

# FirstSpirit™

*Unlock Your Content*

## RealtimeTargeting FirstSpirit Version 5.x

<b>Version</b>	<b>1.2</b>
<b>Status</b>	<b>RELEASED</b>
<b>Datum</b>	<b>2015-01-20</b>
<b>Abteilung</b>	Produkt Management
<b>Copyright</b>	2014 e-Spirit AG

### e-Spirit AG

Barcelonaweg 14  
44269 Dortmund | Germany

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

[info@e-spirit.com](mailto:info@e-spirit.com)  
[www.e-spirit.com](http://www.e-spirit.com)

e-Spirit

# Inhaltsverzeichnis

<b>1</b>	<b>Einleitung</b> .....	<b>4</b>
1.1	Architekturübersicht .....	6
1.1.1	User-Experience Pipeline .....	7
1.1.2	Profiling & Segmentation Service .....	7
1.1.3	Content-Erstellung und Zuweisung in FirstSpirit .....	8
1.1.4	Content-Auslieferung .....	8
1.2	Technische Voraussetzungen .....	9
1.3	Datenschutz .....	9
1.4	Themen dieser Dokumentation .....	10
<b>2</b>	<b>Komponenten</b> .....	<b>11</b>
2.1	FirstSpirit-Modul .....	11
2.2	FirstSpirit-Komponenten .....	11
2.3	Persona-Simulator .....	12
2.4	Profiling & Segmentation Service: Woopra™ .....	13
<b>3</b>	<b>Installation und Konfiguration</b> .....	<b>14</b>
3.1	Registrierung und Konfiguration von Woopra™ .....	14
3.2	Installation des Moduls .....	16
3.3	Konfiguration der Projekt-Komponenten .....	17
3.4	Einrichtung des Persona-Simulators .....	19
3.4.1	Auswahl der Persona Preview Parameter .....	19
3.4.2	Anlegen der Persona-Auflösung .....	19
3.5	Konfiguration der Web-Komponenten .....	21
3.5.1	web.xml .....	22
3.5.2	Pipeline-Filter .....	23



3.5.2.1 UxpServletFilter .....	24
3.5.2.2 WoopraCookieFilter.....	24
3.5.2.3 WoopraFirstVisitFilter .....	24
3.5.2.4 GetWoopraUserInformationFilter .....	25
3.5.2.5 RequestParamFilter .....	26
3.5.2.6 ExtractUserInformationFilter.....	26
3.6 Proxy-Unterstützung.....	27
<b>4 Projektanpassungen .....</b>	<b>29</b>
4.1 WoopraValueProvider.....	29
4.2 ContentTargeting-Tag.....	30
4.2.1 Präfix .....	31
4.2.2 <rtt:showContent> .....	32
4.3 Persona-Simulator .....	34
4.4 Tracking-Code.....	36
<b>5 Verwendung in FirstSpirit .....</b>	<b>37</b>
5.1 FS_LIST .....	37
5.2 Erweiterte Anwendungsfälle.....	40
5.2.1 Smart Forms .....	40
5.2.2 Scoring.....	46
<b>6 Erweiterungen .....</b>	<b>50</b>
6.1 Implementierung zusätzlicher Filter für die UXP.....	50
6.1.1.1 ExamplelpFilter.....	51
6.1.1.2 ExampleLocationFilter .....	52
6.1.1.3 Ausgabe im Template.....	53
6.2 Austausch des Profiling & Segmentation Service .....	53
6.2.1 Bereitstellung der Liste aller Segmente .....	54



- 6.2.2 Filter zur Ermittlung der Informationen eines Besuchers ..... 55
- 6.2.3 Tracking des Besucherverhaltens ..... 55
  
- 7 Tutorial ..... 56**
  - 7.1 Seitenvorlage ..... 56
    - 7.1.1 Taglibs einbinden ..... 56
    - 7.1.2 CSS und JavaScript einbinden..... 56
    - 7.1.3 Previewleiste und Personas einbinden ..... 57
    - 7.1.4 Personas anlegen..... 57
  - 7.2 Absatztemplates ..... 59
    - 7.2.1 Teaser Absatz..... 59
    - 7.2.2 Teaser Varianten Absatz..... 61
  - 7.3 Woopra™ Tracking einrichten ..... 63
  - 7.4 Scoring einrichten..... 63
  - 7.5 SmartForms einrichten ..... 65
    - 7.5.1 Formular anlegen ..... 66
    - 7.5.2 Anpassungen Template „FormularStart“ ..... 66
    - 7.5.3 Anpassungen Template „formText“ ..... 67
    - 7.5.4 Anpassungen web.xml..... 67
  
- 8 Rechtliche Hinweise ..... 68**



# 1 Einleitung

Das FirstSpirit-Modul *Real-time Targeting* ermöglicht eine zielgruppenspezifische und individuell auf den Webseiten-Besucher zugeschnittene Auslieferung von FirstSpirit-Inhalten auf Webseiten. Mit der Hilfe von FirstSpirit können somit jedem einzelnen Besucher die seinen Bedürfnissen entsprechenden Informationen zum richtigen Zeitpunkt auf dem passenden Ausgabegerät bereitgestellt werden.

Im Ergebnis erhöht sich damit für den Webseiten-Besucher die individuelle Relevanz der angebotenen Informationen. Dies führt zu:

- höherer Zufriedenheit
- langfristiger Kundenbindung
- höheren Konversionen
- und zu einer optimierten User-Experience.

Die folgenden Aspekte bilden die Grundlage für eine zielgruppenspezifische Auslieferung von Inhalten auf einer Webseite:

## 1. Zielgruppendefinition

Bei der Projektumsetzung ist zunächst konzeptionell festzulegen, welche Zielgruppen für den konkreten Webauftritt relevant sind. Diese Einteilung ist in jedem Projekt und für jeden Kunden immer individuell. Die Zielgruppen werden häufig auch Segmente oder (Besucher-)Gruppen genannt.

## 2. Zielgruppenerkennung

Nach der Zielgruppendefinition ist festzulegen, wie erkannt bzw. ermittelt wird, welche Webseiten-Besucher in welche Zielgruppe fallen. Die dafür benötigten Regeln hängen stark von den Arten der Zielgruppen ab.

Dazu einige Beispiele:

### a. *Mobile-User vs. Desktop-User*

Da der Browser eines Besuchers diese Information direkt bereitstellt, ist sie sehr einfach zu ermitteln.



b. *Interessent vs. Bestandskunde*

Diese Information lässt sich technisch über verschiedene Wege ermitteln. Sie kann beispielsweise über einen Login-Bereich auf der Webseite festgestellt werden.

c. *Interessent für ein spezifisches Thema*

Die Erfassung des Interesses eines Webseiten-Besuchers für ein konkretes Thema kann explizit oder implizit erfolgen.

Bei der expliziten Zuordnung wird der Besucher direkt gefragt, ob er sich für ein Thema interessiert. Diese Information wird im Besucherprofil gespeichert.

Die implizierte Zuordnung arbeitet in der Regel mit einem „Scoring“-Mechanismus. Dabei wird festgehalten, welche Arten von Artikeln sich der Besucher besonders häufig anschaut. Ab einem festgelegten Schwellwert kann dann eine Zuordnung zu einer Zielgruppe durchgeführt werden.

Für die Erkennung der Zielgruppen ist grundsätzlich ein Profil für jeden Webseiten-Besucher notwendig, um eine hohe Flexibilität bei der Erstellung der Zielgruppenregeln zu erreichen.

### 3. Pflege von Inhalten für unterschiedliche Zielgruppen

Erst die redaktionelle Pflege verschiedener Inhalte sowie die Zuordnung dieser Inhalte zu den unterschiedlichen Zielgruppen – den sogenannten Segmenten – führen zu einem optimalen Besuchserlebnis für den Webseiten-Besucher. Beides erfolgt beim Modul *Real-time Targeting* direkt in der FirstSpirit-Oberfläche. FirstSpirit muss dabei die Liste der definierten Segmente kennen, um diese dem Redakteur bei der Pflege spezifischer Inhalte anbieten zu können.

### 4. Auslieferung zielgruppenspezifischer Inhalte

Die Auslieferung der für einen Webseiten-Besucher passenden Inhalte erfolgt stets in Echtzeit (*Real-time*), da sich die Zuordnung zu einem Segment gegebenenfalls seitenabhängig ändern kann. Das Modul *Real-time Targeting* prüft bei der Auslieferung:

- a. welchen Segmenten der Besucher aktuell zugeordnet ist
- b. welche Seiteninhalte für diese Segmente relevant sind



Neben einer zielgruppenspezifischen Auslieferung von Inhalten unterstützt das Modul *Real-time Targeting* beispielweise über den Zugriff auf einzelne Profildaten (Umsatz im letzten Monat, letzter Login usw.) auch eine Ansprache von Einzelpersonen.

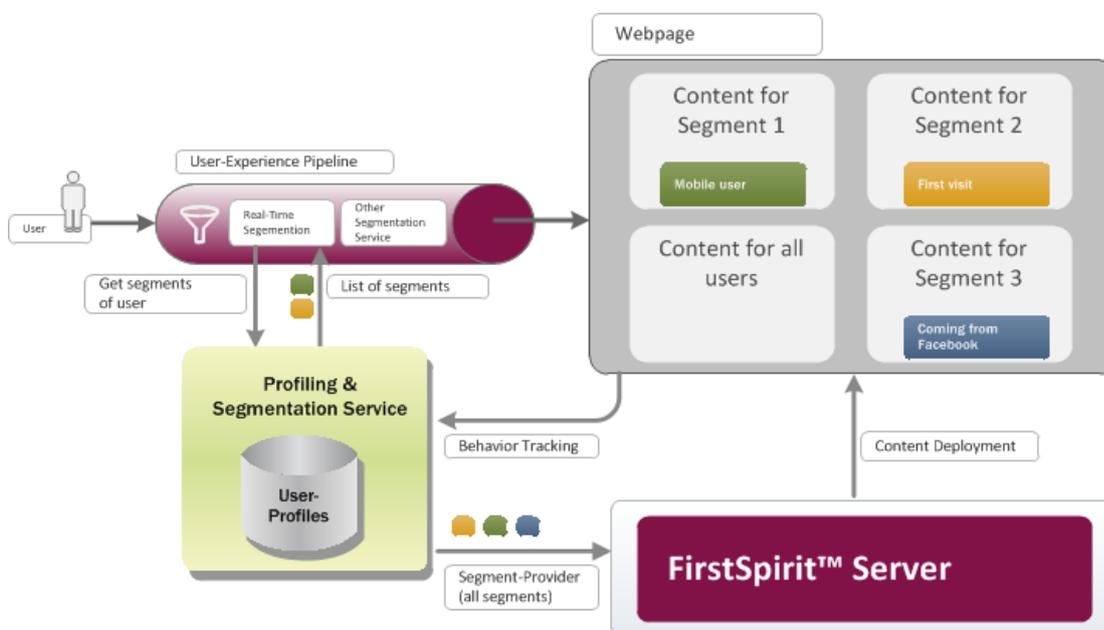
*Das Zusammenspiel der beschriebenen Aspekte wird in der im nachfolgenden Unterkapitel dargestellten Architekturübersicht technisch erläutert.*

## 1.1 Architekturübersicht

Die Architektur des *Real-time Targeting*-Moduls setzt sich aus vier, miteinander interagierenden Elementen zusammen (vgl. *Abbildung 1-1*):

- User-Experience Pipeline Kapitel 1.1.1, Seite 7
- Profiling & Segmentation Service Kapitel 1.1.2, Seite 7
- Content-Erstellung und Zuweisung in FirstSpirit Kapitel 1.1.3, Seite 8
- Content-Auslieferung auf der Webseite Kapitel 1.1.4, Seite 8

*Die Interaktion der einzelnen Elemente wird in den nachfolgenden Unterkapiteln näher erläutert.*



**Abbildung 1-1: Architektur**



### 1.1.1 User-Experience Pipeline

Die User-Experience Pipeline (UXP) wird bei jedem Seitenaufruf eines Webseiten-Besuchers durchlaufen. Ihr Ziel ist die Ermittlung aller Zielgruppeninformationen bzw. Segmente, in denen der aktuelle Besucher enthalten ist. Dazu werden 1 bis n Profiling & Segmentation Services (siehe Kapitel 1.1.2, Seite 7) aufgerufen.

Standardmäßig verwendet das *Real-time Targeting*-Modul genau einen solchen Service (*Woopra™*). Falls andere oder weitere Dienste gewünscht sind, können diese jederzeit projektspezifisch zur UXP hinzugefügt werden. Das Modul besitzt an dieser Stelle eine komplett offene Architektur und ist nicht auf einen spezifischen Dienst beschränkt.

### 1.1.2 Profiling & Segmentation Service

Dieser Dienst übernimmt drei unterschiedliche Aufgaben:

- **Besucher-Tracking und Speicherung von Besucherprofilen**

Über eine Tracking-Logik in der Webseite oder in einer mobilen App werden alle Interaktionen des Besuchers gespeichert und in einem Profil zusammengefasst. Über diese Persistenz steht die komplette Historie eines Webseiten-Besuchers zur Verfügung. Diese Historie kann wiederum für die Definition von Segmenten verwendet werden (vgl. nachfolgender Punkt).

In der Regel können neben Standardaktivitäten (Betrachtung einer bestimmten Webseite, Herunterladen einer konkreten Datei) auch projektspezifische Ereignisse von der Webseite in das Besucherprofil übernommen werden (Besucher hat ein spezielles Produkt gekauft, der Bestellprozess wurde durch den Besucher abgebrochen usw.).

- **Regeldefinition für Segmente**

Auf Basis der Besucherprofile kann über Regeln die Zugehörigkeit von Besuchern zu Segmenten definiert werden. Üblicherweise besitzt der Profiling & Segmentation Service eine grafische Oberfläche zur Definierung der Regeln. Die Regeln können sowohl einmalig beim Aufsetzen des Projekts als auch jederzeit ad hoc erstellt oder modifiziert werden.

Der Service muss die Liste aller vorhandenen Segmente per API an FirstSpirit liefern können, damit diese den Redakteuren für die Content-Zuweisung zur Verfügung steht.



- **Real-time Auslieferung der Segmente eines Besuchers**

Eine zentrale Aufgabe des Profiling & Segmentation Services ist die Bereitstellung der einem identifizierten Webseiten-Besucher zugehörigen Segmenten-Liste. Diese Schnittstelle muss „Real-time“-fähig sein, damit die nachgelagerte Auslieferung der Webseite nicht verzögert wird. Die Identifizierung des Webseiten-Besuchers erfolgt normalerweise über eine ID (z. B. ein Cookie).

Das FirstSpirit-Modul *Real-time Targeting* unterstützt standardmäßig die Anbindung des Woopra™-Services (<https://www.woopra.com>). Woopra™ verfügt über die drei dargestellten Schnittstellen und enthält fertige Implementierungen für ihre Verwendung. Des Weiteren zeichnet sich Woopra™ durch eine fertige Integration in CRM-Systeme (<http://salesforce.com>) aus, wodurch weitere sinnvolle Anwendungsszenarien ermöglicht werden.

Andere Dienste sind jedoch, wie bereits beschrieben, jederzeit alternativ einsetzbar.

### 1.1.3 Content-Erstellung und Zuweisung in FirstSpirit

Die Redakteure besitzen die Aufgabe, spezifische Inhalte zu erstellen, um so eine zielgruppenorientierte Kommunikation zu erreichen. Dazu wird in der Regel zunächst eine allgemeingültige Version eines Inhalts erstellt und darauf aufbauend eine spezifische Variante entwickelt. Solch eine Variante wird anschließend einer oder mehreren Zielgruppen bzw. Segmenten zugeordnet.

Eine Webseite enthält damit (physikalisch gesehen) alle Content-Varianten. Zum Auslieferungszeitpunkt werden jedoch nur die passenden Inhalte bereitgestellt (vgl. Kapitel 1.1.4, Seite 8).

### 1.1.4 Content-Auslieferung

Nach der Ermittlung der einem Besucher zugehörigen Segmenten-Liste kann die Laufzeitumgebung des Webserver die „passenden“ Inhalte der Webseite filtern und ausgeben. Es werden immer nur die gemäß der redaktionellen Parametrisierung korrekten Varianten dargestellt. Eine Auslieferung irrelevanter Inhalte findet nicht statt.

Technisch wird dafür eine Java Tag-Library verwendet. Sie vergleicht die in der UXP ermittelten Segmente mit der redaktionellen Konfiguration der Seite und liefert die passenden Inhalte aus.



## 1.2 Technische Voraussetzungen

Das *Real-time Targeting*-Modul besitzt die folgenden technischen Voraussetzungen:

- FirstSpirit-Version: FirstSpirit 5.1 (oder höher)
- Laufzeitumgebung: Tomcat 7 (bzw. Webserver mit einer Servlet-Engine in der Version 3.0 oder höher und JSTL in der Version 1.0 oder höher)
- verwendeter Profiling & Segmentation Service: Woopra™ ([www.woopra.com](http://www.woopra.com))
- JavaScript-Bibliothek: jQuery 1.9.1 (oder höher)



*Das Real-time Targeting-Modul unterstützt aktiv als Laufzeitumgebung aktuell nur Tomcat 7 und 8. Die Tests werden mit Tomcat 8 durchgeführt. Der Einsatz anderer Webserver wurde nicht getestet und wird somit **nicht** gewährleistet.*



*Das Real-time Targeting-Modul verwendet standardmäßig den hier genannten Service Woopra™. Falls jedoch die Verwendung anderer oder weiterer Dienste gewünscht ist, können diese jederzeit projektspezifisch der UXP (vgl. Kapitel 1.1.1, Seite 7) hinzugefügt werden. Eine Beschreibung der dafür notwendigen Schritte finden Sie in Kapitel 6.*

## 1.3 Datenschutz

Wie bereits in der Einleitung beschrieben, dient das *Real-time Targeting*-Modul der Auslieferung spezifischer Webseiten-Inhalte. Dafür werden die Besucher – basierend auf ihren Interaktionen mit der Webseite – zuvor definierten Zielgruppen zugeordnet.

Es wird ausdrücklich darauf hingewiesen, dass für die Zielgruppenerkennung personenbezogene Daten erhoben, genutzt und an den Profiling & Segmentation Service übermittelt werden.

Der Standard-Tracking-Mechanismus von Woopra™ sammelt die folgenden Daten:

- Browsertyp sowie Domain-Name und IP-Adresse
- Geräte-ID und Betriebssystem des Endgeräts



- Referrer sowie Zeitpunkt des Zugriffs und Gesamtdauer des Besuchs
- besuchte Webseiten und geklickte Links
- alle übermittelte Daten

Des Weiteren kommt ein Cookie zum Einsatz, der auf dem Endgerät der Webseiten-Besucher gespeichert wird. Er verwendet eine eindeutige ID zur Identifizierung des entsprechenden Benutzers und ist zwei Jahre ab dessen letzter Interaktion auf der Webseite gültig.

Diese Aspekte besitzen datenschutzrechtliche Relevanz. Die Webseiten-Besucher sind gegebenenfalls auf die Erhebung dieser Daten hinzuweisen.

Werden im Projekt weitere Informationen an Woopra™ gesendet oder wird ein anderer Profiling & Segmentation Service eingesetzt, so ist die genannte Liste entsprechend zu ergänzen bzw. anzupassen.



*Mit dem Real-time Targeting-Modul wird lediglich die Möglichkeit zur Ermittlung, Analyse und Nutzung spezifischer Interessen der Webseiten-Besucher bereitgestellt. Das Thema Datenschutz muss daher jeweils projektspezifisch behandelt werden.*

## 1.4 Themen dieser Dokumentation

Dieses Dokument beschreibt die Funktionalität zur Echtzeit-Erfassung des Verhaltens von Besuchern eines Web-Auftritts, beschreibt die dafür erforderlichen Komponenten und erläutert deren Installation bzw. Konfiguration:

**Kapitel 2:** Dieses Kapitel enthält eine Beschreibung der benötigten Komponenten.

**Kapitel 3:** Die Installation & Konfiguration der Komponenten auf dem FirstSpirit-Server wird in diesem Kapitel dargestellt.

**Kapitel 4:** In diesem Kapitel werden die durchzuführenden Anpassungen innerhalb des verwendeten FirstSpirit-Projekts erläutert.

**Kapitel 5:** Dieses Kapitel skizziert einige Anwendungsszenarien für den Einsatz des *Real-time Targeting*-Moduls.

**Kapitel 6:** Die Schritte zur Erweiterung des Moduls bzw. zum Austausch des verwendeten Profiling & Segmentation Services sind in diesem Kapitel enthalten.



## 2 Komponenten

Das FirstSpirit-Modul *Real-time Targeting* besteht aus verschiedenen Komponenten:

- FirstSpirit-Modul siehe Kapitel 2.1, Seite 11
- FirstSpirit-Komponenten siehe Kapitel 2.2, Seite 11
- Persona-Simulator siehe Kapitel 2.3, Seite 12
- Profiling & Segmentation Service siehe Kapitel 2.4, Seite 13

Die einzelnen Komponenten werden in den folgenden Unterkapiteln vorgestellt.

### 2.1 FirstSpirit-Modul

Das *Real-time Targeting*-Modul ist die Komponente, die auf dem FirstSpirit-Server installiert werden muss.

Über die durch das Modul auf dem FirstSpirit-Server bereitgestellten Funktionalitäten werden die innerhalb des Profiling & Segmentation Services definierten Segmente (vgl. *Kapitel 2.4, Seite 13*) in das verwendete FirstSpirit-Projekt übertragen und müssen dort dem Redakteur bereitgestellt werden, um diesem eine zielgruppenspezifische Zuordnung der Projekt-Inhalte zu ermöglichen. Eine automatische Auswertung dieser Zuordnung erfolgt während der Ausgabe der Inhalte.

Des Weiteren wird den Redakteuren in Form des Persona-Simulators eine Funktionalität zur Überprüfung der vorgenommenen Zuordnung zur Verfügung gestellt (*siehe 2.3, Seite 12*).

### 2.2 FirstSpirit-Komponenten

Die verschiedenen FirstSpirit-Komponenten fungieren als Bindeglied zwischen FirstSpirit und dem verwendeten Profiling & Segmentation Service (*siehe auch Kapitel 2.4, Seite 13*).

Es existieren *zwei Projekt-Komponenten*: Während über die erste Komponente der Template-Import in das verwendete Projekt ausgelöst wird, werden in der zweiten Komponente die Account-Daten des Profiling & Segmentation Services verwaltet. Mit diesen stellt das Modul eine projektspezifische Verbindung zwischen dem FirstSpirit-Server und dem Service her.



Die *Web-Komponenten* steuern die zielgruppenspezifische Ausgabe der durch die Redakteure erzeugten Inhalte in der Vorschau, im ContentCreator und auf der Webseite (*siehe auch Kapitel 2.3, Seite 12*).

Auf diesem Weg werden die Funktionalitäten der beiden Systeme zusammengeführt und optimal genutzt.

## 2.3 Persona-Simulator

Mit dem *Persona-Simulator* wird den Redakteuren in der Vorschau und im ContentCreator eine Funktionalität bereitgestellt, die es ihnen ermöglicht, eine Persona anzunehmen, deren Zugehörigkeit zu bestimmten Zielgruppen fest definiert ist (*siehe auch Kapitel 4.3, Seite 34*). Die Redakteure erhalten so eine zielgruppenspezifische Sicht auf die Webseite und können auf diesem Weg die Darstellung der von ihnen erstellten Inhaltsvarianten überprüfen. Ohne den *Persona-Simulator* würde den Redakteuren immer nur der allgemeine Inhalt angezeigt und eine Überprüfung wäre nur auf der Live-Webseite möglich.

Die Realisierung des *Persona-Simulators* basiert auf der Multi Perspective Preview (*siehe Roadmap 2013-2017 & Release Notes für FirstSpirit 5.1*). Dadurch fügt er sich nahtlos in den ContentCreator ein (*vgl. Abbildung 2-1*).

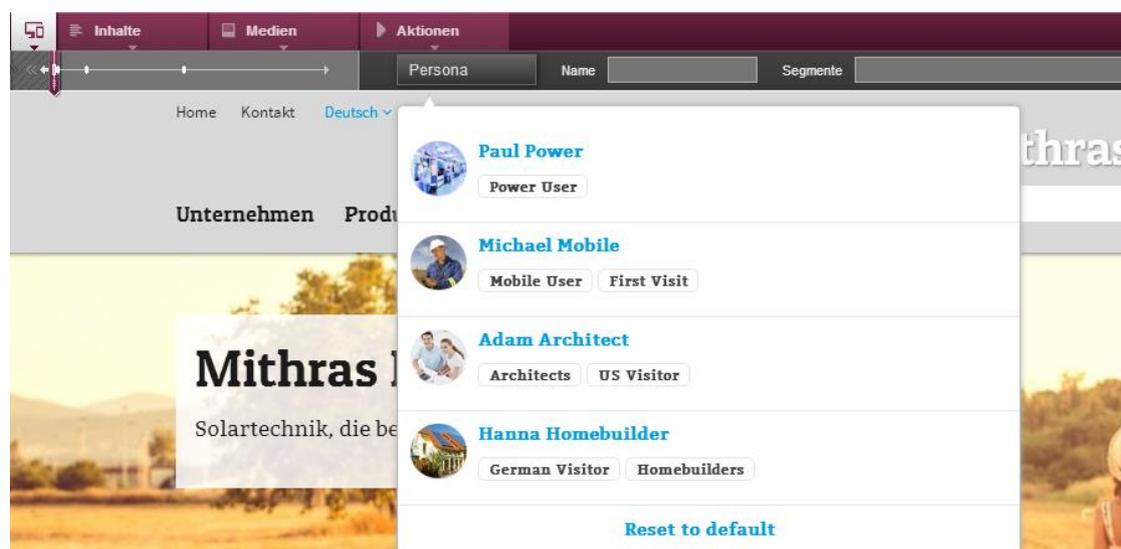


Abbildung 2-1: Persona-Simulator im ContentCreator

Des Weiteren wurde in der Vorschau des SiteArchitects eine Leiste eingefügt, die in Bezug auf den *Persona-Simulator* dem Aussehen und der Funktionalität der ContentCreator-Leiste entspricht (*vgl. Abbildung 2-2*). Die Redakteure können den *Persona-Simulator* somit auch im SiteArchitect verwenden.



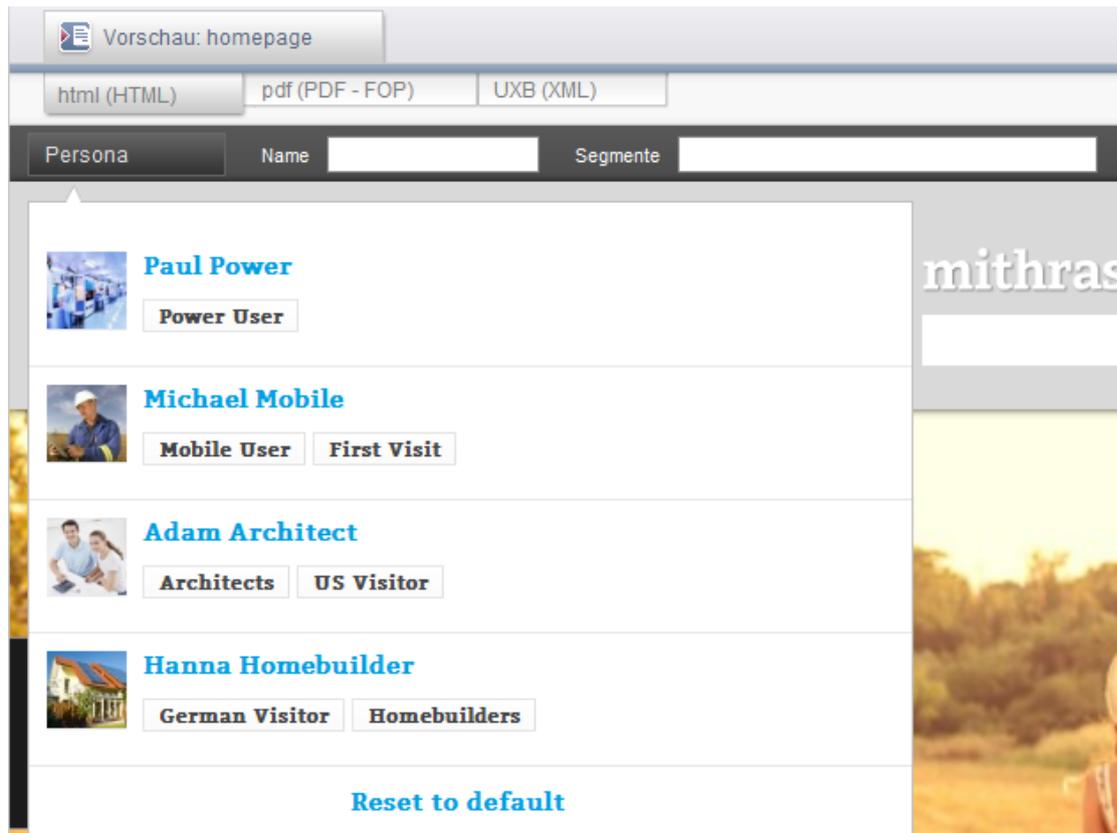


Abbildung 2-2: Persona-Simulator im SiteArchitect

## 2.4 Profiling & Segmentation Service: Woopra™

Woopra™ ist ein Service zur Echtzeit-Erfassung des Verhaltens von Webseiten-Besuchern. Die Woopra-Oberfläche stellt für die Analyse dieses Verhaltens zahlreiche Informationen bereit, die sich bei Bedarf auch manuell erweitern lassen.

Auf diese Weise können verschiedene Besucher erkannt und bei weiteren Besuchen erneut identifiziert werden. Dies bietet die Möglichkeit, sie verschiedenen Zielgruppen (sogenannten Segmenten), die sich innerhalb des Services frei definieren lassen, zuzuordnen und ihnen über FirstSpirit zielgruppenspezifische Inhalte bereitzustellen. Der einzelne Besucher erhält so immer die für ihn relevanten Informationen und seine User-Experience wird optimiert.



*Das Real-time Targeting-Modul verwendet standardmäßig den hier genannten Service Woopra™. Falls jedoch die Verwendung anderer oder weiterer Dienste gewünscht ist, können diese jederzeit projektspezifisch der UXP (vgl. Kapitel 1.1.1, Seite 7) hinzugefügt werden. Eine Beschreibung der dafür notwendigen Schritte finden Sie in Kapitel 6.*



### 3 Installation und Konfiguration

Vor dem Einsatz der *Real-time Targeting*-Funktionalität sind verschiedene Schritte erforderlich:

- Registrierung und Konfiguration von Woopra™ siehe Kapitel 3.1, Seite 14
- Installation des Moduls siehe Kapitel 3.2, Seite 16
- Konfiguration der Projekt-Komponenten siehe Kapitel 3.3, Seite 17
- Einrichtung des Persona-Simulators siehe Kapitel 3.4, Seite 19
- Konfiguration der Web-Komponenten siehe Kapitel 3.5, Seite 21

Die Konfiguration eines Proxy-Servers ist nur dann notwendig, wenn ein solcher für den Zugriff auf das Internet verwendet wird (siehe dazu Kapitel 3.6, Seite 27).

#### 3.1 Registrierung und Konfiguration von Woopra™

Das Tracking eines Besuchers des Web-Auftritts erfolgt über einen entsprechenden Profiling & Segmentation Service, welcher durch FirstSpirit für die Funktionalität des *Real-time Targetings* genutzt, jedoch nicht selbst bereitgestellt wird.

Zur Verwendung des von Woopra™ angebotenen Services wird ein Account benötigt, der auf dieser Seite zu registrieren ist:

<http://www.woopra.com>

Auf der Unterseite *My Websites*, die über den gleichnamigen Punkt im Ausklappmenü unter *My Account* erreichbar ist, muss die zu analysierende Website hinzugefügt werden. Sie wird anschließend in der auf der Seite dargestellten Liste der hinzugefügten Websites angezeigt (vgl. *Abbildung 3-1*).

Website	Date Added	Live Stats	Setup	Package
fsdemo.e-spirit.de	Aug 26, 2013	Live Stats	Setup	Shared

Abbildung 3-1: Woopra™ - My Websites

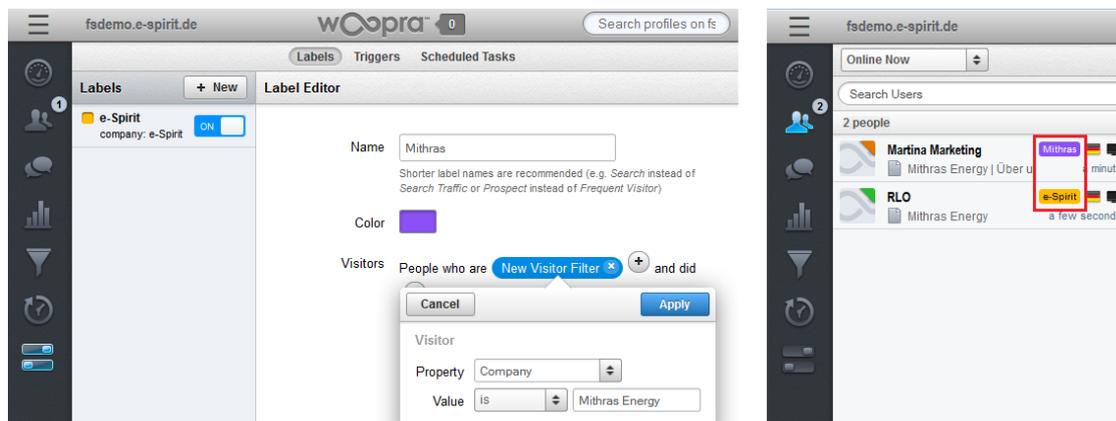


Ein Klick auf *Live Stats* öffnet das Dashboard, in welchem die aktuelle Besucherzahl sowie weitere Informationen in einer Übersicht zusammengestellt sind (vgl. *Abbildung 3-2*).



**Abbildung 3-2: Woopra™ – Dashboard**

Die Besucher des Web-Auftritts lassen sich verschiedenen Segmenten zuordnen. Diese können über den untersten Menüpunkt *Labels* frei definiert werden. Über einen Tracking-Code, der auf der Seite eingebunden werden muss (*siehe Kapitel 4.4, Seite 36*), werden die Aktionen des Besuchers erfasst. Erfüllt ein Besucher aufgrund dieser Aktionen die für die Segmente definierten Bedingungen, wird er dem oder den jeweiligen Segmenten automatisch hinzugefügt (vgl. *Abbildung 3-3*).



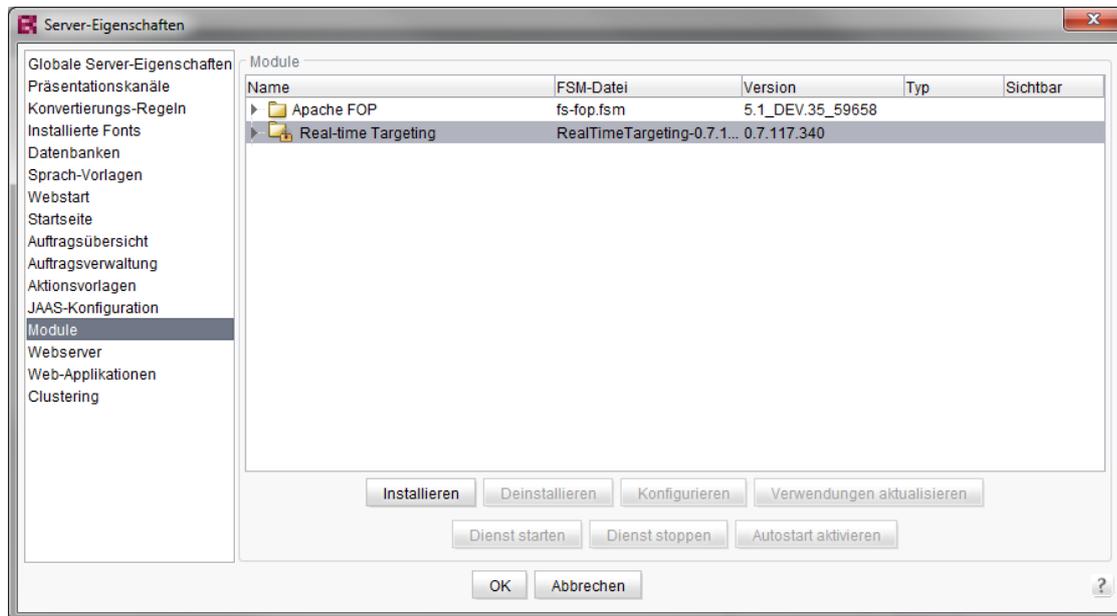
**Abbildung 3-3: Woopra™ – Labels**

Detailliertere Informationen zu Woopra™ entnehmen Sie bitte der Woopra-Dokumentation: <https://www.woopra.com/docs/>  
 Informationen für die Verwendung eines weiteren oder anderen Profiling & Segmentation Services finden Sie in Kapitel 6 ab Seite 50 sowie in der Dokumentation des entsprechenden Anbieters.



## 3.2 Installation des Moduls

Das Modul muss mithilfe der mitgelieferten *rtt-fsm-<Versionsnummer>.fsm*-Datei auf dem FirstSpirit-Server hinzugefügt werden. Für die Installation des Moduls öffnen Sie den *ServerManager* und wählen Sie den Bereich *Module* in den *Server-Eigenschaften*.



**Abbildung 3-4: Modulverwaltung in den Server-Eigenschaften**

Im Hauptpanel ist eine Liste der auf dem Server installierten Module zu sehen. Wählen Sie nach dem Klicken auf *Installieren* die mitgelieferte *rtt-fsm-<Versionsnummer>.fsm* aus und bestätigen Sie die Auswahl mit *Öffnen*. Nach der erfolgreichen Installation wurde der Liste ein Ordner *Real-time Targeting* hinzugefügt, der *Alle Rechte* erhalten muss (siehe *Abbildung 3-4*).



*Nach jeder Modulinstallation bzw. einer Aktualisierung des Moduls ist ein Neustart des FirstSpirit-Servers notwendig.*

Selektieren Sie innerhalb dieses Ordners den *Real-time Targeting Woopra Segment Provider Service* und öffnen Sie die serverweiten Woopra-Einstellungen über einen Klick auf *Konfigurieren* (vgl. *Abbildung 3-5*).

Außer der Angabe eines Proxy-Servers sind die in diesem Dialog dargestellten Informationen bereits initial gefüllt und müssen nur bei Bedarf geändert werden. Soll das Modul über einen Proxy-Server mit dem Profiling & Segmentation Service kommunizieren, müssen dessen Daten hier hinterlegt werden (vgl. auch *Kapitel 3.6*).



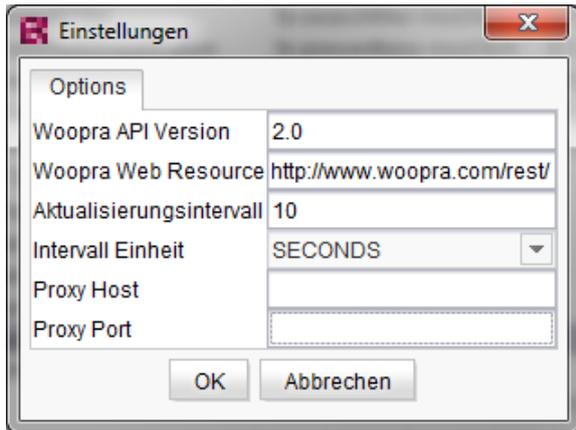


Abbildung 3-5: Einstellungen für Woopra™

Schließen Sie die *Einstellungen* und die *Server-Eigenschaften* nach der Konfiguration jeweils durch Klicken auf *OK*.

Nähere Informationen zur Installationen von Modulen finden Sie in der *FirstSpirit Dokumentation für Administratoren*.

### 3.3 Konfiguration der Projekt-Komponenten

Um dem innerhalb des Moduls enthaltenen *Real-time Targeting Woopra Segment Provider Service* eine Verbindung zu Woopra™ zu ermöglichen und für den Import der benötigten Templates in das verwendete FirstSpirit-Projekt, müssen zwei Projekt-Komponenten hinzugefügt und konfiguriert werden. Öffnen Sie den *Server-Manager* und wählen Sie den Bereich *Projekt-Komponenten* in den *Projekt-Eigenschaften* (siehe Abbildung 3-6).

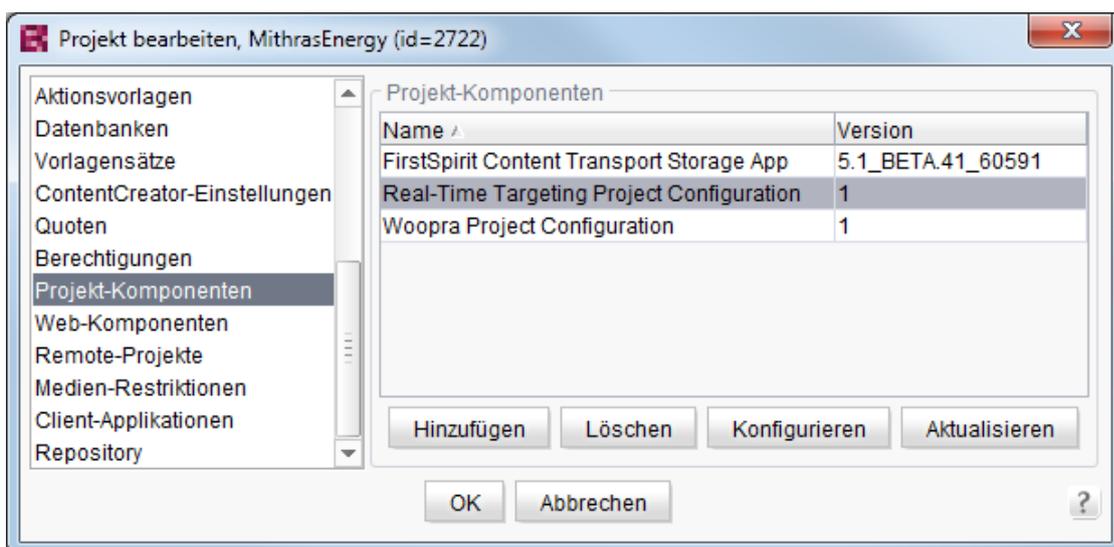


Abbildung 3-6: Projekt-Komponenten



Im Hauptpanel ist eine Liste aller Projekt-Komponenten zu sehen. Wählen Sie nach dem Klicken auf *Hinzufügen* die *Real-time Targeting Project Configuration* sowie (in einem zweiten Schritt) die *Woopra Project Configuration* aus und bestätigen Sie die Auswahl durch *OK*. Beide Projekt-Komponenten werden anschließend der Liste im Hauptpanel hinzugefügt und müssen noch konfiguriert werden.

Selektieren Sie jeweils eine der beiden Komponenten und öffnen Sie den zugehörigen Konfigurationsdialog über *Konfigurieren*.

### Real-time Targeting Project Configuration

Innerhalb des Konfigurationsdialogs der *Real-time Targeting Project Configuration* muss zunächst ein Schema ausgewählt werden, bevor über den gleichnamigen Button die diversen Templates importiert werden können (siehe *Abbildung 3-7*). Das Schema wird für die Datenquelle der *Personas* benötigt (siehe *Kapitel 4.3, Seite 34*).

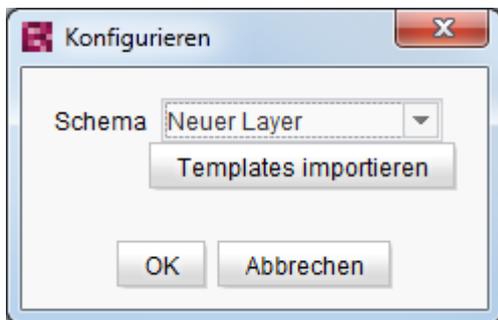


Abbildung 3-7: Real-time Targeting Project Configuration

### Woopra Project Configuration

Innerhalb des Konfigurationsdialogs der *Woopra Project Configuration* müssen die Zugangsdaten für Woopra™ sowie die zu analysierende Webseite hinterlegt werden (vgl. *Abbildung 3-8*). Die Zugangsdaten werden vom Service des Moduls für die Herstellung einer Verbindung zwischen dem FirstSpirit-Server und Woopra™ verwendet. Die angegebene Webseite muss auch in Woopra™ erfasst sein (siehe *Kapitel 3.1, Seite 14*).

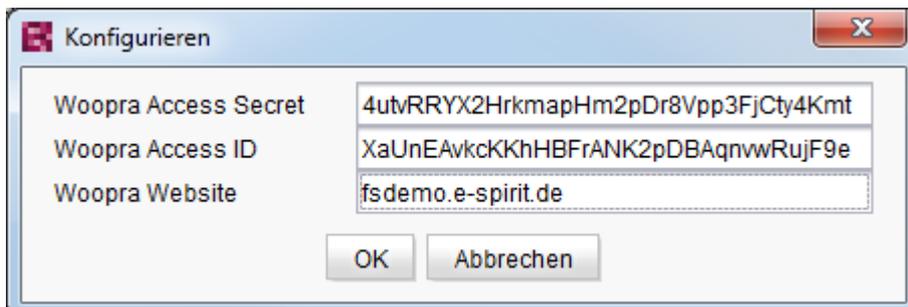


Abbildung 3-8: Woopra Project Configuration



### 3.4 Einrichtung des Persona-Simulators

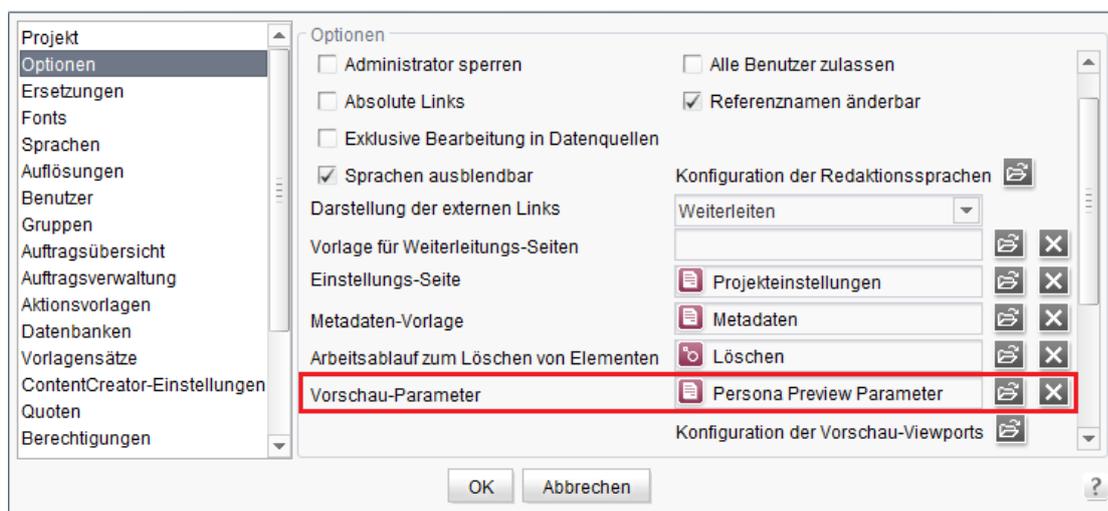
Der Persona-Simulator setzt zwei Anpassungen innerhalb der Projekt-Einstellungen voraus:

- Auswahl der Persona Preview Parameter      siehe Kapitel 3.4.1, Seite 19
- Anlegen der Persona-Auflösung              siehe Kapitel 3.4.2, Seite 19

#### 3.4.1 Auswahl der Persona Preview Parameter

Die Aktivierung der Multi Perspective Preview für die Verwendung des Persona-Simulators im ContentCreator erfolgt durch die Auswahl der *Vorschau-Parameter* in den *Projekt-Optionen*.

Öffnen Sie hierfür den *Server-Manager* und selektieren Sie den Bereich *Optionen* in den *Projekt-Eigenschaften* (vgl. *Abbildung 3-9*). Wählen Sie beim Punkt *Vorschau-Parameter* über den entsprechenden Button das Seiten-Template *Persona Preview Parameter* aus. Dieses Seiten-Template wurde bereits bei der Konfiguration der Projekt-Komponente *Real-time Targeting Project Configuration* automatisch in das Projekt importiert (siehe *Kapitel 3.3, Seite 17*).



**Abbildung 3-9: Vorschau-Parameter auswählen**

#### 3.4.2 Anlegen der Persona-Auflösung

Die einzelnen Personas im Persona-Simulator besitzen ein Bild, für das eine eigene Auflösung angelegt werden muss.



Öffnen Sie hierfür den *Server-Manager* und selektieren Sie den Bereich *Auflösungen* in den *Projekt-Eigenschaften*. Im Hauptpanel ist eine Liste aller bereits existierenden Auflösungen zu sehen.

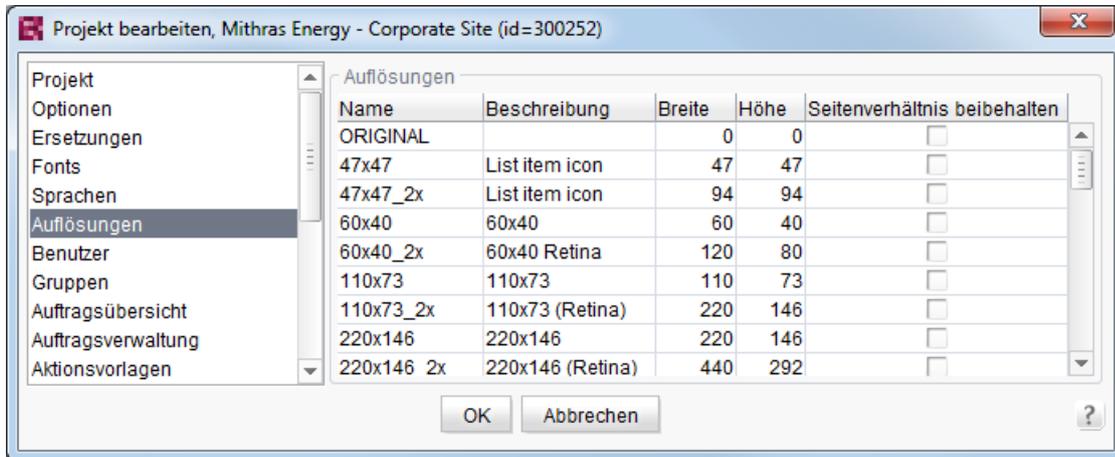


Abbildung 3-10: Auflösungen

Wählen Sie in dem sich per Rechtsklick öffnenden Kontextmenü den Punkt *Neu* und legen Sie eine neue Auflösung mit dem Namen *persona* in der Größe *55x55 Pixel* an.

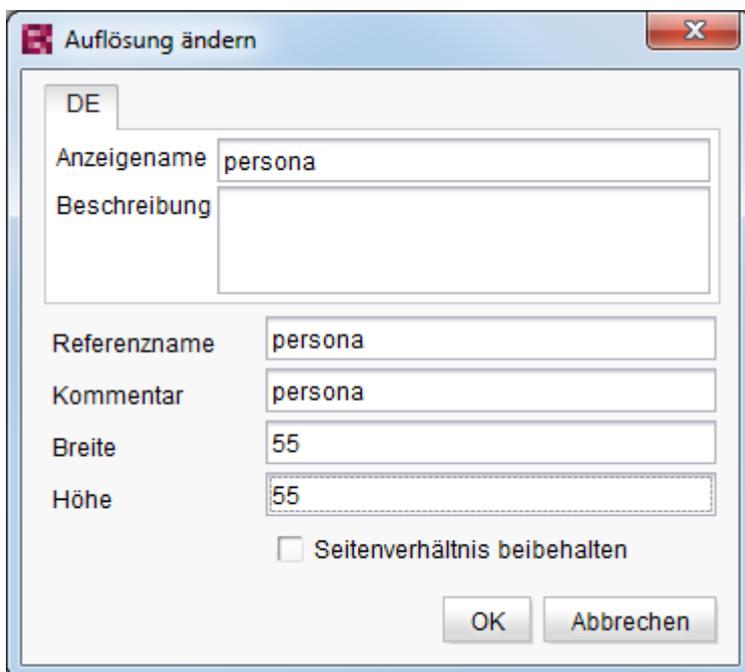


Abbildung 3-11: neue Auflösung anlegen



### 3.5 Konfiguration der Web-Komponenten

Für die *Vorschau*, den *ContentCreator* und die *Produktion* muss jeweils eine Web-Komponente hinzugefügt werden. Öffnen Sie hierfür den *Server-Manager* und wählen Sie den Bereich *Web-Komponenten* in den *Projekt-Eigenschaften*.

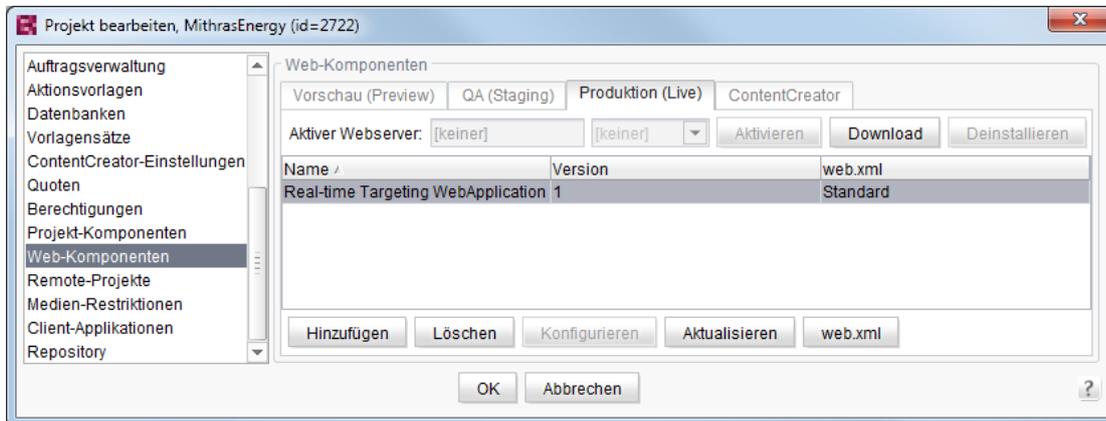


Abbildung 3-12: Web-Komponenten

Innerhalb des Hauptpanels sind verschiedene Registerkarten zu sehen (vgl. *Abbildung 3-12*). Wählen Sie nacheinander *Vorschau (Preview)*, *Produktion (Live)* sowie *ContentCreator* und selektieren Sie nach dem Klicken auf *Hinzufügen* jeweils die *Real-time Targeting WebApplication*, um diese über *OK* hinzuzufügen.

Im Fall des *ContentCreators* und der *Vorschau* muss die Web-Komponente auf einem zur Auswahl stehenden *aktiven Webserver* installiert und anschließend aktiviert werden. Dieser kann über die Auswahlbox gewählt werden.

Für die *Produktion* muss über *Download* eine war-Datei generiert werden, die auf einem Java Web-Application-Server (wie z.B. Tomcat) zu installieren ist. Vorher ist jedoch die Konfiguration der zugehörigen *web.xml* notwendig. Diese wird im nachfolgenden Unterkapitel 3.5.1 beschrieben.



Es wird eine Servlet-Engine benötigt, die die Servlet-API in der Version 3.0 (oder höher) implementiert.

Detailliertere Informationen zum Hinzufügen von Web-Komponenten finden Sie in der *FirstSpirit-Dokumentation für Administratoren*.



### 3.5.1 web.xml

Initial besitzen die drei genannten Web-Komponenten alle dieselbe *web.xml*. Für die Web-Komponente der *Produktion* muss diese initiale *web.xml* jedoch angepasst werden.

Öffnen Sie den *ServerManager* und wählen Sie den Bereich *Web-Komponenten* innerhalb der Eigenschaften des verwendeten Projekts. Wechseln Sie im Hauptpanel auf die Registerkarte *Produktion (Live)* und selektieren Sie dort die *Real-time Targeting WebApplication* (vgl. *Abbildung 3-13*).



**Abbildung 3-13: Web-Komponente Produktion (Live)**

Ein Klick auf den Button *web.xml* öffnet einen Konfigurationsdialog, in dem die *web.xml* editiert werden kann.

Für die Kommunikation mit Woopra™ benötigt die *web.xml* der *Produktion* weitere Filter, die von den anderen beiden Web-Komponenten nicht verwendet werden (*siehe auch Kapitel 3.5.2, Seite 23*). Sie sind daher initial auskommentiert. Aktivieren Sie diese Filter, indem Sie die entsprechenden Kommentar-Kennzeichnungen entfernen. Der *RequestParamFilter* muss im Gegenzug auskommentiert werden, da dieser nur für den *Persona-Simulator* benötigt wird und der gleichzeitige Einsatz der Filter zu unerwünschtem Verhalten führen kann. Speichern Sie die Änderungen anschließend über den gleichnamigen Button.



Es muss darauf geachtet werden, dass die im Filter-Mapping verwendeten URL-Pattern zu den erzeugten Seiten passen (*siehe auch Struktur-Beispiel in Kapitel 3.5.2*).



### 3.5.2 Pipeline-Filter

Für die zielgruppenspezifische Ausgabe der Inhalte auf der Webseite müssen die dem Besucher jeweils aktuell zugeordneten Segmente ermittelt werden. Diese Aufgabe übernehmen die *Pipeline-Filter*:

- UxpServletFilter siehe Kapitel 3.5.2.1, Seite 24
- WoopraCookieFilter siehe Kapitel 3.5.2.2, Seite 24
- WoopraFirstVisitFilter siehe Kapitel 3.5.2.3, Seite 24
- GetWoopraUserInformationFilter siehe Kapitel 3.5.2.4, Seite 25
- RequestParamFilter siehe Kapitel 3.5.2.5, Seite 26
- ExtractUserInformationFilter siehe Kapitel 3.5.2.6, Seite 26

Diese Filter ermöglichen die Bereitstellung einer individualisierten Darstellung der Webseite für verschiedene Zielgruppen. Sie sind modular und können beliebig erweitert werden. Die genaue Definition der jeweils verwendeten Filter erfolgt in der zugehörigen *web.xml* der Web-Komponenten für die Vorschau, die Produktion bzw. den ContentCreator. Anhand dieser Definition in der entsprechenden *web.xml* werden die Filter sukzessiv abgearbeitet.

*Zusätzlich zu den bereits existierenden Filtern können weitere eigene Filter implementiert werden. Eine entsprechende Beschreibung finden Sie in Kapitel 6.1 ab Seite 50.*

Für jeden in einer *web.xml* hinzugefügten Filter wird außerdem ein Mapping benötigt, das definiert, wann der entsprechende Filter greift. Hierbei ist darauf zu achten, dass die im Mapping festgelegte Dateiendung zu den erzeugten Seiten passt. Für Vorschau und ContentCreator wird z.B. das *url-pattern /preview/\** empfohlen.

#### Struktur der Filter-Definition in der *web.xml*

```
<filter>
  <filter-name>FilterName</filter-name>
  <filter-class>FilterClass</filter-class>
</filter>
<filter-mapping>
  <filter-name>FilterName</filter-name>
  <url-pattern>*.DATEIENDUNG</url-pattern>
</filter-mapping>
```





Einige Filter besitzen zusätzliche Parameter, die zwingend angegeben sein müssen. Bei der Installation und Konfiguration des Real-time Targeting-Moduls ist diese Bedingung initial bereits erfüllt.

Lediglich der `GetWoopraUserInfoFilter` muss durch die Woopra-Access-Daten ergänzt werden (vgl. Kapitel 3.5.2.4). Dieser und der `WoopraFirstVisitFilter` benötigen außerdem die Angabe der in Woopra hinterlegten Webseite (vgl. Kapitel 3.1, 3.5.2.3 und 3.5.2.4).

Soll der Profiling & Segmentation Service ausgetauscht werden, müssen die Parameter entsprechend angepasst werden (siehe Kapitel 6, Seite 50).

Sollen weitere Filter zum Einsatz kommen, müssen diese nach demselben Schema ergänzt werden.

### 3.5.2.1 UxpServletFilter

Der `UxpServletFilter` wird **zwingend** zur Steuerung der weiteren Filter benötigt.



Aufgrund seiner Notwendigkeit muss der `UxpServletFilter` im Filter-Mapping **immer** an erster Stelle gesetzt sein.

### 3.5.2.2 WoopraCookieFilter

Der `WoopraCookieFilter` erfasst den Wert des Woopra-Cookies des derzeitigen Besuchers der Webseite und speichert ihn im aktuellen Request-Kontext unter dem Schlüssel `rtt.woopra.cookie`.

Der Filter besitzt den folgenden Parameter:

Name	Bedeutung
cookie_name	Name des gesetzten Cookies.

### 3.5.2.3 WoopraFirstVisitFilter

Beim ersten Aufruf der Webseite besitzt der Besucher noch keinen Woopra-Cookie. Der `WoopraCookieFilter` kann daher nicht greifen. Da Woopra™ zu diesem Zeitpunkt aber bereits Informationen über den Besucher besitzt (z. B. das Land), ist es sinnvoll sie auch direkt bereitzustellen. Aus diesem Grund setzt der `WoopraFirstVisitFilter` den Woopra-Cookie und ruft die Woopra-Tracking API serverseitig auf.





Da der *WoopraFirstVisitFilter* greifen soll, wenn der *WoopraCookieFilter* keinen *Woopra-Cookie* gefunden hat, muss der *WoopraFirstVisitFilter* im Filter-Mapping nach dem *WoopraCookieFilter* und vor dem *GetWoopraLabelsFilter* eingefügt werden.

Name	Bedeutung
cookie_name	Name des gesetzten Cookies
cookie_maxage	die Gültigkeit des Cookies in Sekunden
cookie_domain	die vom Cookie verwendete Domain. Ist der Parameter leer oder nicht angegeben, wird keine Domain gesetzt
cookie_path	Angabe zur Beschränkung der Gültigkeit des Cookies auf einen bestimmten Pfad (default: / )
website	die an Woopra™ gesendete Webseite
woopra_url	URL für den Woopra Tracking Service
woopra_timeout	Timeout, nach dem ein Request als offline markiert wird
request_timeout	Request-Timeout für den Aufruf der Tracking-API

#### 3.5.2.4 GetWoopraUserInformationFilter

Der *GetWoopraUserInformationFilter* dient der Abfrage spezifischer Daten, die zu einem Besucher erfasst und in der Woopra-Umgebung gespeichert wurden. Diese Daten können beispielsweise bei der Verwendung von *Smart Forms* genutzt werden (siehe Kapitel 5.2.1, Seite 40). Für die Abfrage verwendet der *GetWoopraUserInformationFilter* den Wert des Woopra-Cookies des aktuellen Request-Kontextes. Die ermittelten spezifischen Daten des Besuchers werden unter dem Schlüssel *rtt.woopra.user.<Attribut>* gespeichert, während die ihm zugeordneten Segmente an den Schlüssel *segments* übergeben werden.



Name	Bedeutung
woopra_access_id	Access-ID für Woopra™
woopra_access_secret	Access-Secret für Woopra™
website	die an Woopra™ gesendete Webseite
woopra_api_version	Version der Woopra-API
woopra-web_resource	Adresse der Woopra-Rest-Schnittstelle
woopra_date_format	Format für an Woopra™ gesendete Datumsangaben
request_timeout	Request-Timeout für Abfragen an Woopra™
connection_timeout	Connection-Timeout für Abfragen an Woopra™

### 3.5.2.5 RequestParamFilter

Über die Verwendung des *RequestParamFilters* können zu Simulationszwecken die Zielgruppen manuell ausgewählt werden, um so die Funktionalität der *Tags* zu testen. Der Filter ermittelt dafür den Request-Parameter *segments* und speichert dessen Wert in der aktuellen User-Session unter dem Schlüssel *segments*.

Der Request-Parameter *reset* mit dem Wert *true* entfernt den Schlüssel *segments* wieder aus der aktuellen Session.

### 3.5.2.6 ExtractUserInformationFilter

Wird neben dem *Real-time Targeting*-Modul auch das FirstSpirit-Modul *DynamicPersonalization* verwendet, stellt der Filter *ExtractUserInformation* eine Verbindung zwischen den beiden Modulen dar. Er ermittelt innerhalb der aktuellen Session den Benutzer der Personalisierung und speichert dessen Namen, so dass dieser an Woopra™ gesendet werden kann.

Name	Bedeutung
user_session_attribute_name	User der Personalisierung aus der aktuellen Session

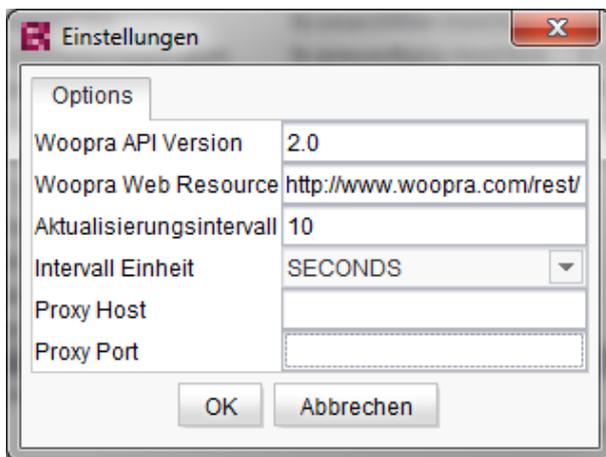


user_name	Name des Users aus der aktuellen Session
-----------	--

### 3.6 Proxy-Unterstützung

Für die Zugriffe auf das Internet kann potentiell ein Proxy-Server eingesetzt werden, der in Abhängigkeit der ihn benötigenden Elemente an verschiedenen Stellen angegeben werden muss.

Sollen die Verbindungen zwischen dem *Real-time Targeting*-Modul und dem Profiling & Segmentation Service über den Proxy-Server erfolgen, so muss er dem Modul innerhalb des *Real-time Targeting Woopra Segment Provider Services* bereitgestellt werden (vgl. *Abbildung 3-14* und *Kapitel 3.2, Seite 16*).



**Abbildung 3-14: Real-time Targeting Woopra Segment Provider Service**

Für die Web-Komponenten bedeutet die Verwendung eines Proxy-Servers lediglich eine Erweiterung der jeweils zugehörigen *web.xml* (siehe auch *Kapitel 3.5, Seite 21*) um einen weiteren Context-Parameter:

```
<context-param>
  <param-name>rtt.httpproxy</param-name>
  <param-value>proxy.mycompany.com:8081</param-value>
</context-param>
```



*Der Parameter-Name rtt.httpproxy darf nicht verändert werden.*

*Die Angabe des Proxy-Servers wird in der Form server:port erwartet.*



Die Verwendung eines Proxy-Servers kann selbstverständlich auch über die Java-Systemeigenschaften `http.proxyHost` und `http.proxyPort` festgelegt werden. Allerdings werden diese Angaben bei der Verwendung des zuvor genannten Parameters ignoriert.



## 4 Projektanpassungen

Nach der Installation des Moduls sowie der Konfiguration der FirstSpirit-Komponenten (*siehe Kapitel 3, Seite 14*) müssen innerhalb des verwendeten Projekts verschiedene Anpassungen vorgenommen werden:

- Bereitstellung des `WoopraValueProviders` siehe Kapitel 4.1, Seite 29
- Einbindung des `ContentTargeting-Tags` siehe Kapitel 4.2, Seite 30
- Definition der `Personas` siehe Kapitel 4.3, Seite 34
- Einbindung des `Tracking-Codes` siehe Kapitel 4.4, Seite 36

Der `WoopraValueProvider` stellt die aus dem Profiling & Segmentation Service ermittelten Segmente innerhalb des verwendeten Projekts bereit, so dass die Redakteure ihnen die zielgruppenspezifischen Inhalte zuordnen können.

Das `ContentTargeting-Tag` steuert die richtige Darstellung dieser Inhalte auf der Webseite.

Die `Personas` ermöglichen den Redakteuren die Überprüfung der zielgruppenspezifischen Zuordnungen der verschiedenen Inhaltsvarianten.

Der `Tracking-Code` ermittelt das Verhalten des Besuchers auf der Webseite.

### 4.1 WoopraValueProvider

Die Bereitstellung der aus dem Profiling & Segmentation Service (*siehe Kapitel 2.4 und 3.1*) ermittelten Segmente für die Redakteure wird über einen `GomIncludeValueProvider` ermöglicht. Dabei handelt es sich um eine standardmäßig von FirstSpirit bereitgestellte Schnittstelle, mit der dynamische Wertemengen an die folgenden Eingabekomponenten übergeben werden können:

- `CMS_INPUT_CHECKBOX`
- `CMS_INPUT_COMBOBOX`
- `CMS_INPUT_LIST`
- `CMS_INPUT_RADIOBUTTON`

Die automatische Übergabe der dynamischen Werte an die entsprechende Eingabekomponente erfolgt über die Einbindung des Datenelements `CMS_INCLUDE_OPTIONS`, in dem der Klassenname des verwendeten



`GomIncludeValueProviders` angegeben werden muss. In diesem Fall handelt es sich dabei um die Klasse `com.espirit.moddev.rtt.gui.WoopraValueProvider`.

Das folgende Beispiel zeigt die Einbindung des `WoopraValueProviders` in der Eingabekomponente `CMS_INPUT_CHECKBOX` (vgl. auch *Abbildung 4-1*):

```
<CMS_INPUT_CHECKBOX
  name="st_segments" gridWidth="3" hFill="yes" useLanguages="no">
  <CMS_INCLUDE_OPTIONS type="public">
    <LABELS>
      <LABEL lang="*">#item</LABEL>
    </LABELS>
    <NAME>com.espirit.moddev.rtt.gui.WoopraValueProvider</NAME>
  </CMS_INCLUDE_OPTIONS>
  <LANGINFOS>
    <LANGINFO lang="*" label="Show To Target Group(s)" />
    <LANGINFO lang="DE" label="Relevanz für Zielgruppen" />
  </LANGINFOS>
</CMS_INPUT_CHECKBOX>
```

Relevanz für Zielgruppen		
<input type="checkbox"/> Architects	<input checked="" type="checkbox"/> Mobile Users	<input type="checkbox"/> Registered
<input type="checkbox"/> German Visitors	<input type="checkbox"/> Homebuilders	

**Abbildung 4-1: Darstellung der Segmente in Form von Checkboxes**

Ein weiteres Beispiel ist im Tutorial in Kapitel 7 beschrieben.

Nähere Informationen zur Verwendung des Datenelements `CMS_INPUT_OPTIONS` finden Sie in der Online Dokumentation von FirstSpirit unter dem Pfad `Vorlagenentwicklung/Formulare/Datenelemente/OPTIONS/PUBLIC`.

## 4.2 ContentTargeting-Tag

Nach einer Generierung sind zunächst immer alle durch die Redakteure erstellten Inhaltsvarianten in der JSP-Seite auf dem Server enthalten. Erst bei der Anforderung der Webseite durch den Besucher erfolgt eine zielgruppenspezifische Filterung, so dass dem Besucher nur die für ihn relevanten Inhalte angezeigt werden.

Diese Filterung erfolgt über die Einbindung einer Tag-Library und des von ihr bereitgestellten `ContentTargeting`-Tags, für dessen Verwendung ein Präfix gewählt werden muss. Die Tag-Library wird bei der Konfiguration des Projekts automatisch in der `web.xml` definiert:



```
<jsp-config>
  <taglib>
    <taglib-uri>targeting</taglib-uri>
    <taglib-location>/WEB-INF/targeting.tld</taglib-location>
  </taglib>
</jsp-config>
```

Die Beschreibung des Präfix und des ContentTargeting-Tags erfolgt in den folgenden Unterkapiteln:

- Präfix siehe Kapitel 4.2.1, Seite 31
- ContentTargeting-Tag siehe Kapitel 4.2.2, Seite 32

#### 4.2.1 Präfix

Für den Einsatz des *Real-time Targeting*-Moduls muss zunächst die Dateiendung des verwendeten Seiten-Templates von *html* zu *jsp* geändert werden. Des Weiteren muss die in der *web.xml* der Web-Applikation definierte Tag-Library (vgl. *Kapitel 4.2, Seite 30*) innerhalb dieses Seiten-Templates eingebunden werden.

*Nähere Informationen zur Änderung der Dateiendung können Sie dem FirstSpirit Handbuch für Entwickler entnehmen.*

Für die Nutzung des ContentTargeting-Tags wird in dieser Dokumentation nachfolgend das Präfix *rtt* verwendet.

#### Beispiel für die Einbindung der Taglib in JSP-Seiten

```
<%@ page language="java" contentType="text/html; charset=UTF-8" %>
<%@ taglib uri="targeting" prefix="rtt" %>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
```



Während das verwendete Präfix (*rtt*) beliebig gewählt werden kann, darf der URI nicht verändert werden.



Wird das Präfix "*rtt*" geändert, so ist der neue Wert ebenfalls für die einzelnen Tags zu verwenden – d.h. "*<myPrefix:showContent>*" anstelle von "*<rtt:showContent>*".



#### 4.2.2 <rtt:showContent>

Das Tag <rtt:showContent> dient zur Ein- bzw. Ausblendung des von ihm umschlossenen Inhalts. Die Sichtbarkeit des Inhalts wird über die Zugehörigkeit des Besuchers zu den definierten Segmenten bzw. über die Angabe von Zeitpunkten geregelt. Die Auswertung des Tags erfolgt über einen Vergleich der Segmente des darzustellenden Inhalts und des Webseiten-Besuchers. Nur wenn diese übereinstimmen, wird der Inhalt dem Besucher bereitgestellt.

Die Sichtbarkeit des vom Tag umschlossenen Inhalts ist außerdem abhängig von der konfigurierten Logik. Nach dieser muss sich der Besucher entweder in allen (AND) oder in mindestens einem (OR) der definierten Segmente befinden. Ist er in keinem Segment enthalten, so wird der Inhalt des Tags ausgegeben, für das kein Segment definiert wurde. Im Fall der Zugehörigkeit zu mehreren Segmenten wird der erste Inhalt ausgegeben, für den die definierte Regel erfüllt ist.

Um auf jedes definierte Segment gezielt reagieren zu können, besteht außerdem die Möglichkeit, die auszugebenden Inhaltsvarianten durch frei definierbare IDs beliebig zu gruppieren.



*Es empfiehlt sich, immer auch einen keinem Segment zugeordneten Inhalt anzulegen, um so auch Besuchern ohne Segment-Zugehörigkeit etwas anzuzeigen.*

*Dieser Inhalt **muss** immer als letztes ausgegeben werden, so dass zunächst die zielgruppenspezifischen Inhaltsvarianten geprüft werden.*

Über die Angabe von Start- und/oder Endzeitpunkten können die Inhalte außerdem zeitabhängig angezeigt werden, wodurch unterschiedlichste Szenarien denkbar sind. So lassen sich beispielsweise für ein Segment in verschiedenen Zeiträumen unterschiedliche Inhalte anzeigen.

Im ContentCreator können Redakteure über einen Zeitstrahl ein Datum auswählen. Auf diesem Weg wird ihnen auch die Betrachtung der Inhalte zu einem bestimmten – vergangenen oder zukünftigen – Zeitpunkt ermöglicht. Für die Berücksichtigung des Datums muss dieses jedoch als Attribut übergeben werden.

Die folgende Tabelle enthält eine Übersicht aller verfügbaren Attribute:



Attribut	Bedeutung
segments	die komma-separierte Liste der Segmente, für die der Inhalt sichtbar sein soll
logic	auf die Segmente anzuwendender Operator (AND oder OR, default: AND)
id	Eindeutiger Bezeichner für eine gemeinsam auszuwertende Gruppe von Inhaltsvarianten. Sobald ein Inhalt der Gruppe ausgegeben wurde, wird danach kein weiterer ausgegeben. (default: defaultID)
startdate	Startdatum, ab wann der Inhalt angezeigt werden soll
enddate	Enddatum, bis wann der Inhalt angezeigt werden soll
referencedate	Das Datum, das im ContentCreator über den Zeitstrahl ausgewählt wurde, kann über folgenden Aufruf abgefragt werden:  <pre>String.valueOf(((java.util.Date) session.getAttribute("fs.preview.#time")).getTime())</pre> Im SiteArchitect lässt sich eine ähnliche Abfrage über <code>#global.startTime</code> simulieren. Diese liefert jedoch nur das Datum der letzten Änderung zurück.

### Beispiel für die Verwendung des ContentTargeting-Tags:

```
<rtt:showContent
  segments="Architects, Homebuilders" logic="OR" id="300642"
  startdate="1336226501635" enddate="1336226503635"
  referencedate="1336226502635">

[... anzuzeigender Inhalt ...]

</rtt:showContent>
```

Ein ausführlicheres Beispiel finden Sie in Kapitel 5, Seite 37.



### 4.3 Persona-Simulator

Durch die in Kapitel 3 beschriebene Installation und Konfiguration des *Real-time Targeting*-Moduls werden drei Formatvorlagen in den TemplateStore importiert. Sie müssen über einen CMS\_RENDER-Aufruf in der verwendeten Seitenvorlage referenziert werden.

Die Formatvorlage *Persona Box* stellt den *Persona-Simulator* in beiden Clients zur Verfügung. Im ContentCreator wird er in der Preview Bar ergänzt. Da diese jedoch in der Vorschau des SiteArchitects nicht existiert, wird dort über die Formatvorlage *Persona Preview Bar* ein Äquivalent eingefügt. Über die Formatvorlage *Persona Skript* wird zusätzlich benötigtes CSS und JavaScript bereitgestellt.

```
[...]  
<head>  
$CMS_RENDER(template:"persona_script")$  
</head>  
$CMS_RENDER(template:"persona_previewbar")$  
$CMS_RENDER(template:"persona_box")$  
[...]
```



Das JavaScript verwendet die Bibliothek [jQuery](#).

*Es ist zwingend erforderlich, dass jQuery in der Version 1.9.1 oder höher in der richtigen Seitenvorlage eingebunden wird. Andernfalls kann der Persona-Simulator nicht genutzt werden und es wird stattdessen zu Fehlermeldungen in der JavaScript-Konsole des Browsers kommen.*

Des Weiteren wird auf der Basis eines in den TemplateStore importierten Schemas im ContentStore die leere Datenquelle *Personas* angelegt. Sie wird vom Persona-Simulator dazu verwendet, in der Vorschau und im ContentCreator die verschiedenen Segmente zu repräsentieren und den Redakteuren eine zielgruppenspezifische Sicht auf die Webseite zu ermöglichen. Hierfür müssen in der Datenquelle entsprechende Personas erzeugt werden.

In jedem neuen Datensatz muss ein Name angegeben und ein Bild ausgewählt werden. Außerdem ist durch die Auswahl mindestens einer Checkbox anzugeben, welches Segment durch die Persona repräsentiert werden soll (*siehe Abbildung 4-2*).

Die Bereitstellung der Segmente in Form der Checkboxen erfolgt automatisch über die Verwendung des WoopraValueProviders (*vgl. Kapitel 4.1, Seite 29*).



Abbildung 4-2: Persona

Die in der Datenquelle erzeugten Personas können anschließend sowohl in der Vorschau als auch im ContentCreator über den Persona-Simulator ausgewählt werden (vgl. Abbildung 4-3). Enthält die Seite spezifische Inhalte für das durch die ausgewählte Persona repräsentierte Segment, so werden diese Inhalte angezeigt.

 Für die korrekte Darstellung der spezifischen Inhalte wird die Durchführung der in Kapitel 4.2 beschriebenen Schritte vorausgesetzt.

Abbildung 4-3: Persona-Simulator

Im Fall von Überschneidungen bei der Repräsentation mehrerer Segmente wird nur der erste zutreffende Inhalt ausgegeben. Enthält die Seite lediglich spezifische



Inhalte, die nicht zu den durch die Persona repräsentierten Segmenten passen, wird nur der Standardinhalt der Seite dargestellt.

Durch den Persona-Simulator wird den Redakteuren so die Möglichkeit geboten, die Zuordnung der von ihnen erstellten zielgruppenspezifischen Inhaltsvarianten in der Vorschau und im ContentCreator zu überprüfen.

Ohne den Simulator würde den Redakteuren immer nur der Standardinhalt der Seite angezeigt. Eine zielgruppenspezifische Darstellung wäre in diesem Fall erst im Live-Stand auf der Webseite möglich.

#### 4.4 Tracking-Code

Woopra™ ermittelt das Verhalten der Webseiten-Besucher über einen Tracking-Code. Dieser wurde in Form einer Formatvorlage während der in Kapitel 3 beschriebenen Installation und Konfiguration des Real-time Targeting-Moduls in das FirstSpirit-Projekt importiert. Dort muss sie lediglich noch mit einem CMS\_RENDER-Aufruf in die verwendete Seitenvorlage eingebunden werden.

```
$CMS_RENDER(template:"woopra_tracking", woopra_domain:"http://website")$
```

Über den Parameter *woopra\_domain* wird die bei Woopra hinterlegte Domain der zu trackenden Webseite übergeben.

Eine detailliertere Erläuterung des Tracking-Codes können Sie der Woopra™-Dokumentation entnehmen:

<https://www.woopra.com/docs/setup/javascript-tracking/>



## 5 Verwendung in FirstSpirit

Die Funktionalitäten des *Real-time Targeting*-Moduls können auf verschiedenste Weisen im FirstSpirit-Projekt bereitgestellt werden. Die jeweils optimale Umsetzung muss daher immer individuell auf Basis der bestehenden projektspezifischen Anforderungen ermittelt werden.

Die folgenden Unterkapitel skizzieren die Erzeugung zielgruppenspezifischer Inhalte mithilfe einer FS\_LIST sowie einen zusätzlichen Anwendungsfall für die Verwendung des *Real-time Targeting*-Moduls.

### 5.1 FS\_LIST

An dieser Stelle wird beispielhaft die Verwendung der *Real-time Targeting*-Funktionalitäten in Verbindung mit einer FS\_LIST skizziert, die in diesem Fall in einem Absatztemplate verwendet wird.

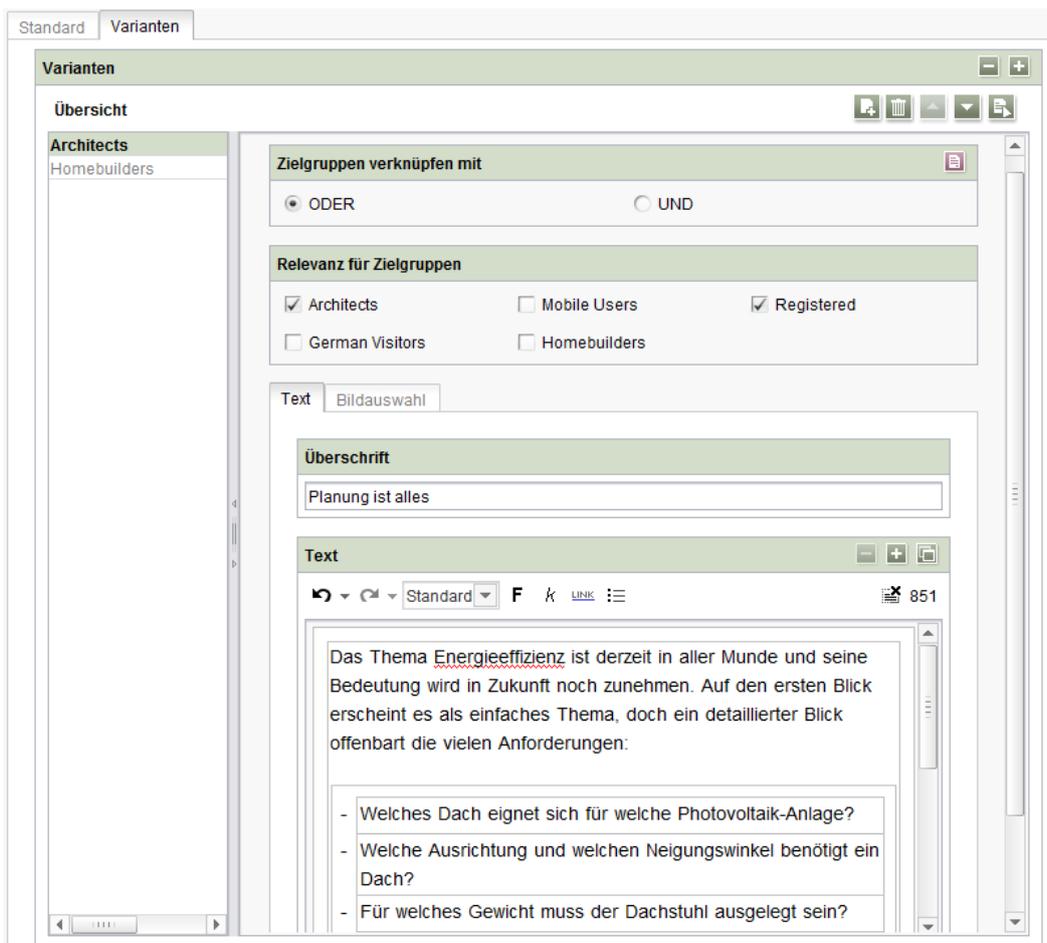


Abbildung 5-1: FS\_LIST



In diesem Beispiel entspricht jede zielgruppenspezifische Inhaltsvariante einem Eintrag in der FS\_LIST, die hier in der Registerkarte *Varianten* enthalten ist (vgl. *Abbildung 5-1*). Jede dieser Varianten beinhaltet eine Reihe von Checkboxes, welche die im Vorfeld definierten Segmente repräsentieren. Die Ermittlung dieser Segmente geschieht automatisch (siehe auch *Kapitel 4.1, Seite 29*).

Da durch die Verwendung von Checkboxes auch eine Mehrfachauswahl zulässig ist, muss zusätzlich festgelegt werden, in welcher Form die selektierten Segmente mit einander verknüpft sind. Dies wurde in diesem Beispiel über Radiobuttons gelöst (vgl. auch *Tabelle in Kapitel 4.2.2, Seite 32*).

Die Sichtbarkeit der jeweiligen Variante hängt von der Gruppenzugehörigkeit des Besuchers der Webseite ab. Im Fall einer UND-Verknüpfung muss der Besucher in allen über die Checkboxes ausgewählten Segmenten enthalten sein, andernfalls genügt die Existenz in mindestens einem dieser Segmente.



Erfüllt ein Besucher die Sichtbarkeitsregel mehrerer Inhaltsvarianten, so wird ihm nur der **erste** zutreffende Inhalt angezeigt.

Für Besucher, die sich keinem Segment zuordnen lassen, sollte immer auch ein Standardtext bereitgestellt werden, welcher in diesem Beispiel in der Registerkarte *Standard* enthalten ist.

Die zielgruppenspezifische Ausgabe der in der FS\_LIST enthaltenen Varianten wird über das ContentTargeting-Tag geregelt, welches im HTML-Kanal des zugehörigen Templates anzugeben ist (siehe auch *Kapitel 4.2.2, Seite 32*).

```
<rtt:showContent
  segments="$CMS_VALUE(st_segments.toString(", "))$" // Checkboxes
  logic="$CMS_VALUE(st_logic)$" // Radiobuttons
  id="$CMS_VALUE(#global.section.id)$" >
[...]
```

Entsprechend der ihm zugeordneten Segmente werden dem Besucher der Webseite die durch dieses Tag umschlossenen Inhalte angezeigt.

In diesem Beispiel wurde der *Startseite* im Demoprojekt *Mithras Energy* ein zielgruppenspezifischer Absatz hinzugefügt. Dieser besitzt laut der in *Abbildung 5-1* sichtbaren Definition neben einem allgemeinen Text (vgl. *Abbildung 5-2*) unter anderem eine Variante für *Architects* (vgl. *Abbildung 5-3*).



**Überschrift**  
Sonnenergie - Stichwort der Zukunft

**Text**

**Willkommen bei Mithras Energy!**

**Sonnenergie - Stichwort der Zukunft**

Sonnenergie ist die kostengünstige und moderne Alternative zu fossilen Brennstoffen. Auch in den dunklen Jahreszeiten oder bei bedecktem Himmel liefert die Sonne ihre Energie tagtäglich frei Haus. Somit ist sie jederzeit in unerschöpflichem Maße vorhanden. Sie müssen sie nur nutzen.

Bereits seit 2009 ist es gesetzlich vorgeschrieben, die Energieversorgung in Neubauten anteilig über erneuerbare Energien abzudecken. Doch auch ältere Häuser lassen sich problemlos modernisieren und so auf die Anforderungen von morgen vorbereiten.

Informieren Sie sich noch heute über unsere Produkte oder vereinbaren Sie ein Beratungsgespräch, damit auch Sie in der Zukunft profitieren können.

Abbildung 5-2: Allgemeine Ansicht des Absatzes

**Varianten**

**Übersicht**

**Architects**  
Homebuilders

**Zielgruppen verknüpfen mit**

ODER  UND

**Relevanz für Zielgruppen**

Architects  
 German Visitors

**Text** Bildauswahl

**Überschrift**  
Planung ist alles

**Text**

**Willkommen bei Mithras Energy!**

**Planung ist alles**

Das Thema Energieeffizienz ist derzeit in aller Munde und seine Bedeutung wird in Zukunft noch zunehmen. Auf den ersten Blick erscheint es als einfaches Thema, doch ein detaillierter Blick offenbart die vielen Anforderungen:

- Welches Dach eignet sich für welche Photovoltaik-Anlage?
- Welche Ausrichtung und welchen Neigungswinkel benötigt ein Dach?
- Für welches Gewicht muss der Dachstuhl ausgelegt sein?
- Wie wird die Installation durchgeführt? Ist eine Integration in das Dach möglich?
- Welche Kosten lassen sich bei einem Neubau einsparen?
- Wie ist eine maximale Energieausbeute erreichbar?
- Mit welchen Anforderungen ist eine Aufschaltung der Anlage verbunden?

Gerne beantworten wir Ihnen diese und weitere Fragen und unterstützen Sie bei der individuellen Planung zu Ihrem Projekt.

Kontaktieren Sie uns und vereinbaren Sie ein Beratungsgespräch.

Abbildung 5-3: Zielgruppenspezifische Ansicht des Absatzes für Architekten



## 5.2 Erweiterte Anwendungsfälle

Zusätzlich zur Bereitstellung zielgruppenspezifischer Inhalte auf der Webseite sind weitere Anwendungsfälle denkbar, die sich auf Basis der erfassten Informationen der Webseiten-Besucher umsetzen lassen.

In den nachfolgenden Unterkapiteln werden stellvertretend einige solcher Anwendungsfälle beschrieben.

### 5.2.1 Smart Forms

Mit den Funktionalitäten des *Real-Time Targetings* kann die Verwendung sogenannter *Smart Forms* umgesetzt werden. Dabei handelt es sich um Formulare, deren Felder beim Aufruf mit bereits bekannten Daten gefüllt sind und somit nur noch bei Bedarf editiert werden müssen. Die entsprechenden Daten werden beim ersten Versand eines Formulars erfasst und können danach für jedes andere Formular der Seite genutzt werden.

Auf diesem Weg werden für den Besucher der Webseite redundante Eingaben vermieden, was eine Verbesserung der Usability darstellt.



*Für das nachfolgende Beispiel wird vorausgesetzt, dass das Real-time Targeting-Modul entsprechend der beschriebenen Schritte in Kapitel 3 und 4 installiert und konfiguriert wurde.*

An dieser Stelle wird beispielhaft die Verwendung von *Smart Forms* anhand des FirstSpirit-Moduls *FormEdit* in Verbindung mit den Funktionalitäten des *Real-time Targetings* skizziert. Der Nutzung des Moduls *FormEdit* ist natürlich keine Pflicht. Es kann auch jede andere Art von Formularen eingesetzt werden.

Ein mit *FormEdit* erstelltes Formular setzt sich aus den drei Teilen *Anfang*, *Block* und *Ende* zusammen, die durch Absatztemplates repräsentiert werden und dem Projekt bereits mit der Installation des *FormEdit*-Moduls hinzugefügt wurden.

In diesem Beispiel sollen nur der Name und die E-Mail-Adresse eines Besuchers vorausgefüllt werden. Zur initialen Erfassung dieser Daten muss der HTML-Bereich des *Anfangs* durch den folgenden Skript-Code ergänzt werden, der eine entsprechende Anpassung erfordert, wenn weitere Daten gespeichert werden sollen:



```
<script>
  var submit_$CMS_VALUE(fr_st_name)$ = false;
  $('#$CMS_VALUE(fr_st_name)$').submit(function(e) {
    var formName = $('#name').val();
    var formEmail = $('#email').val();
    woopra.identify({
      email: formEmail,
      name: formName
    }).push();
    setTimeout(function() {
      submit_$CMS_VALUE(fr_st_name)$ = true;
      $('#$CMS_VALUE(fr_st_name)$').submit();
    }, 2000);
    if(!submit_$CMS_VALUE(fr_st_name)$) {
      e.preventDefault();
    } });
</script>
```

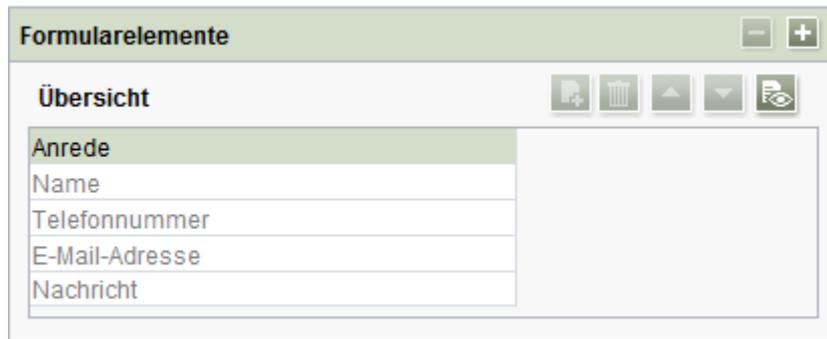
Dieser Code liest die Eingabe des Besuchers aus dem initial verwendeten Formular aus und übermittelt sie in diesem Fall an Woopra™ (siehe Kapitel 2.4, Seite 13). Dort werden die Daten als Attribute des Besuchers gespeichert und können bei Bedarf jederzeit abgefragt werden.

Die einzelnen Felder, aus denen sich das Formular zusammensetzt, sind über eine FS\_LIST innerhalb des *Blocks* zu definieren (vgl. *Abbildung 5-4*). Sie werden durch weitere Absatztemplates repräsentiert, die ebenfalls mit der Installation des *FormEdit*-Moduls bereitgestellt werden.



Für die einwandfreie Funktionalität des zuvor genannten Skripts müssen die Felder „Name“ und „E-Mail-Adresse“ **aller** verwendeten Formulare zwingend die eindeutigen Bezeichner *name* und *email* besitzen.





**Abbildung 5-4: FormEdit – Formular-Block**

Für alle Texteingaben, wie beispielsweise auch der Name und die E-Mail-Adresse eines Besuchers, wird das Formularelement „Text“ verwendet. Um diese Felder in anderen Formularen mit bereits erfassten Daten vorzubefüllen, muss die If-Abfrage für die Eingabekomponente *st\_value* innerhalb des HTML-Bereichs des zugehörigen Absatztemplates um einen Else-Fall ergänzt werden:

```

$CMS_SET(fr_st_value)$
  $CMS_IF(!st_value.isEmpty)$
    $CMS_VALUE(st_value)$
  $CMS_ELSE$
    <c:out value="\${sessionScope['rtt.woopra.user.$CMS_VALUE(st_name)$']}" />
  $CMS_END_IF$
$CMS_END_SET$

```

Des Weiteren muss innerhalb der Projekteigenschaften für die entsprechende Web-Komponente die *web.xml* der Woopra Web-Applikation angepasst werden, indem ihr der Pipeline-Filter *GetWoopraUserInformationFilter* hinzugefügt wird (siehe Kapitel 3.5.2.4, Seite 25). Dieser Filter ermöglicht die Ermittlung der in Woopra™ zu einem Besucher gespeicherten Attribute.

Da diese Attribute initial noch nicht erfasst sind, werden dem Besucher der Webseite zunächst nur leere Formulare angezeigt, die entsprechend der Definition in der FS\_LIST des Blocks strukturiert sind (vgl. *Abbildung 5-5* und *Abbildung 5-6*).



## Kontaktieren Sie uns

Sie wünschen ein Beratungsgespräch? Wir stehen Ihnen gerne mit Rat und Tat zur Seite.

**Beantragen Sie eine Kontaktaufnahme und wir melden uns schnellstmöglich bei Ihnen.**

Anrede \*  Herr  Frau

Name \* Müller

Telefonnummer \* 01723364700

E-Mail-Adresse \* müller@developers.de

Gewünschte Art der Kontaktaufnahme \*  Telefonisch  Schriftlich per E-Mail

Nachricht \* Um Rückruf wird gebeten.

1F9FT9

1F 9F 79 Neu laden

Bitte geben Sie die Buchstaben und Zahlen ein. \*

Löschen

Senden

**Abbildung 5-5: Formular mit Besucherdaten, die an Woopra™ übermittelt werden**

Erst nach der Eingabe der entsprechenden Daten und dem Versand eines solchen Formulars werden die Daten in Woopra™ übernommen und dort gespeichert (vgl. Abbildung 5-6).



Close

Start Chat

### Profile Stats

First Seen Dec 3, 2013  
 Direct or local bookmark

Last Seen 2 minutes ago

Last 90 Days 19 minutes  
 3 visits  
 10 actions

### Profile Info

Name Müller

Email müller@developers.de

Company

Gender

Loginname

**Abbildung 5-6: Woopra™ - gespeicherte Besucherdaten**

Zusätzlich zu den übermittelten Daten werden die Aktivitäten des Besuchers auf der Seite sowie der Zeitpunkt seines letzten Besuches erfasst.

Bei dem Aufruf eines beliebigen anderen Formulars, dessen Felder „Name“ und „E-Mail-Adresse“ die vorgeschriebenen eindeutigen Bezeichner besitzen, werden nun die in Woopra™ gespeicherten Besucherdaten ermittelt und in das Formular geschrieben (vgl. *Abbildung 5-7*).



## Kontakt

Die Website mithrasenergy.de ist ein Demoprojekt der e-Spirit AG, das den Einsatz des Content Management Systems FirstSpirit zeigt. Sie können gerne Kontakt mit uns aufnehmen, wenn Sie Fragen zu diesem Demoprojekt haben.

Anrede \*  Herr  Frau

Firma \*

Name \* Müller

Ort

Telefon

Fax

E-Mail \* müller@developers.de

Produkt

Ihr Anliegen

Bitte geben Sie die folgende Zeichen- und Zahlenkombination (Groß- und Kleinschreibung beachten) ein

**RSTJ19**

Neu laden

Bitte füllen Sie alle Felder aus \*

**Abbildung 5-7: Vorbefüllung eines zweiten Formulars**

Der Besucher besitzt in diesem Beispiel die Möglichkeit, die im Formular angezeigten Daten zu editieren. In anderen Szenarien wäre es jedoch beispielsweise auch denkbar, sie ihm nur noch anzuzeigen oder das gesamte Formular bei der Verfügbarkeit aller erforderlichen Daten auszublenden.

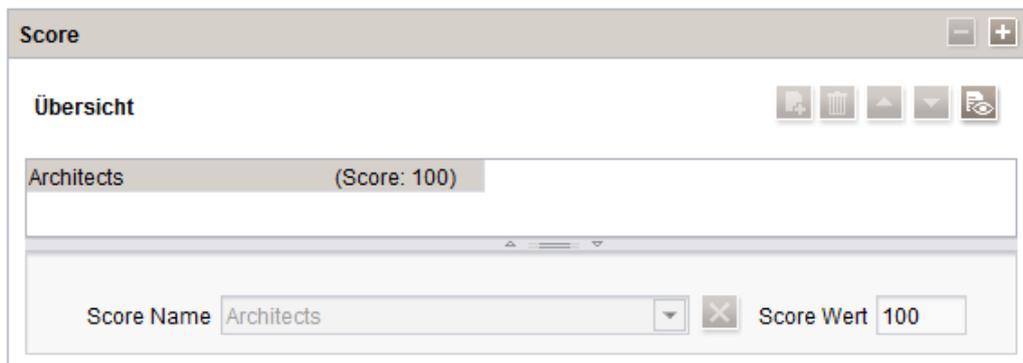


## 5.2.2 Scoring

Wie bereits in Kapitel 1 unter dem Punkt 2c beschrieben, wird das Interesse eines Webseiten-Besuchers für spezifische Themen in der Regel mit einem Scoring-Mechanismus erfasst. Der Besucher erhält für den Aufruf bestimmter Inhalte eine zuvor festgelegte Anzahl von Punkten, die an den Profiling & Segmentation Service übertragen und von diesem summiert werden. Erreicht diese Summe einen definierten Schwellwert, erfolgt eine automatische Zuordnung zu einem entsprechenden Segment.

An dieser Stelle wird die Verwendung des Scorings in Verbindung mit Datensätzen skizziert. Der Mechanismus kann jedoch auch für andere FirstSpirit-Elemente eingesetzt werden.

In diesem Beispiel sollen Webseiten-Besucher durch das Lesen bestimmter Inhalte als Architekten oder Bauherren identifiziert werden. Dafür wurden die Datensätze um eine FS\_LIST ergänzt, deren Einträge eine Auswahlmöglichkeit für die beiden Segmente (Architekten, Bauherren) und ein Feld für die Anzahl der zu vergebenden Punkte enthalten (vgl. *Abbildung 5-8*). Die Redakteure können den Datensätzen auf diesem Weg unterschiedliche Score Werte für die beiden Segmente zuordnen.



**Abbildung 5-8: Definition eines Score Werts für Architekten**

Die in der FS\_LIST hinterlegten Segmente werden zusammen mit den für sie definierten Score Werten in den PageContext übertragen. Dafür wurde die folgende Zeile in den HTML-Bereich der zugehörigen Tabellenvorlage eingefügt:

```
$CMS_SET(#global.pageContext["set_scoring"],cs_scoring)$
```

Der PageContext wird seinerseits über die folgende Abfrage, die im HTML-Bereich der Seiten-Vorlage hinzuzufügen ist, ausgewertet:



```

$CMS_IF(!#global.preview)$
  $CMS_IF(#global.pageContext["set_scoring"] != null)$
    $CMS_SET(scoring, #global.pageContext["set_scoring"])$
  $CMS_END_IF$
$CMS_END_IF$

```

Der zu erreichende Schwellwert und die aus ihm folgende Zuordnung sind im Profiling & Segmentation Service hinterlegt. Dafür wurde in den *Einstellungen* des vom *Real-time Targeting*-Modul verwendeten Services Woopra™ unter dem Punkt *Schema* zunächst ein neues *Action Data Schema* hinzugefügt und dieses durch die *Properties* für Architekten und Bauherren ergänzt (vgl. *Abbildung 5-9*).

Weitere Informationen zur Konfiguration des Action Data Schemas finden Sie in der Woopra™-Dokumentation:

<https://www.woopra.com/docs/manual/configure-schema/>

The screenshot displays the Woopra configuration interface for an Action Data Schema. The interface is divided into several sections:

- Event Info:**
  - Event Name:
  - Display Name:
  - Description:
- Event Properties:**

Property	Key	Type	Aggregate	Value
	architect_score	Number	Amount	1,000
	homebuilder_sc	Number	Amount	1,000

+ Add Property
- Event Template:**

Template: `**$(visitor.name)** triggered content scoring for Architect ${action.architect_score} and Homebuilder ${action.homebuilder_score} and Technicians ${action.technicians_score}`

At the bottom, there are buttons for 'Delete', 'Cancel', and 'Save'.

Abbildung 5-9: Woopra - Action Data Schema



Die *Properties* können dann bei der Konfiguration eines neuen Segments für die Definition des Schwellwertes verwendet werden.

In diesem Beispiel soll ein Webseiten-Besucher, der einen Score Wert von 1000 für Architekten erreicht hat, automatisch als ein eben solcher identifiziert und dem Segment *Architekten* zugeordnet werden (vgl. *Abbildung 5-10*). Für Bauherren gilt diese Anforderung äquivalent.

Name

Shorter label names are recommended (e.g. Search instead of Search Traffic or Prospect instead of Frequent Visitor)

Color

Visitors People who are + and did Scoring \* +

Cancel Apply

**Action**

**Aggregation**

**Timeframe**

From  to  day(s) ago

**Visit**

**Abbildung 5-10: Woopra - Scoring für Architekten**

Die *Properties* des *Action Data Schemas* werden auch im Tracking-Code verwendet, mit dem das Verhalten der Webseiten-Besucher ermittelt wird (vgl. auch *Kapitel 4.4, Seite 36*). Der Tracking-Code wurde bereits mit der in *Kapitel 3* beschriebenen Installation in das Projekt importiert und muss für die Übertragung der Score Werte an den Profiling & Segmentation Service um die folgenden Zeilen ergänzt werden:

```
<script>
[...]
$CMS_SET(hasScore, false)$
$CMS_SET(scores, "")$
```



```
$CMS_IF(scoring != null)$
  $CMS_FOR(entry, scoring)$
    $CMS_IF(entry.st_score_value > 0)$CMS_SET(hasScore,
true)$CMS_END_IF$
    $CMS_IF(#for.index > 0)$CMS_SET(scores, scores + ",")$CMS_END_IF$
    $CMS_SET(scores, scores + entry.st_score_name + " : " +
entry.st_score_value)$
  $CMS_END_FOR$
$CMS_END_IF$

woopra.track();

$CMS_IF(hasScore)$
  woopra.track('scoring', {
    $CMS_VALUE(scores)$
  });
$CMS_END_IF$
</script>
```



Bei der Ergänzung des Tracking-Codes muss die Zeile **woopra.track();** beachtet werden. Sie existiert auch am Ende der initial importierten Formatvorlage und muss dort bei der Code-Ergänzung entsprechend entfernt werden, um eine Dopplung zu vermeiden.

Dieser Code weist den in diesem Fall in den Datenquellen hinterlegten Score Wert dem entsprechenden Property in Woopra™ zu, sobald der Datensatz durch einen Webseiten-Besucher betrachtet wird. In Woopra™ erscheint gleichzeitig die Information zu dieser Aktion (vgl. Abbildung 5-11).

 Visitor #3497 triggered content scoring for Architect 100 and Homebuilder 0

Abbildung 5-11 – Identifizierung eines potentiellen Architekten



## 6 Erweiterungen

Standardmäßig verwendet das *Real-time Targeting*-Modul den Profiling & Segmentation Service Woopra™. Falls andere oder weitere Dienste gewünscht sind, können diese jederzeit projektspezifisch zur User-Experience-Pipeline (UXP) hinzugefügt werden. Das Modul besitzt an dieser Stelle eine komplett offene Architektur und ist nicht auf einen spezifischen Dienst beschränkt.

Grundsätzlich sind dabei zwei verschiedene Szenarien denkbar:

### a) Woopra™ wird als Service beibehalten

Die UXP wird durch zusätzliche Filter ergänzt, die weitere Systeme anbinden und ergänzende Informationen über den Webseiten-Besucher bereitstellen. Die Implementierung dieser Filter wird in Kapitel 6.1 beschrieben.

### b) Woopra™ wird durch einen anderen Service ersetzt

Ein solcher Profiling & Segmentation Service muss einige Anforderungen erfüllen, um für den Einsatz mit dem *Real-time Targeting*-Modul geeignet zu sein. Diese Anforderungen werden im Kapitel 6.2 erläutert.

## 6.1 Implementierung zusätzlicher Filter für die UXP

Für die zielgruppenspezifische Ausgabe der Inhalte auf der Webseite müssen die dem Besucher jeweils aktuell zugeordneten Segmente ermittelt werden. Diese Aufgabe übernehmen die Filter, die die entsprechenden Segmente beim Profiling & Segmentation Service anfordern und sie in die aktuelle Session des Besuchers schreiben. Die Informationen werden dann vom ContentTargeting-Tag für die Darstellung der für den Besucher relevanten Inhalte verwendet.

Soll ein anderer Profiling & Segmentation Service verwendet werden, ist ein entsprechender Filter zu implementieren, der die Ermittlung der Segmente übernimmt.

Dafür muss zunächst das *rta-api.jar* in das eigene Projekt eingebunden und eine neue Klasse angelegt werden, die das *AbstractUxpFilter* erweitert:

```
public class ExampleIpFilter extends AbstractUxpFilter
```

Durch diese Erweiterung werden verschiedene Methoden bereitgestellt.

Die Methoden *init()*, *doFilter()* und *destroy()* dienen lediglich der Ablauf-Steuerung des Filters.



Die `run()`-Methode enthält die eigentliche Logik eines Filters. Um Mehrfachaufrufe zu vermeiden, sollte in ihr zunächst über die `shouldFilter()`-Methode ermittelt werden, ob der Filter bereits ausgeführt wurde.

Der implementierte Filter sollte als Web-Komponente eines FirstSpirit-Moduls bereitgestellt werden. Innerhalb der `web.xml` der Web-Komponente kann der Filter dann konfiguriert werden.

Bei der Installation und Konfiguration der zusätzlichen Filter auf der FirstSpirit Vorschau- und Produktiv-Umgebung ist zu beachten, dass diese in der `web.xml` erst nach dem `UxpServletFilter` eingebunden werden.

In den zwei nachfolgenden Unterkapiteln wird beispielhaft die Implementierung zweier eigener Filter beschrieben:

- `ExampleIpFilter` siehe Kapitel 6.1.1.1, Seite 51
- `ExampleLocationFilter` siehe Kapitel 6.1.1.2, Seite 52

Der `ExampleLocationFilter` soll in diesem Beispiel anhand der über den `ExampleIpFilter` ausgelesenen IP eines Besuchers das zugehörige Land ermitteln und dieses in die Session schreiben, um es später mittels JSP auf der Webseite ausgeben zu können.

#### 6.1.1.1 ExampleIpFilter



*Dieses Beispiel ist nicht für den produktiven Einsatz geeignet!*

In der `run()`-Methode des `ExampleIpFilters` wird anhand des Requests die IP-Adresse des derzeitigen Besuchers ermittelt und im aktuellen Request-Kontext unter dem Variablennamen `rtt.user.ip` gespeichert.

```
RequestContext ctx =
((UxpServletRequest) servletRequest).getRequestContext();
String ipAddress = request.getHeader("X-FORWARDED-FOR");
if (ipAddress == null) {
    ipAddress = request.getRemoteAddr();
}
ctx.set("rtt.user.ip", ipAddress);
```



## 6.1.1.2 ExampleLocationFilter



*Dieses Beispiel ist nicht für den produktiven Einsatz geeignet!*

In der `run()`-Methode des `ExampleLocationFilters` wird zunächst die IP-Adresse des derzeitigen Besuchers aus der Variablen `rtt.user.ip` des Request-Kontextes gelesen. Im Anschluss erfolgt eine Ermittlung des zugehörigen Landes mittels des REST-Services von FreeGeoIp. Dieses wird in der aktuellen Session unter dem Variablennamen `rtt.user.country` gespeichert.

```
RequestContext ctx =
((UxpServletRequest)servletRequest).getRequestContext();
String ip = (String) ctx.get("rtt.user.ip");
String country = null;
String line;
BufferedReader in = null;
try {
    in = new BufferedReader( new InputStreamReader(
        new URL("http://freegeoip.net/xml/"+ip).openStream ());
    while ((line = in.readLine ()) != null) {
        if(line.contains("CountryName")) {
            country =
                line.substring(line.indexOf(">")+1,line.indexOf("</CountryName>"));
            break;
        }
    }
    ctx.getRequest().getSession().setAttribute("rtt.user.country", country);
} catch (IOException e) {
    LOGGER.error("A Problem occured while getting the country!");
} finally {
    if (in != null) {
        try {
            in.close();
        } catch (IOException e) {
            LOGGER.error("A Problem occured while reading the geoip service!");
        }
    }
}
}
```



Soll zu Testzwecken eine IP vorgegeben werden, so kann dies über einen manipulierten Request-Header erfolgen, für dessen Wert *X-FORWARDED-FOR* die gewünschte IP vorgegeben wird.

Alternativ können für die Angabe der IP auch Request-Parameter verwendet werden. In diesem Fall muss die URL durch die Parameter *ip* und *key* erweitert werden. Der Wert für den Parameter *key* lautet *GSJqclQ5qXFbXiUaNyEk* und darf nicht verändert werden. Als Wert für den Parameter *ip* ist die gewünschte IP anzugeben.

### 6.1.1.3 Ausgabe im Template

Innerhalb des Templates kann der durch die Filter gesetzte Wert aus der Session beispielweise mittels eines JSTL-Ausdrucks ausgelesen werden:

```
<c:out value='${sessionScope["rtt.user.country"]}'/>
```

## 6.2 Austausch des Profiling & Segmentation Service

Um Woopra™ durch einen anderen Profiling & Segmentation Service ersetzen zu können, muss dieser bestimmte Anforderungen erfüllen:

- **Bereitstellung der definierten Zielgruppen**

Die Lösung des anderen Anbieters muss eine Schnittstelle zur Verfügung stellen, die alle definierten Segmente zurückliefert. Für diese Schnittstelle muss ein *GomIncludeValueProvider* (siehe Kapitel 4.1, Seite 29) implementiert werden. Über diesen Mechanismus können die Redakteure verschiedene Inhaltsvarianten für die unterschiedlichen Segmente definieren.

- **Abfragen in Real-time**

Der Profiling & Segmentation Service des anderen Anbieters muss über eine Schnittstelle verfügen, über die alle Informationen eines Webseiten-Besuchers zum Zeitpunkt eines Requests in Echtzeit (*Real-time*) abgefragt werden können. Zu diesen Informationen zählt insbesondere die Zugehörigkeit zu Segmenten. Die Schnittstelle wird über einen zu implementierenden Filter innerhalb der UXP angesprochen. Die Implementierung zusätzlicher Filter wird in Kapitel 6.1 beschrieben.



- **Tracking-Mechanismus**

Die Lösung des anderen Anbieters muss einen Tracking-Mechanismus besitzen, um die Aktionen des Webseiten-Besuchers ermitteln zu können. Dies kann beispielsweise per JavaScript oder serverseitig erfolgen.

Die Implementierung sollte in Form eines FirstSpirit-Moduls bereitgestellt werden, welches zusätzlich zum *Real-time Targeting*-Modul auf dem FirstSpirit-Server installiert wird. Hier können die Woopra-Komponenten des *Real-time Targeting*-Moduls als Referenz dienen.

### 6.2.1 Bereitstellung der Liste aller Segmente

Die Bereitstellung der Liste aller Segmente für die Redakteure erfolgt über das zu implementierende Interface *GomIncludeValueProvider*, das in der Access-API von FirstSpirit dokumentiert ist:

<http://www.espirit.com/odfs50/dev/de/espirit/firstspirit/access/store/templatestore/gom/GomIncludeValueProvider.html>

Bei dem Interface handelt es sich um eine standardmäßig von FirstSpirit bereitgestellte Schnittstelle, mit der dynamische Wertemengen an bestimmte Eingabekomponenten übergeben werden können (*siehe Kapitel 4.1, Seite 29*).

Es wird empfohlen, die Kommunikation zwischen dem FirstSpirit-Server und dem gewünschten Profiling & Segmentation Service über einen FirstSpirit-Service zu realisieren, der wiederum aus dem *GomIncludeValueProvider* angesprochen wird. Des Weiteren sollte ein Cache für die zurückgelieferten Zielgruppen implementiert werden, um eine ständige Kommunikation zwischen dem Server und dem Profiling & Segmentation Service zu vermeiden.



## 6.2.2 Filter zur Ermittlung der Informationen eines Besuchers

Die zielgruppenspezifische Ausgabe der Webseiten-Inhalte wird über die dem Besucher zugeordneten Segmente geregelt. Diese Segmente werden durch die Filter vom Profiling & Segmentation Service angefordert und an die Session des Besuchers übergeben.

Soll ein anderer Profiling & Segmentation Service verwendet werden, ist ein entsprechender Filter zu implementieren, der die Ermittlung der Segmente übernimmt (*siehe Kapitel 6.1, Seite 50*).

## 6.2.3 Tracking des Besucherverhaltens

Die meisten Anbieter ermitteln das Verhalten des Webseiten-Besuchers über einen Tracking-Code. Dieser muss innerhalb des Projekts in die verwendete Seitenvorlage eingebunden werden.

Es wird die Implementierung einer Projektkomponente empfohlen, die eine den Tracking-Code enthaltende Formatvorlage in das Projekt importiert. Diese muss dann lediglich über einen CMS\_RENDER-Aufruf in der Seitenvorlage referenziert werden.



## 7 Tutorial

Dieses Tutorial beschreibt beispielhaft die notwendigen Schritte, um die *Real-time Targeting*-Komponenten in ein bestehendes FirstSpirit-Projekt einzubinden.

Zu diesem Zweck wird ein Beispielprojekt mitgeliefert, das im ersten Schritt auf dem FirstSpirit-Server installiert werden muss. Ein Projekt, in dem alle in diesem Tutorial beschriebenen Schritte bereits durchgeführt wurden, liegt ebenfalls bei.

Im zweiten Schritt muss das *Real-time Targeting*-Modul auf dem FirstSpirit-Server installiert und sowohl für den Server als auch für das importierte Projekt konfiguriert werden. Danach müssen die Komponenten für die Vorschau und den ContentCreator installiert werden (*siehe Kapitel 3, Seite 14*).

Sind diese Schritte durchgeführt worden, können in den nächsten Schritten die Vorlagen des Projekts angepasst werden.

### 7.1 Seitenvorlage

Die Seitenvorlage besteht aus einem festen Header sowie zwei festen Teaser-Absätzen. Vor den Teaser-Absätzen liegt der Body-Bereich der Seite (contentcenter). Dieser wird in diesem Beispiel um einen dynamischen Absatz erweitert, der je nach Segmenten des Benutzers andere Inhalte darstellen soll. Danach folgt ein weiterer Body-Bereich, dem im weiteren Verlauf ein Kontaktformular mit *SmartForms* hinzugefügt wird.

#### 7.1.1 Taglibs einbinden

Um die gewünschte Funktionalität zu erreichen, sind zunächst die Taglibraries in den Header der Seitenvorlage einzubinden.

```
<head>
[... ]
<%@taglib prefix="rtt" uri="/WEB-INF/targeting.tld"%>
<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
[... ]
</head>
```

#### 7.1.2 CSS und JavaScript einbinden

Des Weiteren müssen in das Seitentemplate die CSS- und JavaScript-Dateien des



*Real-time Targeting*-Moduls eingebunden werden. Hier bietet es sich an, die Dateien nur in die Vorschau einzubinden, da sie nur dort benutzt werden. Die Formatvorlage *persona\_script* beinhaltet die eigentlichen Aufrufe.

```
<head>
[... ]
$CMS_IF(#global.preview)$CMS_RENDER(template:"persona_script")$CMS_END_IF$
[... ]
</head>
```

### 7.1.3 Previewleiste und Personas einbinden

Die Previewleiste dient der Auswahlmöglichkeit von Personas im SiteArchitect. Sie simuliert dazu die Leiste des MultiPerspectivePreviews aus dem ContentCreator und sollte daher auch nur in der Vorschau des SiteArchitects dargestellt werden. Die Darstellung der Personas in der Vorschau übernimmt die Formatvorlage *persona\_box*, sowohl für den ContentCreator, als auch für den SiteArchitect.

```
<body>
[... ]
$CMS_IF(#global.preview && !#global.is("WEBEDIT"))$
  $CMS_RENDER(template:"persona_previewbar")$
$CMS_END_IF$
$CMS_IF(#global.preview)$
  $CMS_RENDER(template:"persona_box")$
$CMS_END_IF$
[... ]
</body>
```

### 7.1.4 Personas anlegen

In den Datenquellen sind in der Tabelle *Persona* die gewünschten Personas zu pflegen z.B.:



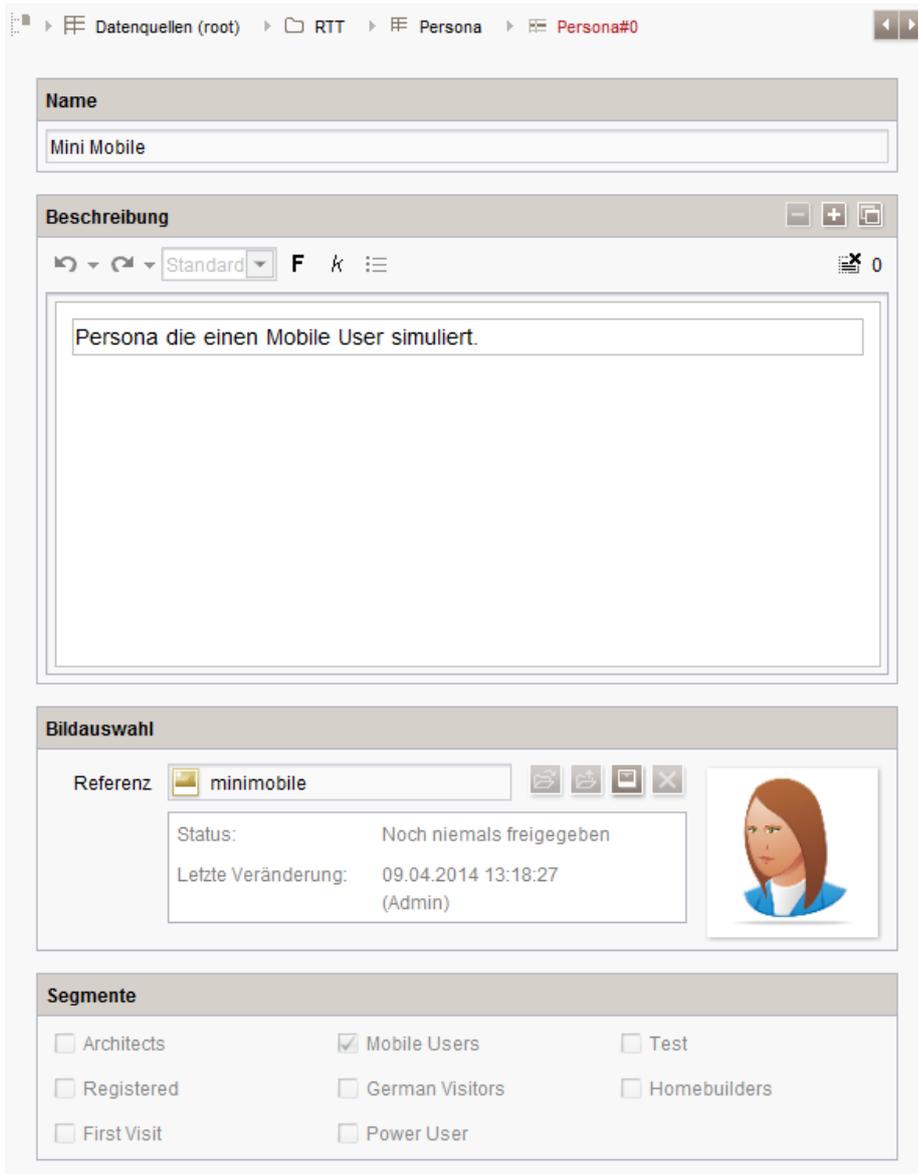


Abbildung 7-1: Neue Persona anlegen

Die Personas sind danach direkt in der Vorschau bzw. im ContentCreator sichtbar.

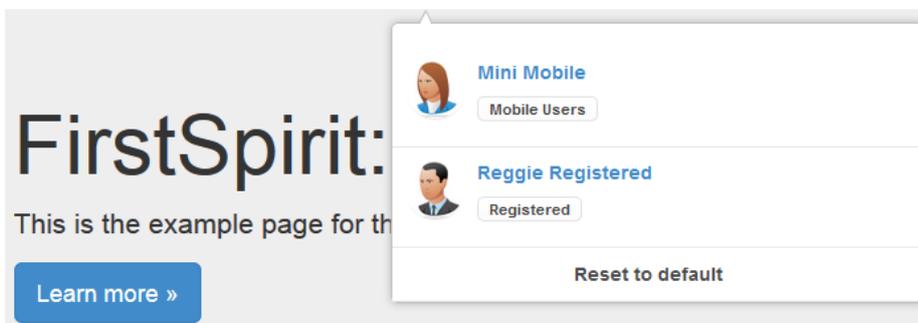


Abbildung 7-2: Persona-Simulator



## 7.2 Absatztemplates

Damit sich die erstellten Personas auch auf einen Absatz auswirken, kann ein neuer oder ein bestehender Absatz um die *Real-time Targeting*-Funktionalität erweitert werden.

### 7.2.1 Teaser Absatz

Da im Beispielprojekt noch kein Absatz existiert, legen wir einen neuen Absatz *Teaser* an und erstellen die benötigten Eingabekomponenten für die Überschrift und den Text.

*Real-time Targeting*-spezifisch kommen noch zwei Eingabekomponenten für den Gültigkeitszeitraum sowie eine Liste für die Varianten hinzu:

```
<CMS_INPUT_DATE name="st_startdate">
<LANGINFOS>
  <LANGINFO lang="*" label="Startdate"/>
  <LANGINFO lang="DE" label="Startdatum"/>
</LANGINFOS>
</CMS_INPUT_DATE>

<CMS_INPUT_DATE name="st_enddate">
<LANGINFOS>
  <LANGINFO lang="*" label="Enddate"/>
  <LANGINFO lang="DE" label="Enddatum"/>
</LANGINFOS>
</CMS_INPUT_DATE>

<FS_LIST name="st_rtt_sectionlist" hFill="yes">
<DATASOURCE type="inline" useLanguages="no">
  <LABELS>
    <LABEL lang="*">
      if (#item.st_targetSegments.empty
        , if (#item.st_startdate.empty
          , if (#item.st_enddate.empty
            , "New variant"
            , "Valid to " + #item.st_enddate.format("MM/dd/yyyy")
          )
          , "Valid from " + #item.st_startdate.format("MM/dd/yyyy")
          + if(!#item.st_enddate.empty, " to "
            + #item.st_enddate.format("MM/dd/yyyy")
          )
        , #item.st_targetSegments.map(x ->
```



```

        x.getLabel("%lang%")).toString(" " +

        #item.st_targetSegmentLogic.getLabel("%lang%") + " ") +
        if(!#item.st_startdate.empty, ", Valid from " +
            #item.st_startdate.format("MM/dd/yyyy")) +
        if(!#item.st_enddate.empty, if(#item.st_startdate.empty, ",
            Valid") + " to " + #item.st_enddate.format("MM/dd/yyyy"))
    </LABEL>
<LABEL lang="DE">
    if (#item.st_targetSegments.empty
        , if (#item.st_startdate.empty
            , if (#item.st_enddate.empty
                , "Neue Variante"
                , "Gültig bis " + #item.st_enddate.format("dd.MM.yyyy"))
                , "Gültig von " + #item.st_startdate.format("dd.MM.yyyy") +
                if(!#item.st_enddate.empty, " bis " +
                    #item.st_enddate.format("dd.MM.yyyy"))
                , #item.st_targetSegments.map(x ->
                    x.getLabel("%lang%")).toString(" " +
                    #item.st_targetSegmentLogic.getLabel("%lang%") + " ") +
                    if(!#item.st_startdate.empty, ",
                    Gültig von " + #item.st_startdate.format("dd.MM.yyyy")) +
                    if(!#item.st_enddate.empty,
                    if(#item.st_startdate.empty, ", Gültig") + " bis " +
                    #item.st_enddate.format("dd.MM.yyyy"))
            )
        )
    </LABEL>
</LABELS>
<ACTIONS>
    <ACTION name="ADD"/>
    <ACTION name="REMOVE"/>
    <ACTION name="UP"/>
    <ACTION name="DOWN"/>
    <ACTION name="EDIT"/>
</ACTIONS>
<COLUMNS>
    <COLUMN show="no">#identifier</COLUMN>
    <COLUMN show="yes" width="150">#text</COLUMN>
</COLUMNS>
<LAYOUT>
    <ADD component="stackedview" constraint="hide"/>
    <ADD component="toolbar" constraint="top"/>
    <ADD component="overview" constraint="left"/>
    <ADD component="simpleview" constraint="center"/>
</LAYOUT>

```



```

<TEMPLATES source="sectiontemplates">
  <TEMPLATE uid="rtt_variant_text_image"/>
</TEMPLATES>
</DATASOURCE>
<LANGINFOS>
  <LANGINFO lang="*" label="Variants"/>
  <LANGINFO lang="DE" label="Varianten"/>
</LANGINFOS>
</FS_LIST>

```

Im Ausgabekanal wird die Standardausgabe in zwei Punkten erweitert. Zunächst benötigt jede Variante eine eindeutige ID, die wir aus dem Kürzel *rttSection* und der ID des Absatzes erzeugen.

```

$CMS_SET(rttSection, "id=\"rttSection\" + #global.section.id + \" \"")$

```

Diese ID wird später als ID des Divs gesetzt, das den Teaser darstellt.

```

<div $CMS_VALUE(rttSection)$class="col-md-4"$CMS_VALUE(editorId())$>

```

Als nächstes wird der JSTL-Tag *rtt:showContent* um den Inhalt des Teasers gelegt, um diesen je nach User-Segmenten ein- oder ausblenden zu können. Als Parameter müssen die ID sowie die Gültigkeitszeiträume übergeben werden.

```

<rtt:showContent
  id="$CMS_VALUE(#global.section.id)$"
  $CMS_VALUE(setStartDate)$
  $CMS_VALUE(setEndDate)$
  referencedate="$CMS_VALUE(#global.startTime.getTimeInMillis())$" >

```

## 7.2.2 Teaser Varianten Absatz

Die eigentlichen Varianten bekommen ein eigenes Absatztemplate, das eine Kopie des vorherigen Teaser-Absatzes ist. Bei der Kopie wird die Eingabekomponente für die Varianten entfernt. Dafür kommen zwei neue Eingabekomponenten für die Segmente und deren Verknüpfungslogik hinzu:

```

<CMS_INPUT_RADIOBUTTON name="st_targetSegmentLogic" gridHeight="1"
gridWidth="2" hFill="yes" useLanguages="no">
<ENTRIES>
  <ENTRY value="OR">
    <LANGINFOS>
      <LANGINFO lang="*" label="OR"/>

```



```

    <LANGINFO lang="DE" label="ODER"/>
  </LANGINFOS>
</ENTRY>
<ENTRY value="AND">
  <LANGINFOS>
    <LANGINFO lang="*" label="AND"/>
    <LANGINFO lang="DE" label="UND"/>
  </LANGINFOS>
</ENTRY>
</ENTRIES>
<LANGINFOS>
  <LANGINFO lang="*" label="Logic to combine segments" description="Should
segments be combined with OR or AND?"/>
  <LANGINFO lang="DE" label="Segmente verknüpfen mit" description="Mit
welcher Logik sollen die Segmente verknüpft werden?"/>
</LANGINFOS>
</CMS_INPUT_RADIOBUTTON>

<CMS_INPUT_CHECKBOX name="st_targetSegments" gridWidth="3" hFill="yes"
useLanguages="no">
<CMS_INCLUDE_OPTIONS type="public">
  <LABELS>
    <LABEL lang="DE">#item</LABEL>
    <LABEL lang="*">#item</LABEL>
  </LABELS>
  <NAME>com.espirit.moddev.rtt.gui.WoopraValueProvider</NAME>
</CMS_INCLUDE_OPTIONS>
<LANGINFOS>
  <LANGINFO lang="*" label="Show to segment(s)" description="Information is
relevant and should be shown to the selected segments."/>
  <LANGINFO lang="DE" label="Relevanz für Segmente"
description="Informationen sind relevant und sollen den ausgewählten
Segmenten angezeigt werden."/>
</LANGINFOS>
</CMS_INPUT_CHECKBOX>

```

Im Ausgabekanal wird die Standardausgabe in mehreren Punkten erweitert. Zunächst benötigt jede Variante eine eindeutige ID, die wir aus dem Kürzel *rttSection* und der ID des Absatzes erzeugen und dem umgebenden Div hinzufügen.

```
<div id="rttSection$CMS_VALUE(#global.section.id)$" [...]
```

Auch hier wird der JSTL-Tag *rtt:showContent* um den Inhalt des Teasers gelegt, um diesen je nach User-Segmenten ein- oder ausblenden zu können. Als Parameter



müssen – neben der ID und den Gültigkeitszeiträumen – auch die Verknüpfungslogik der Segmente sowie die selektierten Segmente selbst übergeben werden.

```
<rtt:showContent
segments="$CMS_VALUE(st_targetSegments.toString(", "))$"
logic="$CMS_VALUE(st_targetSegmentLogic)$"
id="$CMS_VALUE(#global.section.id)$"
$CMS_VALUE(setStartDate)$
$CMS_VALUE(setEndDate)$
referencedate="$CMS_VALUE(#global.startTime.getTimeInMillis())$">
```

Nach dem Abschluss der Arbeiten können auf der Seite neue Absätze mit Varianten angelegt werden und mithilfe des Persona-Simulators kann der Redakteur bereits in der Vorschau die Ansicht für Benutzer, die bestimmte Segmente besitzen, testen.

### 7.3 Woopra™ Tracking einrichten

Sollen zusätzlich neue Seiten von *Woopra™* verfolgt werden, so ist die in *Kapitel 7.1* beschriebene Seitenvorlage am Ende des Bodys um folgende Zeile zu erweitern.

```
$CMS_RENDER(template:"woopra_tracking", woopra_domain:"mydomain.com")$
```

Die Domain muss in diesem Fall noch an die eigene, zu verfolgende Domain angepasst werden.

### 7.4 Scoring einrichten

Für die Seite soll im nächsten Schritt die Möglichkeit integriert werden, einen Score-Wert zu vergeben. Dieser soll es später ermöglichen, den User anhand seines Scores in *Woopra™* speziellen Segmenten zuzuordnen.

Damit jeder Seite mehrere, frei definierbare Attribute und Werte zugeordnet werden können, entscheiden wir uns in diesem Fall für eine FS\_LIST-Eingabekomponente. Um diese mit beliebigen Werten füllen zu können, legen wir zunächst eine neue Absatzvorlage an, die Auswahl- bzw. Eingabemöglichkeiten für den Score-Wert sowie den Score-Namen bietet:

```
<CMS_INPUT_COMBOBOX name="st_score_name" noBreak="yes" useLanguages="no">
<ENTRIES>
  <ENTRY value="homepage_score">
    <LANGINFOS>
```



```

    <LANGINFO lang="*" label="Homepage"/>
  </LANGINFOS>
</ENTRY>
</ENTRIES>
<LANGINFOS>
  <LANGINFO lang="*" label="Score Name" description="Name for rtt scoring."/>
  <LANGINFO lang="DE" label="Score Name" description="Name für RTT score."/>
</LANGINFOS>
</CMS_INPUT_COMBOBOX>

```

Diese referenzieren wir dann in der FS\_LIST-Eingabekomponente in der Seitenvorlage.

```

<FS_LIST name="pt_scoring" hFill="yes" rows="10">
<DATASOURCE type="inline" useLanguages="no">
  <LABELS>
    <LABEL lang="*">#item.st_score_name + "\t\t (Score: " +
#item.st_score_value + ")</LABEL>
  </LABELS>
  <ACTIONS>
    <ACTION name="ADD"/>
    <ACTION name="REMOVE"/>
    <ACTION name="UP"/>
    <ACTION name="DOWN"/>
    <ACTION name="EDIT"/>
  </ACTIONS>
  <COLUMNS>
    <COLUMN show="no">#identifier</COLUMN>
  </COLUMNS>
  <LAYOUT>
    <ADD component="toolbar" constraint="top"/>
    <ADD component="overview" constraint="center"/>
    <ADD component="simpleview" constraint="bottom"/>
  </LAYOUT>
  <TEMPLATES source="sectiontemplates">
    <TEMPLATE uid="scoring"/>
  </TEMPLATES>
</DATASOURCE>
<LANGINFOS>
  <LANGINFO lang="*" label="Score" description="Add score."/>
</LANGINFOS>
</FS_LIST>

```

Als letzter Schritt muss der in *Kapitel 7.3* eingerichtete *Woopra™*-Trackingcode noch



um den Score erweitert werden. Dazu schreiben wir die in den Eingabekomponenten ausgewählten Werte in den Trackingcode der Formatvorlage (*woopra\_tracking*):

```
<script>
[... ]
$CMS_SET(hasScore, false)$
$CMS_SET(scores, "")$

$CMS_IF(pt_scoring != null)$
  $CMS_FOR(entry, pt_scoring)$
    $CMS_IF(entry.st_score_value > 0)$CMS_SET(hasScore,
true)$CMS_END_IF$
    $CMS_IF(#for.index > 0)$CMS_SET(scores, scores + ",")$CMS_END_IF$
    $CMS_SET(scores, scores + entry.st_score_name + " : " +
entry.st_score_value)$
  $CMS_END_FOR$
$CMS_END_IF$

woopra.track();

$CMS_IF(hasScore)$
  woopra.track('scoring', {
    $CMS_VALUE(scores)$
  });
$CMS_END_IF$
</script>
```

Danach wird der Score des entsprechenden Elements für jeden Besucher der Seite um den konfigurierten Wert erhöht. In *Woopra*™ lassen sich dadurch noch weitere Regeln erstellen, um den Besucher ab einem bestimmten Wert einem Segment zuzuordnen (s. *Kapitel 5.2.2*).

## 7.5 SmartForms einrichten

Um die Seite um eine *SmartForm* zu erweitern, verwenden wir als Basis das FirstSpirit-Modul *FormEdit*. Dazu muss dieses gemäß Installationsanleitung installiert und die Templates ins Projekt importiert werden. Alternativ lassen sich *SmartForms* auch ohne *FormEdit* benutzen, dann muss lediglich das Formulargerüst auf anderem Weg erzeugt werden. Mittels *SmartForms* sollen der Name und die E-Mail des Benutzers beim erneuten Aufruf des Formulars in diesem Beispiel automatisch ausgefüllt werden.



### 7.5.1 Formular anlegen

Im ersten Schritt legen wir im Inhaltsbereich *contentbottom* ein neues Formular an, indem wir *form start*, *form block* und *form end* Absätze hinzufügen. Innerhalb des *form block* – Elements werden noch die Text-Blöcke *name* und *email* hinzugefügt. Die eindeutigen Bezeichner sind dabei für die Funktionalität der Scripte der *SmartForms* von entscheidender Bedeutung.

### 7.5.2 Anpassungen Template „FormularStart“

Das FormularStart-Template muss zur Benutzung von *SmartForms* erweitert werden. Über das hinzugefügte Script werden die Werte von Name und E-Mail des Benutzers vor dem Versand des Formulars an *Woopra*™ weiter geleitet.

```
<script>
  var submit_$CMS_VALUE(fr_st_name)$ = false;
  $('#$CMS_VALUE(fr_st_name)$').submit(function(e) {
    var formName = $('#name').val();
    var formEmail = $('#email').val();
    woopra.identify({
      email: formEmail,
      name: formName
    }).push();
    setTimeout(function() {
      submit_$CMS_VALUE(fr_st_name)$ = true;
      $('#$CMS_VALUE(fr_st_name)$').submit();
    }, 2000);
    if(!submit_$CMS_VALUE(fr_st_name)$) {
      e.preventDefault();
    }
  });
</script>
```



### 7.5.3 Anpassungen Template „formText“

Um beim erneuten Aufruf des Formulars die Eingabekomponenten für Name und E-Mail mit Werten aus Woopra™ vorzubelegen, ist die Erweiterung des Templates *formText* notwendig. Wenn der Benutzer eine Eingabe vorgenommen hat, so wird dieser Wert angezeigt, ansonsten wird der gespeicherte Wert aus *Woopra™* angezeigt. Dieser befindet sich aufgrund der definierten Filter (s. *Anpassungen web.xml* im nächsten Kapitel) bereits in der User-Session.

```
$CMS_SET(fr_st_value)$  
  $CMS_IF(!st_value.isEmpty)$  
    $CMS_VALUE(st_value)$  
  $CMS_ELSE$  
    <c:out value="$ {sessionScope['rtt.woopra.user.$CMS_VALUE(st_name)$'] }"/>  
  $CMS_END_IF$  
$CMS_END_SET$
```

### 7.5.4 Anpassungen web.xml

Innerhalb der Projekteigenschaften für die entsprechende Web-Komponente muss die *web.xml* der *Woopra™* Web-Applikation angepasst werden, indem ihr die Pipeline-Filter *WoopraFirstVisitFilter* und *GetWoopraUserInformationFilter* hinzugefügt werden (siehe Kapitel 3.5.2.3 und Kapitel 3.5.2.4). Diese Filter ermöglichen die Ermittlung der in *Woopra™* zu einem Besucher gespeicherten Attribute.



## 8 Rechtliche Hinweise

Das Real-time Targeting-Modul ist ein Produkt der e-Spirit AG, Dortmund, Germany.

Für die Verwendung des Modules 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 in der Datei „third-party-dependencies.txt“, die mit dem Modul ausgeliefert wird.

