffnen") für bestimmte Benutzer bzw. Gruppen angezeigt und für andere Benutzer bzw. Gruppen ausgeblendet werden.



1.2 Thema dieser Dokumentation

Kapitel 2: Beschreibt die Konfiguration und Installation des Moduls FirstSpirit DynamicPersonalization (ab Seite 14).

Kapitel 3: Stellt die verschiedenen Module zum Login, zur Authentifizierung, zur Gruppenzugehörigkeit der Benutzer und zum Zugriff auf Benutzerattribute vor und erläutert die Konfiguration der Module über Parameter (ab Seite 22).

Kapitel 4: Erläutert die Verwendung des "Login-" und des "Logout"-Servlets (ab Seite 49).

Kapitel 5: Beschreibt die Tags aus der FirstSpirit-Personalization-Taglib und erläutert die Konfiguration und Funktionsweise anhand einiger Beispiele (ab Seite 49).

Kapitel 6: Behandelt Möglichkeiten, um die Funktionalität von FirstSpirit DynamicPersonalization gezielt an die Anforderungen eines Projekts anzupassen (ab Seite 62).

Kapitel 7: Ein Spezialfall für den Einsatz von FirstSpirit DynamicPersonalization, ist die Konfiguration mit JOSSO (Java Open Single Sign-On)¹. Das Kapitel erläutert alle notwendigen Schritte zur Installation und Konfiguration (ab Seite 68).

1.3 Begriffe und Konzepte

1.3.1 Konzept Single Sign-On

Der Begriff Single Sign-On (SSO) steht für die Möglichkeit, sich mit einer einzigen Anmeldung gegenüber einer Menge von Anwendungen zu authentifizieren, anstatt diesen Anmeldevorgang für jede Anwendung einzeln durchführen zu müssen.

Für die Verwendung des Moduls FirstSpirit DynamicPersonalization bedeutet SSO, dass die Anmeldeinformationen des aktuellen Benutzers aus FirstSpirit oder aus einem Fremdsystem (z. B. LDAP) übernommen werden und an die Stelle von Login und Passwort treten. So kann aus jeder Umgebung heraus die Redaktionsumgebung oder der WebClient gestartet werden und der angemeldete Benutzer wird ohne erneuten Anmeldedialog auf dem FirstSpirit-Server authentifiziert.

¹ <http://www.josso.org/>



Technisch umgesetzt wird dies mithilfe eines Cookies (z. B. FirstSpirit-SSO) oder eines Tickets (z. B. SAP Portal). Dieses wird von FirstSpirit bereitgestellt und enthält alle notwendigen Informationen wie Benutzername, Anmeldezeitpunkt und Ablaufdatum. Der FirstSpirit Server kann über eine spezielle Anmeldekonfiguration dieses Cookie bzw. Ticket entgegennehmen und verifizieren.

1.3.2 FirstSpirit PermissionService

Die Vergabe und Prüfung von Benutzerrechten (Rechtevergabe für die generierten Inhalte) erfolgt in FirstSpirit über das FirstSpirit-Permission-Modul. Das Modul ist im Standard-Lieferumfang von FirstSpirit enthalten und steht direkt nach der Installation zur Verfügung.

Das Modul besteht aus zwei Komponenten:

- Editor (Rechtedefinitions-Komponente) – (siehe Kapitel 1.3.2.1 Seite 8)
- Service (PermissionService) – (siehe Kapitel 1.3.2.2 Seite 9)

Zum Einsatz von FirstSpirit DynamicPersonalization mit der Rechtedefinitions-Komponente siehe Kapitel 1.3.2.3 Seite 9 .

1.3.2.1 Rechtedefinitions-Komponente (CMS_INPUT_PERMISSION)

Über die Rechtedefinitions-Komponente kann der FirstSpirit-Client um die Definition von Benutzerrechten (für die generierten Inhalte) erweitert werden. Die Komponente kommt in der Regel als Eingabekomponente im Metadaten-Reiter zum Einsatz. Die Komponente wird (projektspezifisch im Metadaten-Formular) mit einem eindeutigen Gruppendokument parametrisiert und arbeitet mit dem passenden Service zusammen, der die Gruppeneinstellungen vom Server lädt und bereitstellt (siehe Kapitel 1.3.2.2).

Zur Benutzung der Rechtedefinitions-Komponente siehe "FirstSpirit Handbuch für Redakteure (JavaClient)".



1.3.2.2 Permission-Service

Der Permission-Service ist eine Serverkomponente, die über die Rechtedefinitions-Komponente angesprochen werden kann. Der Permission-Service ist ein spezieller Dienst des FirstSpirit-Servers, dessen Aufgabe in der Verwaltung von Gruppen- und Benutzerkonfigurationen besteht. Als Systemdienst kann der Permission-Service über das FirstSpirit Server Monitoring (FirstSpirit / Steuerung / Dienste) oder über die Server- und Projektkonfiguration (Servereigenschaften / Module /System-Modul) aktiviert werden.

Weitere Informationen zum Starten und Stoppen von Systemdiensten siehe "FirstSpirit Handbuch für Administratoren".

1.3.2.3 Einsatz von FirstSpirit DynamicPersonalization mit der Rechtedefinitions-Komponente

Das Modul FirstSpirit DynamicPersonalization besteht aus einer Java-Tag-Library, die genutzt werden kann, um JSP-Seiten zu personalisieren, d.h. es kann entweder das Anzeigen der Seite komplett oder teilweise unterbunden werden (siehe Kapitel 5 Seite 52). Ob ein (Teil-)Bereich einer Seite sichtbar ist oder nicht hängt davon ab, ob der angemeldete Benutzer über eine ausreichende Berechtigung verfügt. Dies wird entschieden, indem die Soll-Berechtigung der Seite mit der Ist-Berechtigung des Benutzers verglichen wird.

Die Definition der Soll-Berechtigung (Wer darf ein Dokument oder einen Teil des Dokumentes sehen?) erfolgt in diesem Fall über die Rechtedefinitions-Komponente. Die Auswertung, d.h. der Vergleich der Ist-Konfiguration (die sich aus dem Login-Kontext des Benutzers ergibt) mit der Soll-Konfiguration wird über FirstSpirit DynamicPersonalization durchgeführt. Dabei besteht ein sehr enger (logischer) Zusammenhang zwischen den Gruppen, die in der Rechtedefinitions-Komponente verwendet werden und den Gruppen, auf die sich das Gruppen-Modul der Personalisierung bezieht.

Beispiel: Wenn ein Benutzer zu einer Gruppe gehört, die nicht in der Rechtedefinitions-Komponente auftaucht, können für ihn unter Umständen keine Rechte gesetzt werden.



Dieser Zusammenhang zwischen Gruppen-Modell in der Rechtedefinitions-Komponente und im Personalisierungs-Modul kann auf unterschiedlichen Wegen hergestellt werden:

1. externe Quelle: Die Erzeugung der Gruppen-Definitionsdatei für die Rechtedefinitions-Komponente und das Gruppen-Modul der Personalisierung werten die gleichen externen Datenquellen aus (z. B.: LDAP-Server oder AD-Server). In diesem Fall liegt die Datenhoheit komplett extern.
2. interne Quelle: Die Rechtedefinitions-Komponente erzeugt eine Gruppen-Definitionsdatei und das Personalisierungs-Modul verwendet diese Daten bzw. bezieht sich auf diese Datei. Die Datei kann von der Rechtedefinitions-Komponente durchaus durch die Abfrage eines LDAP-Servers erzeugt werden. Der Unterschied ist, dass das Personalisierungsmodul selbst nicht den LDAP-Server abfragt, sondern die von der Rechtedefinitions-Komponente erzeugte Datenbasis verwendet. Die Gruppendifinition kann auch direkt über FirstSpirit, beispielsweise über die Datenquellen-Verwaltung, vorgenommen werden.

Während also im ersten Fall das Gruppen-Modul der Personalisierung auf die Modi "JDBC" oder "LDAP" konfiguriert wird (und damit die Laufzeitumgebung eine permanente Verbindung zur Datenquelle benötigt), ist für den zweiten Fall das Gruppen-Modul auf "GroupService" zu konfigurieren und verwendet dann die gleichen XML-Dateien wie die Rechtedefinitions-Komponente. Im zweiten Fall ist daher kein permanenter Zugriff auf eine externe Ressource erforderlich. Es ist aber zu beachten, dass bei der Deployment-Konfiguration die Gruppen-Konfigurationsdateien auch auf das Live-System veröffentlicht werden.

1.3.3 Login-Package

FirstSpirit DynamicPersonalization ist in vier unterschiedliche Modularten unterteilt:

- Login-Module zum Ermitteln von Login-Informationen (siehe Kapitel 1.3.4 Seite 12).
- Authentifizierungs-Module zum Auswerten der ermittelten Login-Informationen (siehe Kapitel 1.3.5 Seite 12)
- Gruppen-Module zum Auslesen von Gruppeninformationen eines Benutzers (siehe Kapitel 1.3.6 Seite 12)
- Attribut-Module zum Auslesen von Benutzerattributen (siehe Kapitel 1.3.7 Seite 13)

Als "Login-Package" wird der Zusammenschluss je eines Login-Moduls, eines Authentifizierungs-Moduls und optional eines Gruppen-Moduls und/oder Attribut-



Moduls bezeichnet. Ein Login-Package kann also auch nur aus einem Login- und einem Authentifizierungs-Modul gebildet werden, wenn Gruppeninformationen und Attribute nicht benötigt werden. Auf das Authentifizierungs-Modul kann, abhängig vom verwendeten Login-Modul, in einigen Fällen ebenfalls verzichtet werden. So wird beispielsweise beim Kerberos- oder NTLM-Login-Modul keine Auswertung der Login-Daten benötigt, da dies bereits beim Auslesen der Login-Daten geschieht.

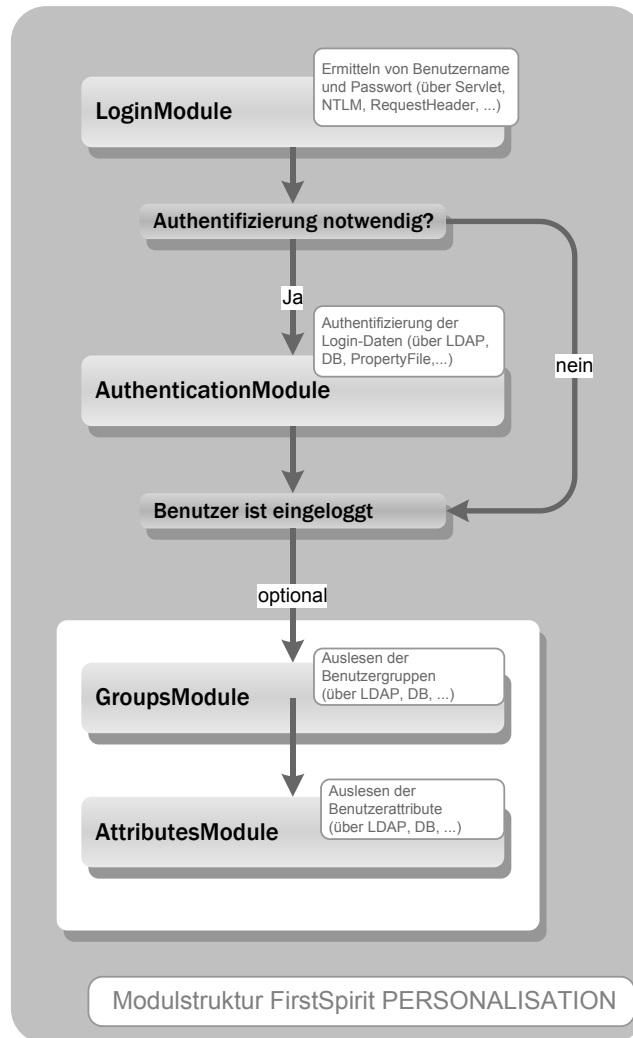


Abbildung 1-1: Modulstruktur von FirstSpirit DynamicPersonalization

Bei der Authentifizierung wird zunächst versucht den Benutzer gegen das erste Login-Package (mit der höchsten Priorität) zu authentifizieren. Gelingt dies nicht, wird versucht den Anwender über das Login-Package mit der zweithöchsten Priorität anzumelden, usw., bis der Anmeldeversuch entweder erfolgreich ist oder nach dem Login-Package mit der niedrigsten Priorität abgebrochen werden muss. Es "gewinnt" das erste Login-Package mit dem der Benutzer erfolgreich authentifiziert werden kann (siehe Kapitel 2.3 Seite 17)

Weitere Informationen zu den Modulen, deren Abhängigkeiten und zur detaillierten



Konfiguration befinden sich in den nachfolgenden Kapiteln.

1.3.4 Login-Modul

Das Login-Modul ist ein Pflichtmodul. Das bedeutet, jedes Login-Package muss ein Login-Modul enthalten.

Es ermittelt die Benutzerinformationen, die für die Authentifizierung des Benutzers nötig sind, üblicherweise der Login-Name und das Passwort. Es kann sich jedoch auch um ein Cookie (z. B. bei Authentifizierung über FirstSpirit-SSO) oder ein Ticket (z. B. bei Authentifizierung über Kerberos, NTLM oder SAP Portal) handeln. Die Authentifizierung wird anschließend, basierend auf diesen Daten, vom Authentifizierungs-Modul durchgeführt (siehe Kapitel 1.3.5 Seite 12).

Zur Konfiguration des Login-Moduls siehe Kapitel 3.1 Seite 22.

1.3.5 Authentifizierungs-Modul

Das Authentifizierungs-Modul ist abhängig vom verwendeten Login-Modul entweder ein Pflicht-Modul oder es kann optional verwendet werden. Über das Authentifizierungs-Modul kann der Benutzer, anhand der ermittelten Login-Informationen (aus dem Login-Modul), gegen ein Fremdsystem, eine FirstSpirit Datenbank oder gegen den FirstSpirit-Server authentifiziert werden (siehe Kapitel 1.3.4.2 Seite 12). Die Authentifizierungs-Instanz ist abhängig vom gewählten Authentifizierungs-Modul (z. B. "LDAP" – Authentifizierung des Benutzers erfolgt gegen den LDAP-Server).

Es kann jedoch möglich sein, dass die ermittelten Login-Informationen bereits an anderer Stelle authentifiziert wurden, so dass eine nachträgliche Authentifizierung nicht mehr notwendig ist.

Zur Konfiguration des Authentifizierungs-Moduls siehe Kapitel 3.2 Seite 29.

1.3.6 Gruppen-Modul

Das Gruppen-Modul kann optional eingesetzt werden, um die Gruppen-Informationen für den angemeldeten Benutzer auszulesen. Gruppeninformationen sind für einen Benutzer nur verfügbar, sofern diese Informationen im Fremdsystem (oder über FirstSpirit) vorhanden sind und der Benutzer bereits erfolgreich authentifiziert wurde.

Der Einsatz der optionalen Gruppen-Module ist nur notwendig, wenn zwischen



einzelnen Gruppen unterschieden werden soll, beispielsweise wenn die Anzeige bestimmter Bereiche einer Seite bestimmten Gruppen vorbehalten ist. In diesem Fall, kann die Gruppenzugehörigkeit des angemeldeten Benutzers für die Steuerung der Anzeige herangezogen werden.

Zur Konfiguration des Gruppen-Moduls siehe Kapitel 3.3 Seite 38.

1.3.7 Attribut-Modul

Das Attribut-Modul kann optional eingesetzt werden, um weitere Attribute für den angemeldeten Benutzer auszulesen. Dabei handelt es sich beispielsweise um eine Telefonnummer, Email-Adresse, Raumnummer, usw. Diese Attribute sind für einen Benutzer nur verfügbar, sofern diese Informationen im Fremdsystem (oder über FirstSpirit) vorhanden sind und der Benutzer bereits erfolgreich authentifiziert wurde.

Zur Konfiguration des Attribut-Moduls siehe Kapitel 3.4 Seite 45.



2 Installation und Konfiguration

Die Konfiguration von FirstSpirit DynamicPersonalization erfolgt in erster Linie über die FirstSpirit Anwendung zur Server- und Projektkonfiguration.

2.1 Installation des Moduls FirstSpirit DynamicPersonalization

Das Modul FirstSpirit DynamicPersonalization muss zunächst innerhalb der Anwendung zur Server- und Projektkonfiguration installiert werden. Im Bereich Servereigenschaften wird dazu der Menüeintrag "Module" selektiert. Mit einem Klick auf den Button "Installieren" öffnet sich ein Dateiauswahldialog. Hier kann die zu installierende fsm-Datei (fs-perso.fsm) ausgewählt werden. Die erfolgreich installierte Datei wird anschließend im Dialog "Server Eigenschaften" angezeigt:

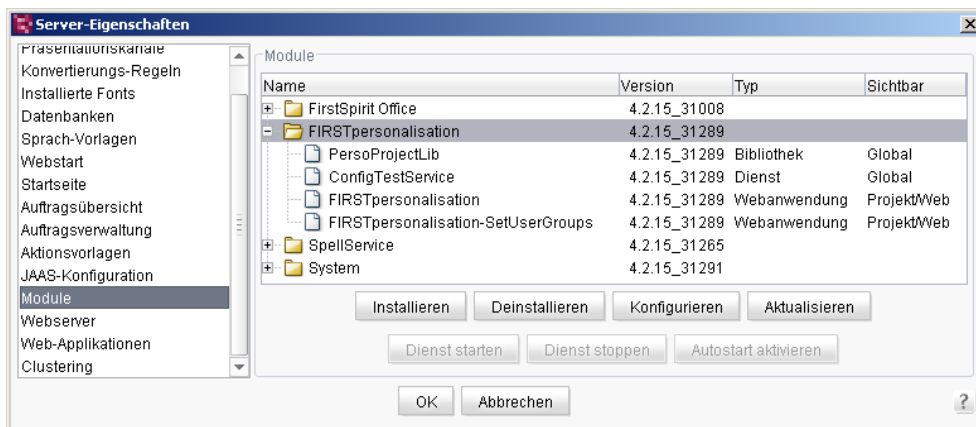


Abbildung 2-1: Installation des Moduls auf dem FirstSpirit-Server

Bestandteil des Moduls FirstSpirit DynamicPersonalization ist die globale Bibliothek "PersonalisationProjectLib" (siehe Kapitel 5 Seite 52), die Klassen beinhaltet, die nach der Installation auf dem FirstSpirit-Server, innerhalb der Clients, in Skripten und anderen Modulen verfügbar sind.

Außerdem enthält das Modul einen globalen Dienst "ConfigTestService", zum Testen der Konfiguration (Beispiel siehe Kapitel 3.2.1 Seite 31) und drei Webanwendungen, die im jeweiligen Projekt installiert werden können.

Die Webanwendung "FIRSTpersonalisation" stellt JSP-Tags (siehe Kapitel 4 Seite 49) und Servlets (siehe Kapitel 5 Seite 52) zur Verfügung, die innerhalb des Projekts verwendet und aufgerufen werden können.

Die Webanwendung "FIRSTpersonalisation SetUserGroups" kann genutzt werden,



um eine Testumgebung für den Redakteur zu erstellen. Die Webkomponente ermöglicht die Definition von Benutzergruppen für den eingeloggten Benutzer. Ist beispielsweise ein "Redakteur" über "FirstSpirit SSO" gegen den FirstSpirit-Server authentifiziert, stellt die Webkomponente Mittel zur Verfügung, um die aktuelle Seite für einen Benutzer der Gruppe "Kunden" darzustellen. Innerhalb der Vorschau-Umgebung greifen dabei die Sicherheitsmechanismen des Preview-Servlets. Das bedeutet, die Rechte eines Benutzers können über die Webanwendung zwar eingeschränkt, aber nicht erweitert werden (das gilt nicht für die Staging- bzw. Live-Umgebung). Der Benutzer der Gruppe "Redakteure" kann sich also beispielsweise die "Kunden"-Bereiche der Website ansehen, nicht aber die aktuellen Geschäftsberichte der Gruppe "Manager", für die er als "Redakteur" keine Rechte besitzt.



Die Webanwendung "FIRSTpersonalisation SetUserGroups" stellt lediglich die Möglichkeiten für die Definition von Benutzergruppen zur Verfügung. Die Realisierung im Projekt (Aufruf des Servlets) muss durch den Vorlagenentwickler kundenspezifisch für das jeweilige Projekt durchgeführt werden.

Beide Komponenten sind "Sichtbar" für die Bereiche "Projekt/Web". Damit handelt es sich um eine "web-lokale" Komponente. Diese kann nach der Installation den unterschiedlichen Web-Bereichen (preview, staging, live) innerhalb der gewünschten Projekte zugefügt werden (siehe Kapitel 2.2 Seite 16).

Eine ausführliche Beschreibung zur Installation eines Moduls findet sich im "FirstSpirit Handbuch für Administratoren".



2.2 Installieren der Webanwendung im Projekt

Die Webanwendung muss nun im gewünschten Projekt installiert werden. Dazu wird innerhalb der Projekteigenschaften der Menüeintrag "Web-Komponenten" aufgerufen. In diesem Bereich können die Web-Komponenten für ein Projekt aktiviert werden.



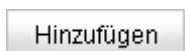
Abbildung 2-2: Installation der Webanwendung innerhalb der Web-Bereiche

Es existieren für jedes Projekt drei unterschiedliche Web-Bereiche. Über die jeweilige Registerkarte können die Web-Komponenten für jeden Bereich einzeln aktiviert und konfiguriert werden:



Abbildung 2-3: Web-Bereiche innerhalb eines Projekts

- Vorschau (Preview): Ort für die Projektinhalte für die eine Vorschau angefordert wurde.
- QA (Staging): Ort für die generierten Projektinhalte
- Produktion (Live): Ort für die veröffentlichten Projektinhalte



Mit einem Klick auf den Button öffnet sich der Dialog "Hinzufügen". In der Liste werden alle Web-Komponenten angezeigt, die auf dem Server installiert sind (siehe Kapitel 2.1 Seite 14).

Nach dem Hinzufügen zu einem Webbereich, besteht die Möglichkeit, die Komponenten zu konfigurieren, entweder mit einer von der Komponente erzeugten oder einer generischen GUI (siehe Kapitel 2.3 Seite 17). Nach der Konfiguration müssen die Komponenten noch aktiviert werden. Dabei kann eine Komponente innerhalb eines Projekts nur für bestimmte Bereiche aktiviert bzw. deaktiviert werden

Weitere Informationen zu diesem Dialog siehe "FirstSpirit Handbuch für



Administratoren".

2.3 Konfiguration der Webanwendung

Konfigurieren Mit einem Klick auf den Button (vgl. Abbildung 2-2) kann die Komponente innerhalb des Web-Bereichs konfiguriert werden. Im Konfigurationsdialog können neue Login-Packages angelegt und bestehende bearbeitet werden:

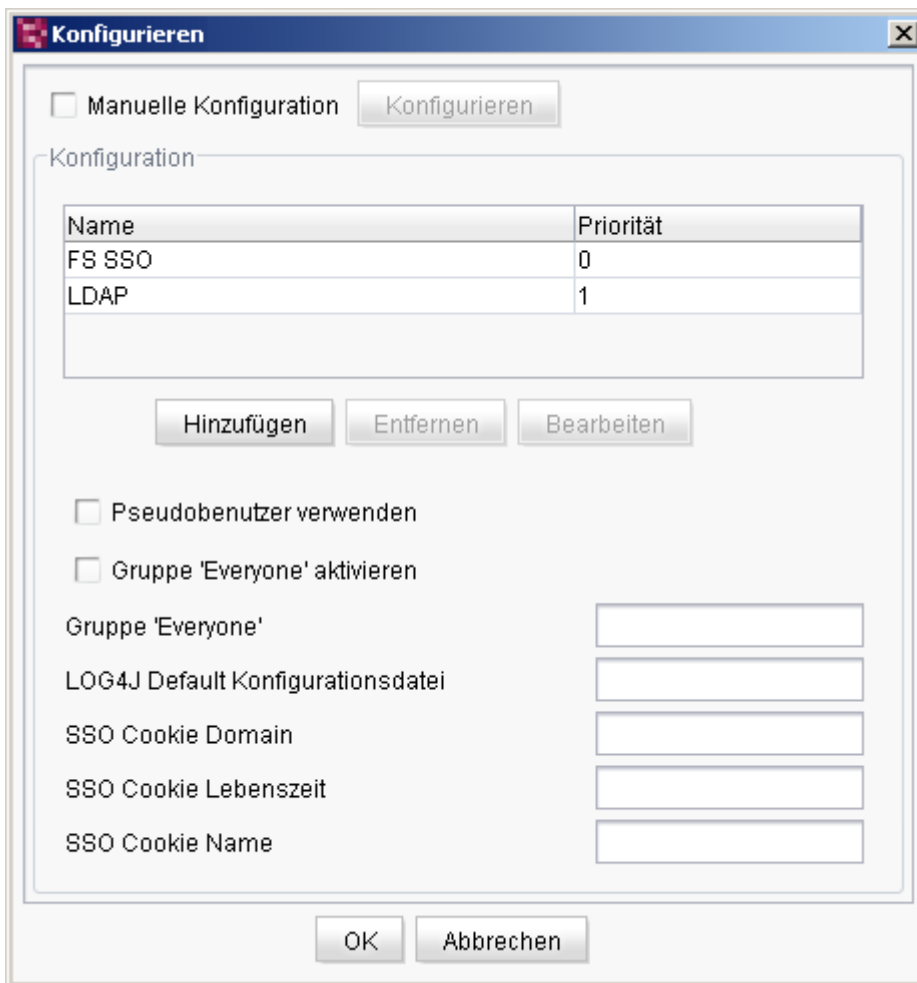


Abbildung 2-4: Konfigurieren der Webanwendung Integration

Manuelle Konfiguration **Konfigurieren** Wird die Checkbox "manuelle Konfiguration" aktiviert, kann mit einem Klick auf den Button die Datei "web.xml" der Webanwendung geöffnet werden (siehe Kapitel 2.5 Seite 21).

Hinzufügen Mit einem Klick auf den Button können neue Login-Packages definiert



werden (siehe Kapitel 2.4 Seite 20).

Name: Eindeutiger Name des Login-Packages. Dieser Name wird beim Hinzufügen eines Login-Packages vom Administrator vergeben.

Priorität: Die Priorität (natürliche Zahl ≥ 0) bestimmt die Abfolge der Abarbeitung der Login-Packages: Beim Anmeldeversuch eines Anwenders wird zunächst versucht, ihn über das Login-Package mit der höchsten Priorität (Priorität "0") anzumelden. Schlägt dieser Versuch fehl, wird versucht den Anwender über das Login-Package mit zweithöchster Priorität anzumelden, usw. bis der Anmeldeversuch entweder erfolgreich ist oder nach dem Login-Package mit der niedrigsten Priorität abgebrochen werden muss.

Beispiel: Es werden mehrere Login-Packages angegeben, die in der Reihenfolge ihrer Priorität zur Authentifizierung des Benutzers verwendet werden. So könnte das Modul FirstSpirit DynamicPersonalization beispielsweise so konfiguriert sein, dass ein Benutzer mit erster Priorität über den "ORMapper" authentifiziert wird. Schlägt dieser Authentifizierungsversuch fehl, wird das Login-Package mit nächst höherer Priorität verwendet - dies könnte z. B. die Authentifizierung gegen LDAP sein. Schlägt auch dies fehl, wird abhängig von der Priorität das nächste Login-Package verwendet (z. B. "Property File"). Solange die Authentifizierung des Benutzers nicht erfolgreich durchgeführt werden kann, wird immer mit dem Login-Package mit nächst höherer Priorität weiter gearbeitet, bis schließlich nach dem Login-Package mit der niedrigsten Priorität der Anmeldeversuch abgebrochen wird. In diesem Fall kann der Benutzer nicht am System angemeldet werden.

Sollte ein Login-Package eine erfolgreiche Authentifizierung durchgeführt haben, beispielsweise das zweite Login-Package, dass im Beispiel gegen LDAP authentifiziert, so werden die Login-Packages mit niedrigerer Priorität ignoriert und der Benutzer gilt als angemeldet.

Neben der Definition der Login-Packages können über diesen Dialog weitere globale Einstellungen vorgenommen werden:

Pseudobnutzer verwenden: Wird diese Checkbox aktiviert, wird die gesamte Personalisierung ausgeschaltet, da jeder Benutzer automatisch als Pseudobnutzer angemeldet ist. Diese Einstellung wird eventuell zum Testen benötigt, um Seiten zu betrachten, die nur für angemeldete Benutzer sichtbar sein dürfen, in denen aber noch keine Anbindung an eine Benutzerprüfung existiert. Standardmäßig ist die Checkbox nicht aktiviert.

Gruppe 'Everyone' aktivieren: Wird diese Checkbox aktiviert, so wird jedem angemeldeten Benutzer automatisch eine bestimmte Gruppe zugewiesen. Damit wird sichergestellt, dass alle Benutzer zu einer gemeinsamen Gruppe gehören. Die



gewünschte Gruppe wird über den Parameter "Gruppe 'Everyone'" festgelegt. Standardmäßig ist die Checkbox nicht aktiviert.

Gruppe 'Everyone': Mit dem Eingabefeld "Gruppe 'Everyone'" kann die Gruppe angegeben werden, die für die Gruppe "Everyone" verwendet werden soll. Als Wert kann eine beliebige gültige Gruppe angegeben werden. Standardmäßig ist der Wert für die Gruppe "Everyone": "*". Wenn die Checkbox "Gruppe 'Everyone'" aktiviert ist, ist jeder angemeldete Benutzer automatisch Mitglied der Gruppe "Everyone".

LOG4J Default Konfigurationsdatei: Dieser Parameter gibt den absoluten Pfad zur Konfigurationsdatei für das Logging an und wird für den Fall benötigt, dass kein durch den Applikation-Server vorkonfiguriertes Log4j vorhanden ist.

SSO Cookie Domain: Domain, für die der Cookie gültig ist. Der hier eingetragene Wert ist nur relevant, wenn für das verwendete Authentifizierungs-Modul die Checkbox "Erstelle Cookie" aktiviert wurde. Diese Einstellung kann auch im Login-Modul konfiguriert werden, wenn kein Authentifizierungs-Modul notwendig ist (z. B. Kerberos oder NTLM).

SSO Cookie Lebenszeit: Lebenszeit des Cookies bzw. Zeitpunkt der automatischen Löschung des Cookies. Der hier eingetragene Wert ist nur relevant, wenn für das verwendete Authentifizierungs-Modul die Checkbox "Erstelle Cookie" aktiviert wurde

SSO Cookie Name: Beliebiger Name des Cookies. Der hier eingetragene Wert ist nur relevant, wenn für das verwendete Authentifizierungs-Modul die Checkbox "Erstelle Cookie" aktiviert wurde



2.4 Konfiguration eines Login-Packages

Hinzufügen

Mit einem Klick auf den Button wird ein neues Login-Package konfiguriert (vgl. Abbildung 2-3). Innerhalb eines Login-Packages darf jede Art von Modul maximal einmal vorhanden sein. Welches Modul verwendet werden soll, wird über folgenden Dialog konfiguriert:

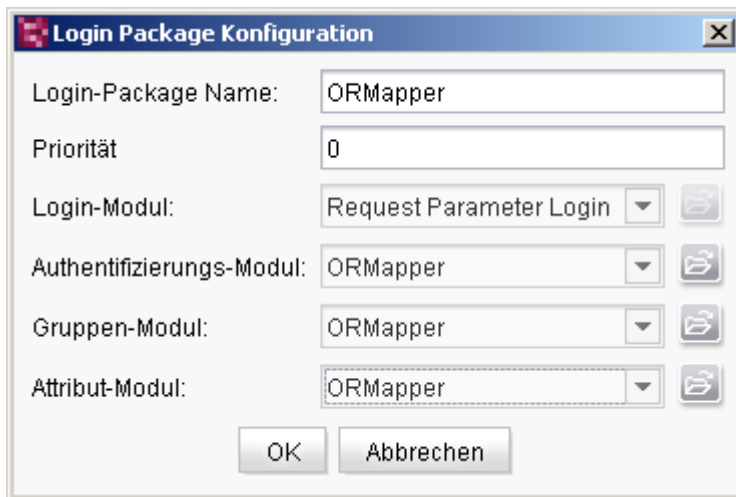


Abbildung 2-5: Login Package Konfiguration

Die einzelnen Module werden dann anhand ihrer Parameter konfiguriert. Welche Parameter benötigt werden bzw. möglich sind, ist abhängig vom jeweiligen Modul (siehe Kapitel 3 Seite 22):

- Login-Package-Name und Priorität (siehe Kapitel 2.3 Seite 17)
- Konfiguration Login-Modul: (siehe Kapitel 3.1 Seite 22)
- Konfiguration Authentifizierungs-Modul (siehe Kapitel 3.2 Seite 29)
- Konfiguration Gruppen-Modul (siehe Kapitel 3.3 Seite 38)
- Konfiguration Attribut-Modul (siehe Kapitel 3.4 Seite 45)



2.5 Weitere Konfigurationsmöglichkeiten (web.xml)

Jede FirstSpirit Webanwendung stellt eine Konfigurations-Datei "web.xml" bereit. Für eine herkömmliche Konfiguration muss diese Datei nicht manuell bearbeitet werden, da Änderungen direkt über die FirstSpirit GUI vorgenommen werden können.

Für spezielle Konfigurationsmöglichkeiten, beispielsweise das Einrichten von Filtern, kann die web.xml jedoch auch direkt bearbeitet werden (Zur Bearbeitung der "web.xml" siehe "FirstSpirit Handbuch für Administratoren"). Hier sollten aber nur die Einstellungen vorgenommen werden, die über das Konfigurieren der Module in der Server- und Projektkonfiguration hinausgehen.

2.5.1 Filter: NtlmPreAuthenticationFilter

Der optionale Filter "NtlmPreAuthenticationFilter", wird nur im Zusammenhang mit einer konfigurierten NTLM-Authentifizierung benötigt. Dieser Filter muss vor alle Servlets geschaltet werden, die durch einen POST-Request aufgerufen werden. Soll FirstSpirit DynamicPersonalization ohne NTLM-Authentifizierung laufen oder keine HTML-Formulardaten über http-POST gesendet, wird dieser Filter nicht benötigt.

```
<filter>
  <filter-name>NtlmPreAuthenticationFilter</filter-name>
  <filter-class>
de.espirit.firstspirit.opt.personalisation.filters.NtlmPreAuthentica
tionFilter
  </filter-class>

  <init-param>
    <param-name>domainController</param-name>
    <param-value>myDomainController</param-value>
  </init-param>
</filter>

<filter-mapping>
  <filter-name>NtlmPreAuthenticationFilter</filter-name>
  <servlet-name>myServlet</servlet-name>
</filter-mapping>
```

Für das Tag "<param-value>" muss ein gültiger Domain-Controller angegeben werden.

Das Filter-Mapping muss für jedes vorhandene Servlet, das durch einen POST-Request aufgerufen wird, eingetragen werden.



3 Module

3.1 Login-Module

Die Login-Module übernehmen das Ermitteln der Login-Daten (Login-Name und -Passwort) eines Benutzers (siehe Kapitel 1.3.4 Seite 12). Die Authentifizierung erfolgt anschließend über das ausgewählte Authentifizierungs-Modul. Es kann jedoch möglich sein, dass die ermittelten Login-Daten bereits an anderer Stelle authentifiziert wurden, so dass eine nachträgliche Authentifizierung nicht mehr notwendig ist. Dies gilt für alle Login-Module mit Ausnahme des Moduls "Request Parameter Login".

Folgende Login-Module stehen zur Verfügung:

- Kerberos Login
Anmeldung über Kerberos/SPNEGO (Integrierte Windows-Anmeldung, IWA)
(vgl. Kapitel 3.1.1 Seite 23).
- NTLM Login
Anmeldung über NTLM
(vgl. Kapitel 3.1.2 Seite 25).
- Anonymous NTLM Login
Anmeldung mit einem festen NTLM-Benutzer (anonyme Anmeldung)
(vgl. Kapitel 3.1.3 Seite 26).
- Request Parameter Login
Anmeldung mithilfe von Request-Parametern
(vgl. Kapitel 3.1.4 Seite 27).
- Request Header Login
Anmeldung mithilfe des Request-Headers
(vgl. Kapitel 3.1.5 Seite 27).
- Portlet Login
Anmeldung über einen IBM WebSphere Portal Server
(vgl. Kapitel 3.1.6 Seite 28).
- SAP Portlet Login
Anmeldung über einen SAP Portal Server
(vgl. Kapitel 3.1.7 Seite 28).
- FS SSO
Anmeldung über FirstSpirit Single Sign On (vgl. Kapitel 3.1.8 Seite 28)



3.1.1 Login-Modul "Kerberos Login"

Dieses Modul dient zur passwortlosen automatischen Anmeldung über Kerberos 5 und SPNEGO. Kerberos ist auf Server-Seite von allen aktuellen Unix-Systemen unterstützt und bei Microsoft Active Directory seit Windows 2003 / XP, dort als "Integrierte Windows-Anmeldung" (Integrated Windows Authentication, IWA) bekannt, als Ersatz für NTLM. Auf Client-Seite wird es von allen Desktop-Betriebssystemen unterstützt und den meisten Web-Browsern: Mozilla Firefox, Microsoft Internet Explorer, Safari, KDE, Gnome und andere.

Die Spezifikation zu Kerberos empfiehlt, Anmeldungen über Kerberos nur über sichere Datenkanäle zu verwenden, also beispielsweise *https*, aufgrund der theoretisch möglichen Gefahr von Replay-Attacks in einem kurzen Zeitraum nach der Anmeldung. Ein Auslesen des verschlüsselten Anmelde-Tickets oder Fälschen eines Anmelde-Tickets ist aber auch bei Verwendung von unverschlüsseltem *http* nicht möglich.

Parameter:

`useFullPrincipal`: Legt fest, ob der vollständige Kerberos-Anmeldename inklusive @-Zeichen und Kerberos-Realm (Wert "`true`") oder ohne @ und Kerberos-Realm (Wert "`false`") als Benutzername z. B. an das Gruppen-Modul weitergereicht wird. "`false`" ist bei Systemen ausreichend, deren Benutzerkonten alle in einem Kerberos-Realm (entspricht unter Windows einer Active Directory Domain) eingetragen sind. Falls Anmeldungen von mehreren Kerberos-Realms oder Active Directory Domains erfolgen, muss "`true`" angegeben werden, da meistens der reine Benutzername über mehrere Domains nicht eindeutig ist.
Standardwert: "`false`"

`userAgents`: Hier kann ein Suchmuster auf die Browser-Kennung angegeben werden, um die Kerberos-Anmeldung nur für ausgewählte Web-Browser zu aktivieren, da Kerberos einen nicht ganz standardkonformen HTTP-Header ("WWW-Authenticate: Negotiate") verwendet, den einige ältere Web-Browser als Fehler interpretieren. Um Kerberos bei allen Web-Browsern zu verwenden, ".*" eintragen.
Standardwert:
".*(Firefox|Iceweasel|Konqueror|MSIE|Opera|Safari|Shiretoko).*"



3.1.1.1 Konfiguration des Kerberos-Login-Moduls

Zusätzlich zu den Einstellungen über die Server- und Projektkonfiguration ist bei Verwendung des Kerberos-Login-Moduls folgende weitere Konfiguration notwendig:

Jedem Hostnamen eines Web-Servers wird ein eindeutiger Kerberos Service Principal Name (SPN) zugeordnet, der sich wie folgt ergibt:

```
HTTP/hostname.domain.tld@REALM
```

"HTTP" als Bezeichner wird auch bei https verwendet!

"hostname.domain.tld" ist der vollständige DNS-Hostname inklusive Domain, wie er im Web-Browser zur Benutzung des Web-Servers auf Client-Seite eingegeben wird.

"REALM" ist der so genannte Kerberos-Realm, der meistens mit der Domain übereinstimmt, aber immer vollständig in Großbuchstaben geschrieben wird.

Zu einem SPN müssen ein oder mehrere Schlüssel in verschiedenen Krypto-Verfahren erstellt werden, die in einer Kerberos-Keytab-Datei zusammengefasst werden. Diese Kerberos-Keytab-Datei wird dem Application-Server als Java-Property beim Start übergeben. Sofern Tomcat als Application-Server eingesetzt wird, folgenden Parameter in CATALINA_OPTS ergänzen, bei anderen Application-Servern können möglicherweise Java-Properties direkt eingegeben werden:

```
-Djava.security.auth.login.config=/pfad/zur/jaas.conf
```

Die Datei `jaas.conf` muss folgende Zeilen enthalten:

```
com.sun.security.jgss.accept {
  com.sun.security.auth.module.Krb5LoginModule required
  principal="HTTP/www.mydomain.net@MYDOMAIN.NET"
  keyTab="/pfad/zur/krb5-www_mydomain_net-HTTP.keytab"
  useKeyTab="true"
  storeKey="true"
  isInitiator="false"
  doNotPrompt="true"
  debug="false";
};
```

Bei "principal" wird der zuvor erwähnte SPN eingetragen, der den Hostnamen des Webservers enthält, wie er aus Browser-Sicht verwendet wird. Während der Einrichtungsphase sollte der Parameter "debug" auf "true" gestellt werden, um Kerberos-Fehlermeldungen im stdout/stderr-Logging des Application-Servers ablesen zu können (bei Tomcat in catalina.out). Zusätzlich wird je nach DNS-Konfiguration des lokalen Netzes eine Datei `/etc/krb5.conf` bzw. `c:\windows\krb5.ini`



auf dem Applikation-Server benötigt. Zu den weiteren Parametern der jaas.conf-Datei sowie zur Erzeugung der beim Parameter "keyTab" angegebenen Kerberos-Keytab-Datei und dem Inhalt der krb5.conf-Datei, siehe FirstSpirit-Admin-Handbuch Kapitel "Kerberos-Ticket (Integrierte Windows-Anmeldung)".

3.1.2 Login-Modul "NTLM Login"

Authentifiziert einen Benutzer automatisch ohne Passworteingabe über NTLM. Dieses Modul benötigt kein Authentifizierungs-Modul.

Es wird ausschließlich NTLM v1 unterstützt, nicht das seit Windows 2003 als Standard verwendete NTLM v2. Windows 2003 kann zwar auf NTLM v1 umgeschaltet werden, was aber aus Sicherheitsgründen nicht empfehlenswert ist. In diesem Fall sollte Kerberos (Kapitel 3.1.1 Seite 23) verwendet werden, was bei Microsoft auch als "Integrierte Windows-Anmeldung" (Integrated Windows Authentication, IWA) bezeichnet wird.



Abbildung 3-1: Login-Modul "NTLM Login"

Domain-Controller: Name des zu verwendenden Domain-Controllers. Über diesen Parameter werden die zur Anmeldung zugelassenen Windows-Domänen angegeben. Optional können Domänen-Server angegeben werden. Es können mehrere Domains eingetragen werden, die alle nacheinander zur Anmeldung überprüft werden. Als Trennzeichen wird ; verwendet.

Erstelle Cookie: Wird die Checkbox aktiviert, wird ein Cookie für die Authentifizierung über SSO erstellt (Konzept siehe Kapitel 1.3.1 Seite 7). Die Konfiguration des Cookies (Domain, Name, Lebenszeit) erfolgt über die Webanwendung (siehe Kapitel 2.3 Seite 17).

Zur NTLM-Authentifizierung siehe auch Kapitel 2.5.1 Seite 21 "NtlmPreAuthenticationFilter".



3.1.3 Login-Modul "Anonymous NTLM Login"

Authentifiziert einen Benutzer über NTLM anhand von konfiguriertem Benutzernamen, Passwort und Domain. Dieses Modul benötigt kein Authentifizierungs-Modul.

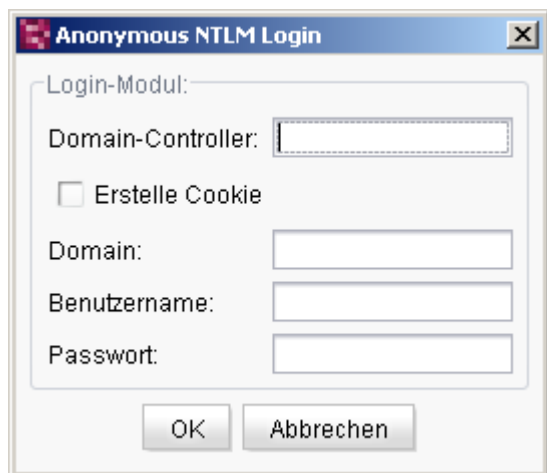


Abbildung 3-2: Login-Modul "Anonymous NTLM Login"

Domain-Controller: Name des zu verwendenden Domain-Controllers (siehe Kapitel 3.1.1 Seite 23).

Erstelle Cookie: Wird die Checkbox aktiviert, wird ein Cookie für die Authentifizierung über SSO erstellt (Konzept siehe Kapitel 1.3.1 Seite 7). Die Konfiguration des Cookies (Domain, Name, Lebenszeit) erfolgt über die Webanwendung (siehe Kapitel 2.3 Seite 17).

Domain: Name der zu verwendenden Domain.

Benutzername: Name des Benutzers für den der Login bzw. die Authentifizierung erfolgen soll.

Passwort: Passwort mit dem der Login des Benutzers bzw. die Authentifizierung erfolgen soll.

Zur NTLM-Authentifizierung siehe auch Kapitel 2.5.1 Seite 21 "NtlmPreAuthenticationFilter".



3.1.4 Login-Modul "Request Parameter Login"

Das Login-Modul "Request Parameter Login" liest den Benutzernamen und das Passwort aus dem HTTP-Request aus und kann beispielsweise in Kombination mit einem Login-Servlet verwendet werden. Im Request werden die Parameter "login" und "password" erwartet.



Dieses Modul benötigt ein Authentifizierungs-Modul, welches die erhaltenen Login-Daten überprüft. Informationen zu Authentifizierungs-Modulen siehe Kapitel 3.2 ab Seite 29.

Parameter: Dieses Modul benötigt keine Konfigurationsparameter.

3.1.5 Login-Modul "Request Header Login"

Das Login-Modul "Request Header Login" liest den Benutzernamen aus dem Request-Header aus. Da nur Benutzernamen von bereits authentifizierten Benutzern im Header erwartet werden, benötigt dieses Modul kein Authentifizierungs-Modul.



Abbildung 3-3: Login-Modul "Request Header Login"

Remote User Header Parameter: Name des Request-Headers, der den Benutzernamen enthält. Der Parameter ist optional, bei Nichtangabe wird der Benutzername aus dem RemoteUser-Attribut des Requests ausgelesen.

Parse DN: Gibt an, ob der gesamte Header-Wert als Benutzername interpretiert werden soll (Wert: true) oder nur das Teilstück zwischen dem erstem ',' und dem nächsten Komma (Wert: false). Dies wird in Fällen benötigt, in denen im Header ein kompletter DN steht, der als ersten Eintrag den Benutzernamen enthält.



Beispiel eines Header-Werts mit komplettem DN:

```
cn=brown,ou=reporting,o=mycompany,c=US
```

3.1.6 Login-Modul "Portlet Login"

Dieses Login-Modul liest den Benutzernamen aus einem angebundenen IBM WebSphere Portal Server aus. Da sich der Benutzer bereits über das Portal authentifiziert hat, benötigt dieses Modul kein Authentifizierungs-Modul.

Parameter: Dieses Modul benötigt keine Konfigurationsparameter.

3.1.7 Login-Modul "SAP Portlet Login"

Dieses Login-Modul liest den Benutzernamen aus einem angebundenen SAP Portal Server aus. Da sich der Benutzer bereits über das Portal authentifiziert hat, benötigt dieses Modul kein Authentifizierungs-Modul.

Parameter: Dieses Modul benötigt keine Konfigurationsparameter.

3.1.8 Login-Modul "FS SSO"

FirstSpirit bietet die Möglichkeit an Single-Sign-On (SSO) für die Startseite, den JavaClient, den WebClient, das Server-Monitoring und die Server- und Projektkonfiguration zu konfigurieren. Dadurch muss sich ein Benutzer nur einmalig anmelden. Die Anmeldeinformation wird nach der Anmeldung von den anderen Anwendungen, soweit konfiguriert, weiterverwendet (Konzept siehe Kapitel 1.3.1 Seite 7). Mit dem Login-Modul "FS SSO" kann diese Anmeldeinformation auch im Modul FirstSpirit DynamicPersonalization genutzt und der Benutzer angemeldet werden. Da sich der Benutzer bereits über den FirstSpirit-Server authentifiziert hat, benötigt dieses Modul kein Authentifizierungs-Modul.

Parameter: Dieses Modul benötigt keine Konfigurationsparameter.



3.2 Authentifizierungs-Module

Wurden durch ein Login-Modul Login-Daten ermittelt, die noch nicht authentifiziert wurden, so wird ein Authentifizierungs-Modul benötigt, das diese Aufgabe übernimmt.

Es stehen folgende Authentifizierungs-Module zur Verfügung:

- JDBC
Authentifizierung gegen eine JDBC-Datenbank
(siehe Kapitel 3.2.1 Seite 30)
- LDAP
Authentifizierung gegen einen LDAP-Server
(vgl. Kapitel 3.2.2 Seite 31).
- ORMapper
Authentifizierung gegen eine Datenquelle aus der FirstSpirit Datenquellen-Verwaltung
(vgl. Kapitel 3.2.3 Seite 35).
- Property File
Authentifizierung gegen eine Property-Datei
(vgl. Kapitel 3.2.4 Seite 36).
- Permission Service
Authentifizierung gegen den FirstSpirit Systemdienst "Permission Service"
(vgl. Kapitel 3.2.5 Seite 37).



3.2.1 Authentifizierungs-Modul „JDBC“

Das Authentifizierungs-Modul "JDBC" dient der Authentifizierung gegenüber den Benutzerkonto-Informationen, die innerhalb einer Datenbank gespeichert und über JDBC erreichbar sind.

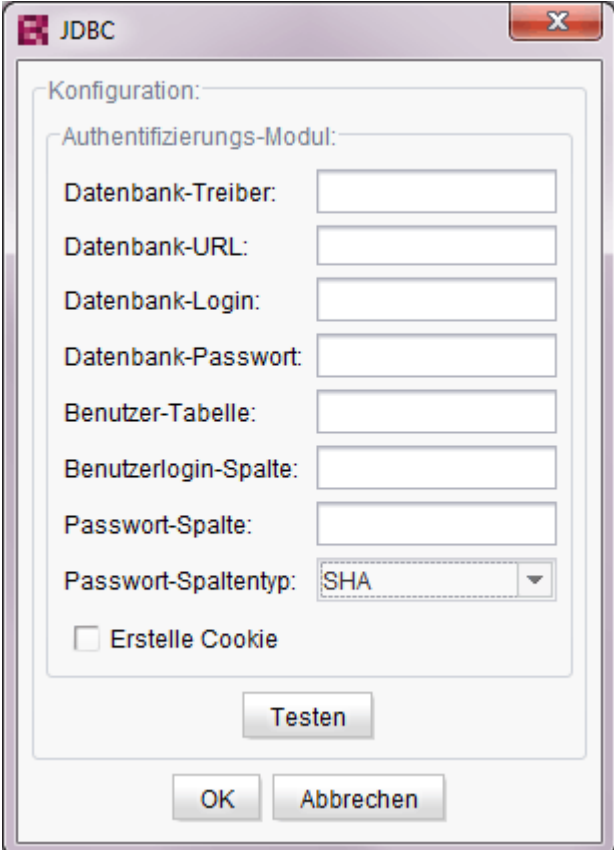


Abbildung 3-4: Authentifizierungs-Modul "JDBC-Konfiguration"

Datenbank-Treiber: Angabe des vollständigen Klassennamens des verwendeten JDBC-Datenbank-Treibers, z.B. `com.mysql.jdbc.Driver`.

Datenbank-URL: Angabe des JDBC-URL zu einem Datenbank-Server und einer dort vorhandenen Datenbank, z.B. `jdbc:mysql://myServer:3306/mydb`.

Datenbank-Login: Gültiger Login-Name eines Datenbank-Users. Mit diesem Account baut der FirstSpirit-Server zur Laufzeit eine Verbindung zur Datenbank auf.

Datenbank-Passwort: Gültiges Passwort zum Login-Name unter „Datenbank-Login“.

Benutzer-Tabelle: Angabe einer Tabelle, die die Benutzerkonto-Informationen für



die Authentifizierung enthält.

Benutzerlogin-Spalte: Angabe eines Tabellenbereichs (Spalte) für die Benutzer-Login-Informationen.

Passwort-Spalte: Angabe eines Tabellenbereichs (Spalte) für die Benutzer-Passwort-Informationen. Abhängig von der Einstellung des Parameters „Passwort-Spaltentyp“ wird ein bestimmtes Encoding für die Inhalte dieses Tabellenbereichs angenommen.

Passwort-Spaltentyp: Die Authentifizierungs-Informationen der „Passwort-Spalte“ können nach bestimmten Algorithmen encodiert (SHA, MD5-Tomcat, MD5-Base64) oder in Klartext (clear) enthalten sein. Diese Algorithmen müssen beim Abgleichen der Authentifizierungsinformationen berücksichtigt werden.

Erstelle Cookie: Wird die Checkbox aktiviert, wird ein Cookie für die Authentifizierung über SSO erstellt (Konzept siehe Kapitel 1.3.1 Seite 7). Die Konfiguration des Cookies (Domain, Name, Lebenszeit) erfolgt über die Webanwendung (siehe Kapitel 2.3 Seite 17).

Testen

Mit einem Klick auf den Button kann die Konfiguration des Authentifizierungs-Moduls getestet werden. Dazu muss der globale Dienst "ConfigTestService" gestartet sein (siehe Kapitel 2.1 Seite 14). Das Starten und Stoppen von Diensten erfolgt entweder über die Server- und Projektkonfiguration (Servereigenschaften / Module) oder über das FirstSpirit Server Monitoring (FirstSpirit / Steuerung / Dienste).



3.2.2 Authentifizierungs-Modul "LDAP"

Das Authentifizierungs-Modul "LDAP" dient der Authentifizierung gegenüber einem LDAP-Server.

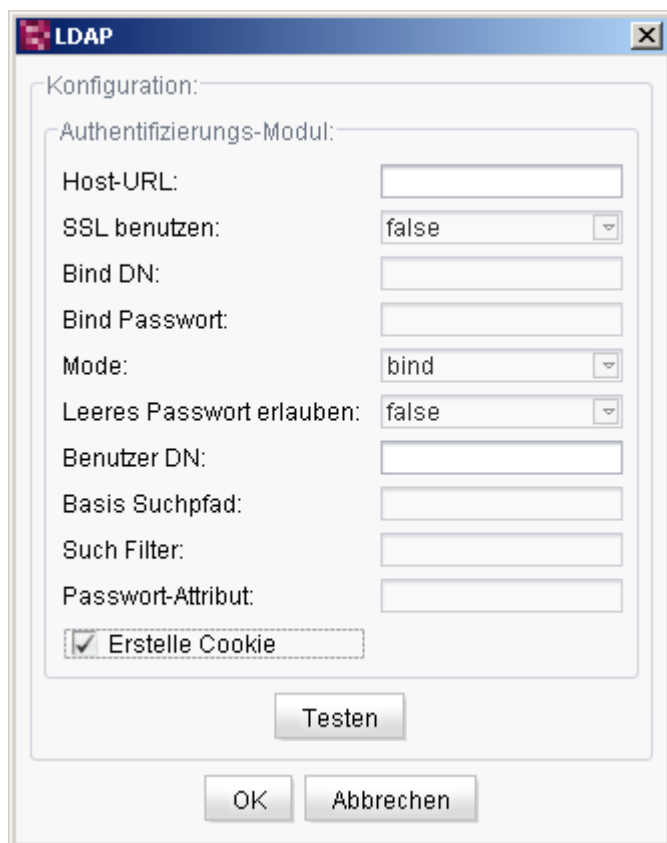


Abbildung 3-5: Authentifizierungs-Modul "LDAP-Konfiguration"

Host-URL: Eine kommaseparierte Liste von LDAP-Servern

SSL benutzen: Gibt an, ob SSL verwendet werden soll (Wert: true) oder nicht. (Wert: false).

Bind DN: DN eines Benutzers, mit dem ein initialer Kontext des LDAP-Servers geholt werden kann. Der Parameter wird nur in Verbindung mit "search_bind" und "search_compare" benötigt (vgl. "Mode").

Bind Passwort: Passwort zu dem unter "Bind DN" angegebenen DN. Der Parameter wird nur in Verbindung mit "search_bind" und "search_compare" benötigt (vgl. "Mode").

Mode: Gibt die Authentifizierungsart an. Mögliche Werte: `bind`, `search_bind` und `search_compare`:



- **Bind:** Mit dem zu einem kompletten DN erweiterten Login-Namen und -Passwort wird eine Authentifizierung (sogenannter "Bind") gegenüber dem LDAP-Server durchgeführt. Name und Passwort werden an den LDAP-Server geschickt. Dazu muss der "Distinguished Name" (DN), das heißt der eindeutige Schlüssel zur Identifizierung des Benutzers innerhalb des LDAP-Servers bekannt sein. Falls der DN existiert, wird das übergebene Passwort mit Hilfe der "Bind"-Operation geprüft.
- **Search & Bind:** Falls der "Distinguished Name" (DN) eines Benutzers nicht bekannt ist, kann man ihn innerhalb eines Teilbaums des LDAP-Servers suchen. Dazu muss ein Suchfilter und ein Startknoten definiert werden (siehe Beschreibung zu "Basis-Suchpfad" und "Such Filter"). Wird ein Startknoten gefunden so wird ein "Bind" (gegenüber dem LDAP-Server) ausgeführt. (siehe LDAP Bind).
- **Search & Compare:** Analog zum "Search & Bind" wird bei diesem Mode der komplette DN des Benutzers anhand einer Suche auf dem LDAP-Baum ermittelt. Allerdings wird, nachdem ein passender Knoten gefunden worden ist, keine "Bind"-Operation durchgeführt. Stattdessen wird das eingegebene Passwort mit einem beliebigen (zuvor definierten) LDAP-Attribut verglichen (siehe Beschreibung zu "Passwort-Attribut").

Leeres Passwort erlauben: Gibt an, ob leere Passwörter verwendet werden dürfen oder nicht.

Benutzer DN: Eine durch "#" separierte Liste von DNs, die jeweils die Zeichenfolge \$USER_LOGIN\$ beinhalten müssen. Die Zeichenfolge wird dann automatisch durch den jeweiligen Benutzernamen ersetzt. Der Parameter ist nicht optional, wenn für den Parameter mode der Wert "bind" vergeben wurde.

Beispiel:

```
cn=$USER_LOGIN$,cn=Recipients,ou=E-SPIRIT,o=e-Spirit
```

oder

```
cn=$USER_LOGIN$,cn=Recipients,ou=E-SPIRIT,o=e-Spirit#cn=$USER_LOGIN$,cn=others,ou=E-SPIRIT,o=e-Spirit
```

Basis Suchpfad: Gibt den DN des Startknotens an, von dem aus im LDAP-Baum gesucht werden soll. Der Parameter wird nur in Verbindung mit "search_bind" und "search_compare" benötigt.



Beispiel:

```
cn=Recipients,ou=E-SPIRIT,o=e-Spirit
```

Such Filter: Ein Filter, der für die Suche im LDAP-Baum verwendet werden soll. Innerhalb des Filters kann durch die Zeichenkette \$USER_LOGIN\$ der jeweilige Benutzername eingefügt werden. Der Parameter wird nur in Verbindung mit "search_bind" und "search_compare" benötigt.

Beispiel:

```
(sn=$USER_LOGIN$)
```

Passwort-Attribut: Ein Attribut, dessen Wert mit dem eingegebenen Passwort verglichen wird. Der Parameter wird nur in Verbindung mit "search_compare" benötigt.

Erstelle Cookie: Wird die Checkbox aktiviert, wird ein Cookie für die Authentifizierung über SSO erstellt (Konzept siehe Kapitel 1.3.1 Seite 7). Die Konfiguration des Cookies (Domain, Name, Lebenszeit) erfolgt über die Webanwendung (siehe Kapitel 2.3 Seite 17).

Testen

Mit einem Klick auf den Button kann die Konfiguration des Authentifizierungs-Moduls getestet werden. Dazu muss der globale Dienst "ConfigTestService" gestartet sein (siehe Kapitel 2.1 Seite 14). Das Starten und Stoppen von Diensten erfolgt entweder über die Server- und Projektkonfiguration (Servereigenschaften / Module) oder über das FirstSpirit Server Monitoring (FirstSpirit / Steuerung / Dienste).



3.2.3 Authentifizierungs-Modul "ORMapper"

Mit dem Authentifizierungs-Modul "ORMapper" kann eine Authentifizierung gegen eine Tabelle aus einer Datenbank von FirstSpirit durchgeführt werden.



Abbildung 3-6: Authentifizierungs-Modul "ORMapper"

OR-Schema: Name des Konfigurationsschemas, das verwendet werden soll.

Benutzer-Tabelle: Name der Tabelle mit den Benutzerinformationen.

Benutzerlogin-Spalte: Name der Tabellenspalte, die den Benutzernamen enthält.

Passwort-Spalte: Name der Tabellenspalte, die das Benutzerpasswort enthält.

Passwort-Spalentyp: Verschlüsselungsart des Passworts.

Mögliche Werte: `clear`, `SHA`, `MD5-Tomcat` und `MD5-Base64`

Erstelle Cookie: Wird die Checkbox aktiviert, wird ein Cookie für die Authentifizierung über SSO erstellt (Konzept siehe Kapitel 1.3.1 Seite 7). Die Konfiguration des Cookies (Domain, Name, Lebenszeit) erfolgt über die Webanwendung (siehe Kapitel 2.3 Seite 17).

Nähere Informationen zur Konfiguration des Parameters "OR-Schema" können der Modul-Dokumentation "FirstSpirit INTEGRATION" entnommen werden.



3.2.4 Authentifizierungs-Modul "Property File"

Dieses Modul dient dazu, eine Authentifizierung gegen eine Property-Datei durchzuführen.

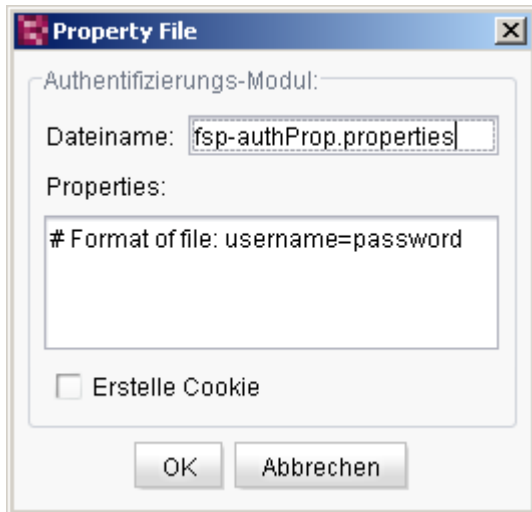


Abbildung 3-7: Authentifizierungs-Modul "Property File"

Dateiname: Name unter dem die Datei gespeichert werden soll.

Properties: Benutzername-/Passwortpaare. Jede Zeile entspricht einem Benutzer. Der Benutzername wird vom Passwort mit "=" getrennt.

Erstelle Cookie: Wird die Checkbox aktiviert, wird ein Cookie für die Authentifizierung über SSO erstellt (Konzept siehe Kapitel 1.3.1 Seite 7). Die Konfiguration des Cookies (Domain, Name, Lebenszeit) erfolgt über die Webanwendung (siehe Kapitel 2.3 Seite 17).



3.2.5 Authentifizierungs-Modul "Permission Service"

Mit dem Authentifizierungs-Modul "Permission Service" ist es möglich eine Authentifizierung gegen den Systemdienst "Permission Service" vorzunehmen (Konzept siehe Kapitel 1.3.2 Seite 8). Als Grundlage der Authentifizierung dient der Inhalt der ausgewählten User-XML-Datei (Angabe "NAME.users" in der Servicekonfigurationsdatei).

Weitere Informationen können der FirstSpirit "Dokumentation für Administratoren" entnommen werden.



Abbildung 3-8: Authentifizierungs-Modul "Permission Service"

User-Datei: Dateiname der ausgewählten User-Datei.

Erstelle Cookie: Wird die Checkbox aktiviert, wird ein Cookie für die Authentifizierung über SSO erstellt (Konzept siehe Kapitel 1.3.1 Seite 7). Die Konfiguration des Cookies (Domain, Name, Lebenszeit) erfolgt über die Webanwendung (siehe Kapitel 2.3 Seite 17).



3.3 Gruppen-Module

Die Gruppen-Module dienen zum Auslesen von Gruppeninformationen für den angemeldeten Benutzer. Der Einsatz der optionalen Gruppen-Module ist nur dann nötig, wenn zwischen Gruppen unterschieden werden soll. Dies ist der Fall, wenn die Anzeige bestimmter Bereiche einer Seite bestimmten Gruppen vorbehalten ist. In diesem Fall können die Gruppenzugehörigkeit des angemeldeten Benutzers für die Steuerung der Anzeige herangezogen werden.

Folgende Gruppen-Module stehen zur Verfügung:

- JDBC
Gruppeninformationen aus einer Datenbank beziehen, die über JDBC erreichbar ist (siehe Kapitel 3.3.1 Seite 39)
- LDAP Group
Gruppeninformation von einem LDAP-Server beziehen (Gruppenattribute sind dem Benutzer zugeordnet)
(vgl. Kapitel 3.3.2 Seite 39).
- LDAP Group Iterate
Gruppeninformation von einem LDAP-Server beziehen (Gruppenattribute sind dem Gruppenobjekt zugeordnet)
(vgl. Kapitel 3.3.3 Seite 41).
- ORMapper
Gruppeninformationen von einer Datenquelle aus der FirstSpirit Datenquellen-Verwaltung beziehen
(vgl. Kapitel 3.3.4 Seite 43).
- Portlet
Gruppeninformationen von einer IBM WebSphere Portal Server beziehen
(vgl. Kapitel 3.3.5 Seite 43).
- SAP Portlet
Gruppeninformationen von einem SAP Portal Server beziehen
(vgl. Kapitel 3.3.6 Seite 44).
- FS SSO Groups
Gruppeninformationen über FirstSpirit Single Sign-On beziehen
(vgl. Kapitel 3.3.7 Seite 44).
- Permission Service
Gruppeninformationen über den FirstSpirit Systemdienst "Permission Service" beziehen
(vgl. Kapitel 3.3.8 Seite 44).



3.3.1 Gruppen-Modul „JDBC“

Das Gruppen-Modul „JDBC“ dient dazu, die Gruppeninformationen des Benutzers aus der Tabelle einer Datenbank zu beziehen, die über JDBC erreichbar ist.

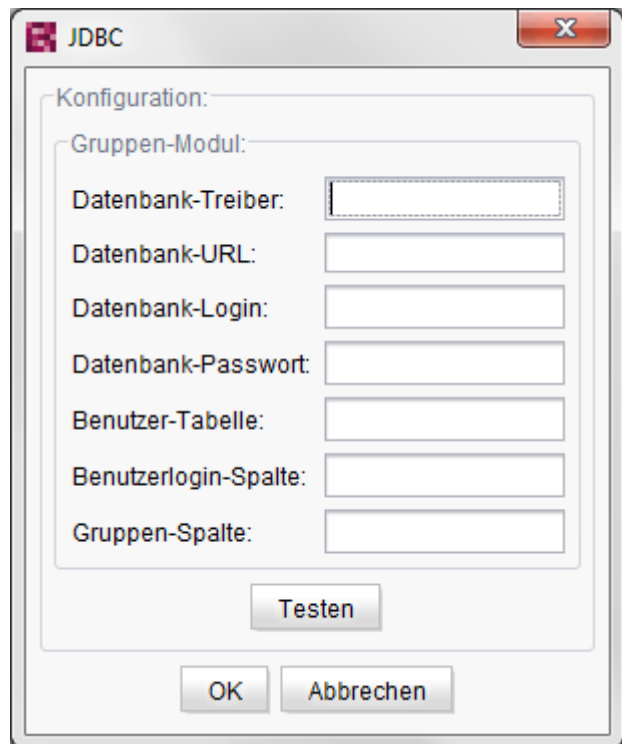


Abbildung 3-9: Gruppen-Modul JDBC

Datenbank-Treiber: Angabe des vollständigen Klassennamens des verwendeten JDBC-Datenbank-Treibers, z.B. `com.mysql.jdbc.Driver`.

Datenbank-URL: Angabe des JDBC-URL zu einem Datenbank-Server und einer dort vorhandenen Datenbank, z.B. `jdbc:mysql://myServer:3306/mydb`.

Datenbank-Login: Gültiger Login-Name eines Datenbank-Users. Mit diesem Account baut der FirstSpirit-Server zur Laufzeit eine Verbindung zur Datenbank auf.

Datenbank-Passwort: Gültiges Passwort zum Login-Name unter „Datenbank-Login“.

Benutzer-Tabelle: Angabe einer Tabelle, die die Benutzerkonto-Informationen enthält.

Benutzerlogin-Spalte: Angabe eines Tabellenbereichs (Spalte) für die Benutzer-Login-Informationen.

Gruppen-Spalte: Angabe eines Tabellenbereichs (Spalte) für die Benutzer-



Gruppeninformationen.

Testen

Mit einem Klick auf den Button kann die Konfiguration des Moduls getestet werden. Dazu muss der globale Dienst "ConfigTestService" gestartet sein (siehe Kapitel 2.1 Seite 14). Das Starten und Stoppen von Diensten erfolgt entweder über die Server- und Projektkonfiguration (Servereigenschaften / Module) oder über das FirstSpirit Server Monitoring (FirstSpirit / Steuerung / Dienste).

3.3.2 Gruppen-Modul "LDAP Group"

Das Gruppen-Modul "LDAP Group" dient dazu, die Gruppeninformationen des Benutzers über einen LDAP-Server zu beziehen. Hierbei sollen die Informationen im LDAP so strukturiert sein, dass zu jedem Benutzer eine Liste von Gruppenattributen vorhanden ist.



Abbildung 3-10: Gruppen-Modul "LDAP Group"

Host-URL: Eine kommaseparierte Liste von LDAP-Servern.

SSL benutzen: Gibt an, ob SSL verwendet werden soll (Wert: true) oder nicht. (Wert: false).

Bind DN: DN eines Benutzers, mit dem ein initialer Kontext des LDAP-Servers geholt werden kann.



Bind Passwort: Passwort zu dem unter "Bind DN" angegebenen DN.

Benutzer DN: Eine durch "#" separierte Liste von DN's, die jeweils die Zeichenfolge \$USER_LOGIN\$ beinhalten müssen. Die Zeichenfolge wird dann automatisch durch den jeweiligen Benutzernamen ersetzt.

Beispiel:

```
cn=$USER_LOGIN$,cn=Recipients,ou=E-SPIRIT,o=e-Spirit
```

Gruppen Attribut: Attribut, in dem die Gruppen enthalten sind, zu denen der Benutzer gehört.

Testen

Mit einem Klick auf den Button kann die Konfiguration des Moduls getestet werden. Dazu muss der globale Dienst "ConfigTestService" gestartet sein (siehe Kapitel 2.1 Seite 14). Das Starten und Stoppen von Diensten erfolgt entweder über die Server- und Projektkonfiguration (Servereigenschaften / Module) oder über das FirstSpirit Server Monitoring (FirstSpirit / Steuerung / Dienste).

3.3.3 Gruppen-Modul "LDAP Group Iterate"

Das Gruppen-Modul "LDAP Group Iterate" dient dazu, Gruppeninformationen des Benutzers über einen LDAP-Server zu beziehen. Das Gruppen-Modul wird eingesetzt, wenn LDAP folgendermaßen konfiguriert wurde:

- die Benutzereinträge enthalten *keine Informationen* über die Gruppen, zu denen sie gehören
- nur die Gruppenobjekte selbst enthalten eine Liste der zugehörigen Benutzer

In diesem Fall muss über alle Gruppen iteriert und die Gruppen extrahiert werden, die den aktuellen Benutzer als Mitglied enthalten.



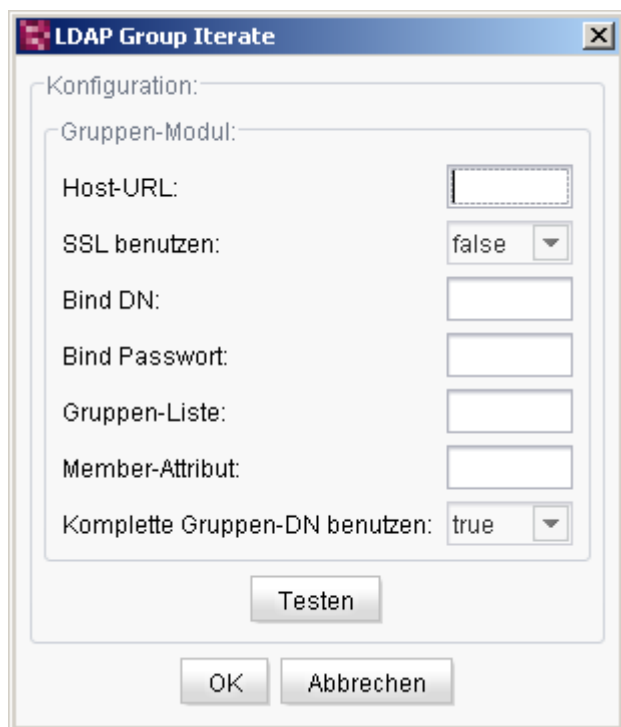


Abbildung 3-11: Gruppen-Modul "LDAP Group Iterate"

Host-URL: Eine kommaseparierte Liste von LDAP-Servern.

SSL benutzen: Gibt an, ob SSL verwendet werden soll (Wert: true) oder nicht. (Wert: false).

Bind DN: DN eines Benutzers, mit dem ein initialer Kontext des LDAP-Servers geholt werden kann.

Bind Passwort: Passwort zum unter "Bind DN" angegebenen DN.

Gruppen-Liste: Eine durch ‚#‘ separierte Liste von DN, die die Gruppen darstellen, in denen die Benutzer als Mitglieder aufgeführt sein können.

Member-Attribut: Attribut, in dem die Benutzer enthalten sind, die Mitglieder der Gruppe sind.

Komplette Gruppen-DN benutzen: Gibt an, ob der komplette DN der Gruppe als Gruppenname verwendet werden soll (Wert: true) oder nicht (Wert: false). Ist dies nicht der Fall, wird nur der letzte Wert des DN als Gruppenname verwendet.

Testen

Mit einem Klick auf den Button kann die Konfiguration des Moduls getestet werden. Dazu muss der globale Dienst "ConfigTestService" gestartet sein (siehe Kapitel 2.1 Seite 14). Das Starten und Stoppen von Diensten erfolgt entweder



über die Server- und Projektkonfiguration (Servereigenschaften / Module) oder über das FirstSpirit Server Monitoring (FirstSpirit / Steuerung / Dienste).

3.3.4 Gruppen-Modul "ORMapper"

Das Gruppen-Modul "ORMapper" dient dazu, die Gruppen eines Benutzers aus einer Tabelle der Datenbank von FirstSpirit auszulesen.

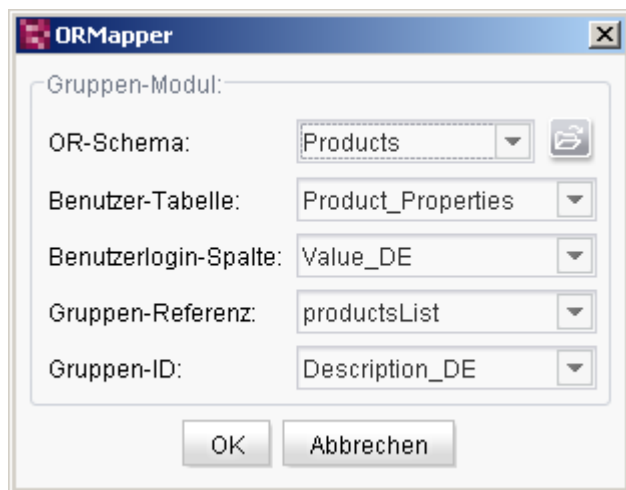


Abbildung 3-12: Gruppen-Modul "ORMapper"

OR-Schema: Name des Konfigurationsschemas, welches verwendet werden soll.

Benutzer-Tabelle: Name der Tabelle mit den Benutzerinformationen.

Benutzerlogin-Spalte: Name der Tabellenspalte, die den Benutzernamen enthält.

Gruppen-Referenz: Die Referenz zwischen Benutzertabelle und Gruppentabelle.

Gruppen-ID: Tabellenspalte der Gruppentabelle, die den Gruppennamen enthält

Nähere Informationen zur Konfiguration des Parameters "OR-Schema" können der Modul-Dokumentation "FirstSpirit INTEGRATION" entnommen werden.

3.3.5 Gruppen-Modul "Portlet"

Dieses Gruppen-Modul liest die Gruppeninformationen aus einem angebundenen IBM WebSphere Portal Server aus.

Parameter: Dieses Gruppen-Modul benötigt keine Konfigurationsparameter.



3.3.6 Gruppen-Modul "SAP Portlet"

Dieses Gruppen-Modul liest die Gruppeninformationen aus einem angebenen SAP Portal Server aus. Dazu muss das Modul "FirstSpirit PORTAL" (vgl. die Dokumentation "FirstSpirit PORTAL SAP EP BP") installiert sein.

Parameter: Dieses Gruppen-Modul benötigt keine Konfigurationsparameter.

3.3.7 Gruppen-Modul "FS SSO Groups"

Mit dem Gruppen-Modul "FS SSO Groups" können die Gruppeninformationen des (am FirstSpirit-Server) angemeldeten Benutzers ausgelesen werden.

Für die Konfiguration von FirstSpirit Single Sign-On siehe FirstSpirit "Dokumentation für Administratoren".

Parameter: Dieses Gruppen-Modul benötigt keine Konfigurationsparameter.

3.3.8 Gruppen-Modul "Permission Service"

Mit dem Gruppen-Modul "Permission Service" ist es möglich, die Gruppeninformationen zu einem Benutzer über den Systemdienst "Permission Service" zu beziehen (Konzept siehe Kapitel 1.3.2 Seite 8). Als Grundlage der Gruppeninformationen eines Benutzers dient der Inhalt der ausgewählten User-XML-Datei (Angabe "NAME.users" in der Servicekonfigurationsdatei).

Weitere Informationen können der FirstSpirit "Dokumentation für Administratoren" entnommen werden.

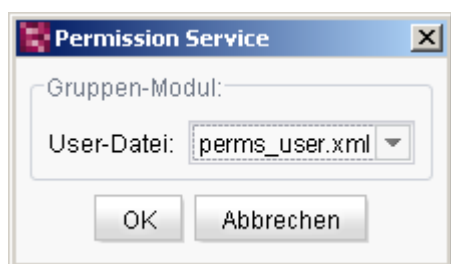


Abbildung 3-13: Gruppen-Modul "Permission Service"

User-Datei: Dateiname der ausgewählten User-Datei.



3.4 Attribut-Module

Die Attribut-Module dienen zum Auslesen von Attributen für den angemeldeten Benutzer. Die Angabe eines Attribut-Moduls ist optional und nur nötig, falls weitere Attribute für den Benutzer ausgelesen werden sollen.

Folgende Attribut-Module stehen zur Verfügung:

- LDAP
Attribute von einem LDAP-Server beziehen
(vgl. Kapitel 3.4.1 Seite 45).
- ORMapper
Attribute von einer Datenquelle aus der FirstSpirit Datenquellen-Verwaltung beziehen
(vgl. Kapitel 3.4.2 Seite 48).

3.4.1 Attribut-Modul "LDAP"

Das Attribut-Modul "LDAP" dient dazu, Attribute eines Benutzers von einem LDAP-Server zu beziehen.

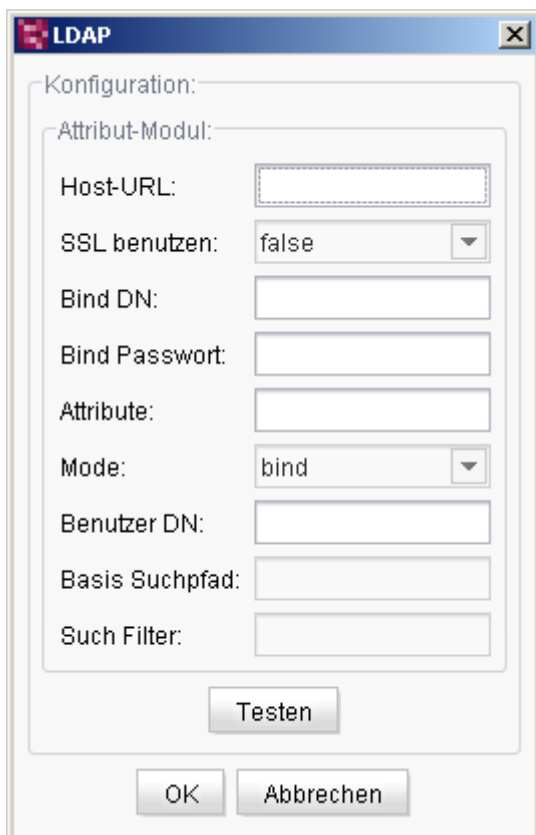


Abbildung 3-14: Attribut-Modul "LDAP"



Host-URL: Eine kommaseparierte Liste von LDAP-Servern

SSL benutzen: Gibt an, ob SSL verwendet werden soll (Wert: true) oder nicht. (Wert: false).

Bind DN: DN eines Benutzers, mit dem ein initialer Kontext des LDAP-Servers geholt werden kann. Der Parameter wird nur in Verbindung mit "search_bind" und "search_compare" benötigt (vgl. "Mode").

Bind Passwort: Passwort zu dem unter "Bind DN" angegebenen DN. Der Parameter wird nur in Verbindung mit "search_bind" und "search_compare" benötigt (vgl. "Mode").

Attribute: Kommaseparierte Liste von Attributen, die ausgelesen werden sollen.

Mode: Gibt die Methode an, die verwendet werden soll, um zu Benutzerknoten innerhalb des LDAP-Baumes zu gelangen. Mögliche Werte: `bind` und `search_compare`:

- **Bind:** Mit dem zu einem kompletten DN erweiterten Login-Namen und -Passwort wird eine Authentifizierung (sogenannter "Bind") gegenüber dem LDAP-Server durchgeführt. Name und Passwort werden an den LDAP-Server geschickt. Dazu muss der "Distinguished Name" (DN), das heißt der eindeutige Schlüssel zur Identifizierung des Benutzers innerhalb des LDAP-Servers bekannt sein. Falls der DN existiert, wird das übergebene Passwort mit Hilfe der "Bind"-Operation geprüft.
- **Search & Compare:** Analog zum "Search & Bind" wird bei diesem Mode der komplette DN des Benutzers anhand einer Suche auf dem LDAP-Baum ermittelt. Allerdings wird, nachdem ein passender Knoten gefunden worden ist, keine "Bind"-Operation durchgeführt..

Benutzer DN: Eine durch "#" separierte Liste von DN's, die jeweils die Zeichenfolge \$USER_LOGIN\$ beinhalten müssen. Die Zeichenfolge wird dann automatisch durch den jeweiligen Benutzernamen ersetzt. Der Parameter ist nicht optional, wenn für den Parameter mode der Wert "bind" vergeben wurde.

Beispiel:

```
cn=$USER_LOGIN$,cn=Recipients,ou=E-SPIRIT,o=e-Spirit
```

oder

```
cn=$USER_LOGIN$,cn=Recipients,ou=E-SPIRIT,o=e-Spirit#cn=$USER_LOGIN$,cn=others,ou=E-SPIRIT,o=e-Spirit
```

Basis Suchpfad: Gibt den DN des Startknotens an, von dem aus im LDAP-Baum gesucht werden soll. Der Parameter wird nur in Verbindung mit "search_compare"



benötigt.

Beispiel:

```
cn=Recipients,ou=E-SPIRIT,o=e-Spirit
```

Such Filter: Ein Filter, der für die Suche im LDAP-Baum verwendet werden soll. Innerhalb des Filters kann durch die Zeichenkette \$USER_LOGIN\$ der jeweilige Benutzername eingefügt werden. Der Parameter wird nur in Verbindung mit "search_compare" benötigt.

Beispiel:

```
(sn=$USER_LOGIN$)
```

Testen

Mit einem Klick auf den Button kann die Konfiguration des Authentifizierungs-Moduls getestet werden. Dazu muss der globale Dienst "ConfigTestService" gestartet sein (siehe Kapitel 2.1 Seite 14). Das Starten und Stoppen von Diensten erfolgt entweder über die Server- und Projektkonfiguration (Servereigenschaften / Module) oder über das FirstSpirit Server Monitoring (FirstSpirit / Steuerung / Dienste).



3.4.2 Attribut-Modul "ORMapper"

Das Attribut-Modul "ORMapper" dient dazu, Attribute eines Benutzers aus einer Tabelle aus der Datenbank von FirstSpirit auszulesen.



Abbildung 3-15: Attribut-Modul "ORMapper"

OR-Schema: Name des Konfigurationsschemas, welches verwendet werden soll.

Benutzer-Tabelle: Name der Tabelle mit den Benutzerinformationen.

Benutzerlogin-Spalte: Name der Tabellenspalte, die den Benutzernamen enthält.

Benutzer-Attribute: Liste von Attributen, die ausgelesen werden sollen.

Nähere Informationen zur Konfiguration des Parameters "OR-Schema" können der Modul-Dokumentation "FirstSpirit INTEGRATION" entnommen werden.



4 Servlets

Das Modul FirstSpirit DynamicPersonalization stellt zwei Servlets zur Verfügung.

Mit dem ersten Servlet ist es möglich, sich mit Hilfe von Request-Parametern am Modul anzumelden. Sinnvoll ist der Einsatz des Servlets, wenn das Login-Modul "Request Parameter Login" verwendet wird (siehe Kapitel 3.1.4 Seite 27).

Das zweite Servlet bietet die Möglichkeit, den aktuell angemeldeten Benutzer abzumelden.

- **"Login-Servlet"**: Benutzer mit Request-Parametern anmelden (vgl. Kapitel 4.1 Seite 49)
- **"Logout-Servlet"**: Benutzer abmelden (vgl. Kapitel 4.2 Seite 51)



Um Open-Redirect-Angriffe zu unterbinden, sind in Projekten, die das Modul FirstSpirit™ DynamicPersonalization verwenden, externe Redirects verboten. D. h. es sind nur relative URLs möglich (z. B.

```
start.jsp  
../bereich/index.html  
../bereich/search.jsp)
```

Ausnahme: Redirects auf denselben Host oder einen Host in der gleichen Domain sind zulässig. Soll zu einer externen URL weitergeleitet werden, muss dafür eine Weiterleitungsseite erstellt werden.

4.1 "Login-Servlet"

Das "Login-Servlet" ermöglicht das Anmelden eines Benutzers am Modul. Die Eingabe des Benutzernamens und -passworts erfolgt üblicherweise in einem HTML-Formular. Die vom Formular übermittelten Request-Parameter nutzt das Servlet für die Anmeldung des Benutzers.

Der Aufruf des Login-Servlets erfolgt mit:

```
<%= request.getContextPath() %>/do.login
```



Request-Parameter:

Parameter	Erwarteter Wert
login	Benutzername
password	Benutzerpasswort
loginPackage	Name des Login-Packages, welches für die Anmeldung verwendet werden soll
login_ok_url	<p>URL (bei erfolgreicher Anmeldung)</p> <p><u>Hinweis:</u> Um Open-Redirect-Angriffe zu unterbinden, sind in Projekten, die das Modul FirstSpirit™ DynamicPersonalization verwenden, externe Redirects verboten. D. h. es sind nur relative URLs möglich (z. B. start.jsp ../bereich/index.html ../bereich/search.jsp)</p> <p>Ausnahme: Redirects auf denselben Host oder einen Host in der gleichen Domain sind zulässig. Soll zu einer externen URL weitergeleitet werden, muss dafür eine Weiterleitungsseite erstellt werden.</p>
wrong_login_url	<p>URL (bei fehlerhafter Anmeldung)</p> <p>Externe Redirects sind verboten (siehe Hinweis oben).</p>

Einfaches Beispiel:

```
<form
  method="POST"
  action="<%= request.getContextPath() %>/do.login">

  Benutzername:
  <input
    type="text"
    name="login"
    value="" />

  Passwort:
  <input
    type="password"
    name="password"
    value="" />

  <input
    type="hidden"
    name="login_ok_url"
    value="<%= CMS_REF(ptLoginOk, abs:1) %>" />
  <input
    type="hidden"
    name="wrong_login_url"
    value="<%= CMS_REF(ptLoginFailed, abs:1) %>" />
```



```
<input
  type="submit"
  value="Login" />
</form>
```

4.2 "Logout-Servlet"

Mit dem "Logout-Servlet" wird der momentan angemeldete Benutzer abgemeldet. Als einziger Parameter wird "redirect_url" benötigt, der angibt, welcher URL nach der Abmeldung angezeigt werden soll.

Der Aufruf des Login-Servlets erfolgt mit:

```
"<%= request.getContextPath() %>/do.logout".
```

Einfaches Beispiel:

```
<form
  method="POST"
  action="<%= request.getContextPath() %>/do.logout">

  <input
    type="hidden"
    name="redirect_url"
    value="/logout.jsp" />

  <input
    type="submit"
    value="Logout" />

</form>
```



Um Open-Redirect-Angriffe zu unterbinden, sind in Projekten, die das Modul FirstSpirit™ DynamicPersonalization verwenden, externe Redirects verboten. D. h. es sind nur relative URLs möglich (z. B.

```
start.jsp
../bereich/index.html
../bereich/search.jsp)
```

Ausnahme: Redirects auf denselben Host oder einen Host in der gleichen Domain sind zulässig. Soll zu einer externen URL weitergeleitet werden, muss dafür eine Weiterleitungsseite erstellt werden.



5 Tags

Mit den FirstSpirit DynamicPersonalization-Tags ist es möglich, Inhalte benutzerspezifisch anzuzeigen. Die Tag-Library umfasst Tags für einen automatischen Login, das Ausblenden von geschützten Inhalten und das Anzeigen von Benutzerinformationen.

In der Regel enthält eine Seite immer das Tag `<fsp:loginRequired>` oder das Tag `<fsp:authorize>`. Das Tag `<fsp:loginRequired>` leitet zu einer Login-Seite um, falls der aktuelle Benutzer noch nicht angemeldet ist. Dort könnte sich der Benutzer über ein Formular anmelden und würde anschließend zur vorher angesteuerten Seite zurückkehren. Das Tag `<fsp:authorize>` hingegen nimmt eine automatische Authentifizierung im Hintergrund vor (eine entsprechende Auswahl der FirstSpirit DynamicPersonalization-Module vorausgesetzt) und zeigt anschließend den Inhalt der Seite an.

Die FirstSpirit-Personalisation-Taglib stellt eine Vielzahl von Tags zur Verfügung:

- `<fsp:loginRequired>`
Anmeldung erzwingen
(siehe Kapitel 5.2 Seite 53).
- `<fsp:authorize>`
Automatische Authentifizierung
(siehe Kapitel 5.3 Seite 54).
- `<fsp:userInfo>`
Ausgabe des Benutzernames
(siehe Kapitel 5.4 Seite 55).
- `<fsp:userGroups>`
Ausgabe von Benutzergruppen
(siehe Kapitel 5.5 Seite 55).
- `<fsp:userAttributes>`
Ausgabe von Benutzerattributen
(siehe Kapitel 5.6 Seite 56).
- `<fsp:isAuthorized>`
Inhaltsanzeige, wenn der Benutzer autorisiert ist
(siehe Kapitel 5.7 Seite 57).
- `<fsp:isNotAuthorized>`
Inhaltsanzeige, wenn der Benutzer nicht autorisiert ist
(siehe Kapitel 5.8 Seite 61).



- `<fsp:logout>`
Benutzer abmelden
(siehe Kapitel 5.9 Seite 61).

5.1 Präfix

Um FirstSpirit DynamicPersonalization verwenden zu können, müssen in den JSP-Seiten die entsprechenden Tag-Library angegeben werden. Diese Dokumentation verwendet das Präfix "fsp" für FirstSpirit-Personalisation-Tags.

Beispiel für die Einbindung in JSP-Seiten:

```
<%@ page language="java" contentType="text/html; charset=UTF-8" %>  
<%@ taglib uri="FIRSTpersonalisation" prefix="fsp" %>
```

Zu beachten ist hierbei, dass der URI für FirstSpirit immer "FIRSTpersonalisation" lautet.



*Wird das Präfix "fsp" auf einen anderen Wert geändert, so ist dieses Präfix für die einzelnen Tags zu verwenden, d.h. "**<myPrefix:ref>**" anstelle von "**<fsp:ref>**".*

5.2 <fsp:loginRequired>

Mit dem Tag `<fsp:loginRequired>` wird der Inhalt der Seite nur dann angezeigt, wenn ein angemeldeter Benutzer ermittelt wurde. Ist der Benutzer nicht angemeldet, wird er an die im Attribut "loginUrl" angegebene Adresse weitergeleitet, um sich dort anzumelden. Für eine funktionierende Weiterleitung ist es dabei wichtig, dass sich das Tag `<fsp:loginRequired>` an oberster Stelle in der Seite befindet (vor jeder Art von HTML oder Kommentaren).



Das Tag `<fsp:loginRequired>` sollte immer nach dem Tag `<fsp:authorize>` verwendet werden. Dies ist insbesondere wichtig, wenn Login-Module verwendet werden, die Single Sign-On verwenden (wie z. B. "FS SSO").



Attribute:

Attribut	Bedeutung	Pflichtparameter
loginUrl	<p>URL der Anmeldeseite, zu der weitergeleitet werden soll, falls der Benutzer noch nicht angemeldet ist.</p> <p>Nach der erfolgreichen Anmeldung über das Login-Servlet, wird der Benutzer zur ursprünglichen Seite weitergeleitet.</p>	Ja

Beispiel:

```
<fsp:loginRequired loginUrl="../../login.jsp" />
```

5.3 <fsp:authorize>

Das Tag <fsp:authorize> führt eine automatische Authentifizierung des Benutzers durch. Welche Methoden dabei verwendet werden, wird durch die konfigurierten Login-Packages festgelegt. Die Konfiguration der Login-Packages ist in den Kapiteln 2.4 (ab Seite 20) und 3 (ab Seite 22) beschrieben.

Attribute:

Attribut	Bedeutung	Pflichtparameter
force	<p>Wird für das Attribut "force" der Wert "false" verwendet, so wird keine Authentifizierung durchgeführt, falls bereits ein Benutzer angemeldet ist. Bei der Angabe des Wertes "true", wird auf jeden Fall eine erneute Authentifizierung durchgeführt, auch wenn bereits ein Benutzer angemeldet ist.</p> <p>Wird das Attribut nicht angegeben, wird es automatisch mit dem Standardwert "false" ausgewertet.</p>	Nein

Beispiel:

```
<fsp:authorize force="false" />
```



5.4 <fsp:userInfo>

Das Tag <fsp:userInfo> liefert in der Variable "login" den Benutzernamen des aktuell angemeldeten Benutzers zurück. Sollte kein Benutzer angemeldet sein, so enthält die Variable einen Leerstring. Das Tag verfügt über keinerlei Parameter.

Variablen:

Variable	Bedeutung	Rückgabedatentyp
login	Liefert den Benutzernamen des aktuell angemeldeten Benutzers oder einen Leerstring zurück.	java.lang.String

Beispiel:

```
<fsp:userInfo>
  Benutzername: <%= login %>
</fsp:userInfo>
```

5.5 <fsp:userGroups>

Das Tag <fsp:userGroups> liefert in der Variable "groupname" die Gruppennamen der Gruppen zurück, zu denen der aktuell angemeldeten Benutzer gehört. Der Inhalt des Tags wird so oft aufgerufen, wie es Gruppen zu dem angemeldeten Benutzer gibt. Bei jedem Durchlauf kann mit der Variable "groupname" genau ein Gruppenname ausgegeben werden.

Variablen:

Variable	Bedeutung	Rückgabedatentyp
groupname	Liefert je Durchlauf einen Gruppennamen aller Gruppen des aktuell angemeldeten Benutzers zurück.	java.lang.String

Beispiel:

```
<fsp:userGroups>
  Gruppennamen: <%= groupname %>
</fsp:userGroups>
```



5.6 <fsp:userAttributes>

Das Tag <fsp:userAttributes> liefert in den Variablen "attributename" und "attributevalue" Namen und Wert der Benutzer-Attribute zurück. Wird das Tag-Attribut "attributes" nicht angegeben, so werden alle Attribute des Benutzers durchlaufen. Wird mit dem Tag-Attribut "attributes" eine kommaseparierte Liste von Attributnamen angegeben, so werden vom Tag nur die angegebenen Attribute durchlaufen.

Attribute:

Attribut	Bedeutung	Pflichtparameter
attributes	Kommaseparierte Liste von Attributen, die vom Tag durchlaufen werden sollen. Wird das Attribut "attributes" nicht angegeben, so werden alle Benutzer-Attribute vom Tag durchlaufen.	Nein

Variablen:

Variable	Bedeutung	Rückgabedatentyp
attributename	Name des Attributs	java.lang.String
attributevalue	Wert des Attributs	java.lang.String

Beispiel:

```
<fsp:userAttributes attributes="mail,phone">
  Attributname: <%= attributename %>
  Attributwert: <%= attributevalue %>
</fsp:userAttributes>
```



5.7 <fsp:isAuthorized>

Das Tag <fsp:isAuthorized> zeigt den eingeschlossenen Inhalt nur dann an, wenn der Benutzer den Kriterien, die über die Attribute angegeben wurden, erfüllt.

Attribute:

Attribut	Bedeutung	Pflichtparameter
userRange	<p>Gibt die Benutzermenge an, auf die sich das Tag beziehen soll.</p> <p>Mögliche Werte sind:</p> <ul style="list-style-type: none"> ▪ all Das Tag lässt alle Benutzer zu. Die Angaben der Attribute "users" und "groups" werden ignoriert. ▪ loggedIn Das Tag lässt nur angemeldete Benutzer zu und wertet dazu die Attribute "users" und "groups" aus. ▪ notLoggedIn Das Tag lässt nur nicht angemeldete Benutzer zu. Die Angaben der Attribute "users" und "groups" werden ignoriert. <p>Standardwert: <code>loggedIn</code></p>	Nein



Attribut	Bedeutung	Pflichtparameter								
users	<p>Enthält eine kommaseparierte Liste der zugelassenen Benutzer.</p> <p><u>Beispiele:</u></p> <table border="1" data-bbox="480 499 1023 987"> <tr> <td data-bbox="480 499 1023 533">users="Benutzer1,Benutzer2"</td> <td data-bbox="480 533 1023 600">Benutzer1 und Benutzer2 werden zugelassen.</td> </tr> <tr> <td data-bbox="480 600 1023 633">users=""</td> <td data-bbox="480 633 1023 701">Angemeldete Benutzer werden nicht zugelassen.</td> </tr> <tr> <td data-bbox="480 701 1023 734">users="*"</td> <td data-bbox="480 734 1023 824">Alle (angemeldeten) Benutzer werden zugelassen.</td> </tr> <tr> <td data-bbox="480 824 1023 857">users="Benutzer1,*"</td> <td data-bbox="480 857 1023 987">Alle (angemeldeten) Benutzer werden zugelassen; die Angabe von Benutzer1 ist überflüssig.</td> </tr> </table> <p>Standardwert: *</p>	users="Benutzer1,Benutzer2"	Benutzer1 und Benutzer2 werden zugelassen.	users=""	Angemeldete Benutzer werden nicht zugelassen.	users="*"	Alle (angemeldeten) Benutzer werden zugelassen.	users="Benutzer1,*"	Alle (angemeldeten) Benutzer werden zugelassen; die Angabe von Benutzer1 ist überflüssig.	Nein
users="Benutzer1,Benutzer2"	Benutzer1 und Benutzer2 werden zugelassen.									
users=""	Angemeldete Benutzer werden nicht zugelassen.									
users="*"	Alle (angemeldeten) Benutzer werden zugelassen.									
users="Benutzer1,*"	Alle (angemeldeten) Benutzer werden zugelassen; die Angabe von Benutzer1 ist überflüssig.									



Attribut	Bedeutung	Pflichtparameter				
groups	<p>Enthält eine kommaseparierte Liste der zugelassenen Benutzergruppen.</p> <p><u>Beispiele:</u></p> <table border="1" data-bbox="480 499 1023 533"> <tr> <td>groups="Gruppe1,Gruppe2"</td> </tr> </table> <p>Benutzer der Gruppen Gruppe1 und Gruppe2 werden zugelassen.</p> <table border="1" data-bbox="480 629 1023 663"> <tr> <td>groups=""</td> </tr> </table> <p>Angemeldete Benutzer werden nicht zugelassen, da auch keine Gruppen zugelassen sind (dies trifft auch auf Benutzer zu, die keiner Gruppe zugeordnet sind).</p> <table border="1" data-bbox="480 808 1023 842"> <tr> <td>groups="*"</td> </tr> </table> <p>Alle (angemeldeten) Benutzer, die mindestens einer Gruppe zugeordnet sind, werden zugelassen.</p> <table border="1" data-bbox="480 965 1023 999"> <tr> <td>groups="Gruppe1,*"</td> </tr> </table> <p>Alle (angemeldeten) Benutzer, die mindestens einer Gruppe zugeordnet sind, werden zugelassen; die Angabe von Gruppe1 ist überflüssig.</p> <p>Default: *</p>	groups="Gruppe1,Gruppe2"	groups=""	groups="*"	groups="Gruppe1,*"	Nein
groups="Gruppe1,Gruppe2"						
groups=""						
groups="*"						
groups="Gruppe1,*"						



Attribut	Bedeutung	Pflichtparameter
exclude	<p>Schließt Benutzer bzw. Gruppen aus, wenn der Wert auf "true" gesetzt ist. Das Attribut wird nur in Verbindung mit <code>userRange="all"</code> oder <code>userRange="loggedIn"</code> ausgewertet.</p> <p>Beispiele:</p> <pre>userRange="all" exclude="true"</pre> <p>Mit dieser Konfiguration werden alle Benutzer ausgeschlossen, das heißt, der Inhalt des Tags wird keinem Benutzer angezeigt.</p> <pre>userRange="loggedIn" exclude="true" users="Benutzer1"</pre> <p>Nur <code>Benutzer1</code> wird ausgeschlossen. Alle anderen (angemeldeten) Benutzer sehen den Inhalt des Tags.</p> <pre>userRange="loggedIn" exclude="true" users="*"</pre> <p>Mit dieser Konfiguration werden alle angemeldeten Benutzer ausgeschlossen, d. h. weder angemeldete noch nicht angemeldete Benutzer sehen den Inhalt des Tags.</p> <p>Für den Fall <code>userRange="loggedIn"</code> bezieht sich das Tag weiterhin nur auf die Menge der angemeldeten Benutzer, auch wenn <code>exclude="true"</code> gesetzt ist! Das heißt:</p> <pre>user="*" exclude="true"</pre> <p>bedeutet → alle angemeldeten Benutzer ausschließen aber NICHT → alle nicht angemeldeten Benutzer erlauben</p> <p>Die Funktionsweise des Tags ist folgendermaßen zu verstehen: "Verwende als Basismenge der eventuell zugelassenen Benutzer nur die angemeldeten Benutzer. Davon schließe alle Benutzer aus (<code>user="*" exclude="true"</code>). (Also lasse keinen Benutzer zu.)"</p> <p>Default: <code>false</code></p>	Nein



Beispiel:

```
<fsp:isAuthorized
  userRange="loggedIn"
  users="admin"
  groups="projektleiter"
  exclude="false">

  Geschützter Inhalt

</fsp:isAuthorized>
```

5.8 <fsp:isNotAuthorized>

Das Tag <fsp:isNotAuthorized> ist die exakte Umkehrung bzw. Verneinung des Tags <fsp:isAuthorized> (siehe Kapitel 5.7 Seite 57). Das Tag <fsp:isNotAuthorized> verfügt über dieselben Attribute, wie das Tag <fsp:isAuthorized>. Die Bedeutung aller Attribute ist identisch. Der Unterschied zwischen den beiden Tags liegt darin, dass das Tag <fsp:isNotAuthorized> seinen Inhalt genau dann anzeigt, wenn das Tag <fsp:isAuthorized> seinen Inhalt verbirgt (also wenn der Benutzer nicht den angegebenen Kriterien entspricht). Zeigt <fsp:isAuthorized> hingegen seinen Inhalt an, dann verbirgt <fsp:isNotAuthorized> seinen Inhalt. Zusammenfassend kann festgehalten werden, dass der Inhalt beider Tags niemals gleichzeitig angezeigt wird, sondern entweder der Inhalt von <fsp:isAuthorized> oder von <fsp:isNotAuthorized>.

Beispiel:

```
<fsp:isNotAuthorized
  userRange="loggedIn"
  users="admin"
  groups="projektleiter"
  exclude="false">

  Ungeschützter Inhalt

</fsp:isNotAuthorized>
```

5.9 <fsp:logout>

Das Tag <fsp:logout> meldet den aktuell angemeldeten Benutzer ab.

Beispiel:

```
<fsp:logout />
```



6 Erweiterung der Funktionalität

Dieses Kapitel behandelt Möglichkeiten, um die Funktionalität von FirstSpirit DynamicPersonalization gezielt an die Anforderungen eines Projekts anzupassen.

6.1 Session-Variablen

Nach erfolgreichem Login sind folgende Session-Variablen verfügbar:

Variable	Bedeutung	Datentyp
FIRSTpersonalisation.user	Der aktuelle Benutzer (siehe Kapitel 6.2 "Interface")	de.espirit.firstspirit.opt.personalisation.User
FIRSTpersonalisation.usergroups	Gruppen des aktuellen Benutzers als Liste von Strings	java.util.List<java.lang.String>

6.2 Interface "User"

Das Interface `de.espirit.firstspirit.opt.personalisation.User` stellt folgende Methoden zur Verfügung:

- `getLogin()`:
Rückgabe des Loginnames des Benutzers (vgl. Kapitel 6.2.1 Seite 63).
- `getGroups()`:
Rückgabe der Gruppen des Benutzers (vgl. Kapitel 6.2.2 Seite 64).



- addGroup(String):
Eine Gruppe dem Benutzer hinzufügen
(vgl. Kapitel 6.2.3 Seite 64).
- setGroups(List<String>):
Setzen der Gruppen des Benutzers
(vgl. Kapitel 6.2.4 Seite 64).
- isInGroup(String):
Prüfung, ob der Benutzer der übergebenen Gruppe angehört
(vgl. Kapitel 6.2.5 Seite 65).
- isInGroups(String):
Prüfung, ob der Benutzer mindestens einer der übergebenen Gruppen angehört
(vgl. Kapitel 6.2.6 Seite 65).
- getAttributes():
Rückgabe aller Attribute des Benutzers
(vgl. Kapitel 6.2.7 Seite 65).
- getAttribute(String):
Rückgabe eines bestimmten Attributwertes des Benutzers
(vgl. Kapitel 6.2.8 Seite 66).
- addAttributes(Map<String, String>):
Mehrere Attribute dem Benutzer hinzufügen
(vgl. Kapitel 6.2.9 Seite 66).
- setAttribute(String, String):
Ein Attribut dem Benutzer hinzufügen
(vgl. Kapitel 6.2.10 Seite 66).
- clearAttributes():
Attribute des Benutzers zurücksetzen
(vgl. Kapitel 6.2.11 Seite 67).

6.2.1 Methode "getLogin()"

Die Methode "getLogin()" liefert den Loginnamen des Benutzers als String zurück.

Methodensignatur:

```
public java.lang.String getLogin();
```

Methodensignatur (Kurzform):

```
getLogin():String
```



6.2.2 Methode "getGroups()"

Die Methode "getGroups()" liefert die Gruppen, welchen der Benutzer angehört, als Liste von Strings zurück.

Methodensignatur:

```
public java.util.List<java.lang.String> getGroups();
```

Methodensignatur (Kurzform):

```
getGroups():List<String>
```

6.2.3 Methode "addGroup(String)"

Mit der Methode "addGroup(String)" kann eine Gruppe zum Benutzer hinzugefügt werden. Die Gruppen-Informationen für einen Benutzer werden über das Gruppen-Modul ermittelt. Soll der Benutzer zusätzlich weitere Gruppenzuordnungen besitzen (z. B. aus einer externen Datenbank), können diese Gruppen über die Methode "addGroup(String)" hinzugefügt werden. Als Übergabeparameter wird der Name der hinzuzufügenden Gruppe erwartet.

Methodensignatur:

```
public void addGroup(java.lang.String groupName);
```

Methodensignatur (Kurzform):

```
addGroup(String):void
```

6.2.4 Methode "setGroups(List<String>)"

Mit der Methode "setGroups(List<String>)" können die Gruppen eines Benutzers gesetzt werden. Eventuelle Gruppenzuweisungen werden dabei nicht übernommen, d.h. die Gruppen werden "neugesetzt".

Methodensignatur:

```
public void setGroups(java.util.List<java.lang.String> groups)
```

Methodensignatur (Kurzform):

```
setGroups(List<String>):void
```



6.2.5 Methode "isInGroup(String)"

Mit der Methode "isInGroup(String)" kann überprüft werden, ob ein Benutzer einer bestimmten Gruppe angehört oder nicht. Die Methode gibt einen Booleschen Wert zurück. Es wird "true" zurückgeliefert, wenn der Benutzer der übergebenen Gruppe angehört und "false", wenn er der Gruppe nicht angehört.

Methodensignatur:

```
public boolean isInGroup(java.lang.String groupName)
```

Methodensignatur (Kurzform):

```
isInGroup(String):boolean
```

6.2.6 Methode "isInGroups(String)"

Mit der Methode "isInGroups(String)" kann überprüft werden, ob ein Benutzer einer der angegebenen Gruppen angehört oder nicht. Der Methode ist eine kommaseparierte Liste von Gruppennamen zu übergeben. Die Methode gibt einen Booleschen Wert zurück. Es wird "true" zurückgeliefert, wenn der Benutzer mindestens einer der übergebenen Gruppen angehört und "false", wenn er keiner Gruppe angehört.

Methodensignatur:

```
public boolean isInGroups(java.lang.String groupNames)
```

Methodensignatur (Kurzform):

```
isInGroups(String):boolean
```

6.2.7 Methode "getAttributes()"

Die Methode "getAttributes()" liefert alle Attribute eines Benutzers als Map zurück. Jedes Attribut ist ein Schlüssel-Wert-Paar der Map und besteht aus dem Attributnamen ("Schlüssel") und dem Attributwert ("Wert").

Methodensignatur:

```
public java.util.Map<java.lang.String, java.lang.String>  
getAttributes()
```

Methodensignatur (Kurzform):

```
getAttributes():Map<String, String>
```



6.2.8 Methode "getAttribute(String)"

Mit der Methode "getAttribute(String)" wird der Wert eines bestimmten Attributes des Benutzers zurückgeliefert.

Methodensignatur:

```
public java.lang.String getAttribute(final java.lang.String name)
```

Methodensignatur (Kurzform):

```
getAttribute(String) :String
```

6.2.9 Methode "addAttributes(Map<String, String>)"

Mit der Methode "addAttributes(Map<String, String>)" können dem Benutzer mehrere Attribute hinzugefügt werden. Die Menge der hinzuzufügenden Attribute ist die Menge der Schlüssel-Wert-Paare der übergebenen Map.

Methodensignatur:

```
public void addAttributes(java.util.Map<java.lang.String,  
java.lang.String> attributes)
```

Methodensignatur (Kurzform):

```
addAttributes(Map<String, String>) :void
```

6.2.10 Methode "setAttribute(String, String)"

Mit der Methode "setAttribute(String, String)" kann dem Benutzer genau ein Attribut hinzugefügt werden. Bei dem ersten Übergabeparameter handelt es sich um den Attributnamen ("Schlüssel") und bei dem zweiten um den Attributwert ("Wert").

Methodensignatur:

```
public void setAttribute(java.lang.String name, java.lang.String  
value)
```

Methodensignatur (Kurzform):

```
setAttribute(String, String) :void
```



6.2.11 Methode "clearAttributes()"

Mit der Methode "clearAttributes()" werden die Attribute des Benutzers zurückgesetzt. Nach dem Aufruf ist die Menge der Attribute "leer".

Methodensignatur:

```
public void clearAttributes()
```

Methodensignatur (Kurzform):

```
clearAttributes():void
```



7 Konfiguration mit JOSSO (Java Open Single Sign-On) ²

Zur Ausführung der folgenden Schritte, muss JOSSO bereits auf dem Webserver installiert sein.

7.1 Konfiguration der Personalisierung

Für die Konfiguration wird das Login-Modul "Request Header Login" der Personalisierung benötigt, welches den Benutzernamen aus dem Request-Header ausliest (siehe Kapitel 3.1.5 Seite 27).

Da nur Benutzernamen von bereits authentifizierten Benutzern im Header erwartet werden, benötigt dieses Modul kein Authentifizierungs-Modul.

Gruppen- und Attribut-Module können dann beliebig gewählt werden.

7.2 Konfiguration des FirstSpirit-Projekts

7.2.1 Über Strukturvariablen oder Projekteinstellungen

Für die Konfiguration mit JOSSO müssen im Projekt folgende Variablen definiert werden. Die Variablen können entweder innerhalb der Struktur-Verwaltung des Projekts über Strukturvariablen definiert werden oder über die Projekteinstellungen. Die angegebenen Variablennamen stellen eine Empfehlung dar.

Variablenname	Wert
ps_josso_loginUrl	http://MYSERVER/josso/signon/usernamePasswordLogin.do (MYSERVER ist gegen die wirkliche Server-URL auszutauschen)
ps_josso_logoutUrl	http://MYSERVER/josso/signon/logout.do (MYSERVER ist gegen die wirkliche Server-URL auszutauschen)

² <http://www.josso.org/>



ps_josso_backTo	Seite (innerhalb des Projekts), die nach erfolgreichem Login/Logout aufgerufen werden soll.
ps_josso_onError	Seite (innerhalb des Projekts), die nach einem Fehler aufgerufen werden soll.

7.2.2 Konfiguration der personalisierten Seiten (Inhalte-Verwaltung)

Jede personalisierte JSP-Inhaltsseite, muss den folgenden Code am Anfang der Seite enthalten.

```
<%@ taglib uri="FIRSTpersonalisation" prefix="fsp" %>
<fsp:authorize/>
```

7.2.2.1 Beispielkonfiguration (Anmeldeformular)

```
<form
  name="login"
  method="post"
  action="$CMS_VALUE (ps_josso_loginUrl) $">
  <input
    type="hidden"
    name="josso_cmd"
    value="login" />
  <input
    type="hidden"
    name="josso_back_to"
    value="$CMS_VALUE (ps_josso_backTo) $" />
  <input
    type="hidden"
    name="josso_on_error"
    value="$CMS_VALUE (ps_josso_onError) $" />
  <input
    type="text"
    name="josso_username"
    value="" />
  <input
    type="password"
    name="josso_password"
    value="" />
  <input
    type="submit"
    value="Login" />
</form>
```



7.2.2.2 Beispielkonfiguration (Abmeldeformular)

```
<form
  name="jossologout"
  method="post"
  action="$CMS_VALUE(ps_josso_logoutUrl) $">
  <input
    type="hidden"
    name="josso_back_to"
    value="$CMS_VALUE(ps_josso_backTo) $" />
  <input
    type="submit"
    value="Logout" />
</form>
```



8 Rechtliche Hinweise

Das Modul "FirstSpirit™ Dynamic Personalization" ist ein Produkt der e-Spirit AG, Dortmund, Germany.

Für die Verwendung des Moduls gilt gegenüber dem Anwender nur die mit der e-Spirit AG vereinbarte Lizenz.

Details zu möglicherweise fremden, nicht von der e-Spirit AG hergestellten, eingesetzten Software-Produkten, deren eigenen Lizenzen und gegebenenfalls Aktualisierungs-Informationen, finden Sie auf der Startseite jedes FirstSpirit-Servers im Bereich "Rechtliche Hinweise".

