greetingcards – TDD mit SPAs, Selenium, QUnit JS, JEE6, REST, JPA 2, EJB 3.1

Single Page Web-Applikationen (**SPAs**) haben nicht nur wegen Ihrer guten Laufzeit-Performance und großen Flexibilität (Dual-Use bzw. Multi-Use) sowohl in Web-Applikationen als auch in modernen Mobile Applikationen zunehmend an Bedeutung gewonnen. Auch im Rahmen der Test-getriebene Entwicklung (**TDD**) als einer wichtige Säule der agilen Architektur- und Softwareentwicklung sind SPAs als Test-Sites für Consumer von WebService-Endpoints bzw. zum integrativen Testen von RESTful-WebServices, z.B. mittels Ajax Calls oder asynchronen Test-Aufrufen, weitverbreitet. Der Einsatz von SPAs ist somit ein wichtiger Teil der Best Practices bei der Integration von Service-Aufrufen in produktive Web-Frontends und Mobile Apps.

Außer Beispielen zu Java/JEE-implementierten integrativen EJB Backend Service-Tests mittels dem Arquillan-Testframework sollen vor allem mit dem QUnit JavaScript-Testframework integrative RESTful WebService-Tests aus der SPA Web- und Mobile Applikation heraus durchgeführt werden. Hierfür wird auch eine „QUnit Tests"-Schaltfläche in der App bereitgestellt.

Weiterhin wird der Registrierungsprozess verschiedener Domain-Daten (diese entsprechen in dem erstellen Beispiel den Entity-Daten) auch integrativ mittels Selenium IDE bzw. Selenium Browser-PlugIn (für FireFox oder Chrome) über die App im Browser durchgeführt und jederzeit für das Browser-PlugIn reproduzierbar als Test-Suite wiederaufrufbar zur Verfügung gestellt. Hierfür wird vor dem Ablauf der automatisierten Selenium Tests die SPA unter http://localhost:8080/greetingards/ geöffnet.

TDD der Beispiel Applikation

Die Beispiel-App 'greetingcards' behandelt das reale Beispiel-Szenario einer Mobile App zur Administration der Grußkarten Domain-Daten und der Überprüfung der korrekten Funktionsweise der 'greetingcards' Mobile App. Dabei wird jeweils die implementierte REST-WebService Funktionalität durch die Tests mit QUnit bereits vor der Service-Implementierung abgedeckt. Mittels integrativer Selenium-Tests wird die über die UX-Benutzeroberfläche bereitgestellte Funktionalität der Domain-Registrierung getestet und diese Tests werden gespeichert, um auch später jederzeit die korrekte Funktionsweise der Eingabe-Aktionen und Link-Aufrufe der Mobile Applikation automatisiert über die erstellte Selenium Test-Suite verifizieren zu können.

Die Technologie:
Die Beispiel-App 'greetingcards' verwendet Backbone.js MVC, Underscore.js und jQuery für das HTML5-/CSS3-Frontend sowie JAX-RS und Java EE (Stateless/Stateful EJB 3.1 und JEE 6 mittels CDI, JPA) für das Service-Backend auf dem JBoss AS 7.4 mit der Produktbezeichnung "JBoss EAP 6.2". Für den Build/das Deployment des Projekts ist mindestens Java 6 und Maven 3 erforderlich. Als Entwicklungsumgebung wurde Eclipse 4.3 (Kepler) in Form des JBoss Developer Studios 7.1 mit Java 7 verwendet und die FireFox WebDeveloper IDE/Tools.

Die Aktivierung von CDI erfolgt, wie beschrieben, mittels **beans.xml** im WEB-INF Verzeichnis.

Die Architektur:
Backbone.js als JavaScript MVC Framework liefert das Model (JavaScript/JSON Objekte mit den angeforderten Daten), eine (SPA) View (HTML, Templates, CSS) und Controller (in der JavaScript Datei app.js) unter Verwendung des **Command Design Patterns** um Events von Browser Eingaben auf Event Handler-Actions zu mappen, die wiederum RESTful Webservices aufrufen um in Interaktion mit den Backend Services die JSON Daten zu erhalten und das Ergebnis zurückzuliefern. Dabei geschieht das Mapping auf die Event Handler Function-Objekte in der von der Backbone.View abgeleiteten Application View.

a) Service Backend:
Dabei entsprechen die im Service-Backend verwendeten Stateless/Stateful Session Beans (EJB 3.1) vom **Design Pattern** her der **Facade** (SLSB – Stateless Session Bean) für ein einheitliches Interface und dem **DAO** (SFSB – Stateful Session Bean). Da SFSB ja im Gegensatz zu SLSB nicht in einem Pool verwaltet werden, sondern serialisiert werden, sind sie gut geeignet zur Bereitstellung und Verarbeitung der serialisierten JSON-Daten. Gleichzeitig werden die SFSBs ('Stateful' bezieht sich nur auf den technischen Lifecycle der Bean) für die RESTful WebServices als zustandslose DAOs eingesetzt, während die SLSBs hier ohne deklarativ annotierte @Transaction oder zusätzliche User Transaction verwendet werden können.

Hier das **QuotesRepository** der für die Grusskarten verwendeten Quotes unter Verwendung der JPA GcQuotes Entity und der benötigten Queries:

```
package de.binaris.greetingcards.data;

import de.binaris.greetingcards.model.GcQuotes;

import javax.enterprise.context.ApplicationScoped;
import javax.inject.Inject;
import javax.persistence.EntityManager;
import javax.persistence.criteria.CriteriaBuilder;
import javax.persistence.criteria.CriteriaQuery;
import javax.persistence.criteria.Root;

import java.util.List;

@ApplicationScoped
public class QuoteRepository {

   @Inject
   private EntityManager em;

   public GcQuotes findById(Long idGcQuotes) {
      return em.find(GcQuotes.class, idGcQuotes);
   }
```

```
    public GcQuotes findByIdQuote(Integer idGcQuotes) {
        CriteriaBuilder cb = em.getCriteriaBuilder();
        CriteriaQuery<GcQuotes> criteria = cb.createQuery(GcQuotes.class);
        Root<GcQuotes> quote = criteria.from(GcQuotes.class);
        criteria.select(quote).where(cb.equal(quote.get("idGcQuotes"), idGcQuotes));
        return em.createQuery(criteria).getSingleResult();
    }

    public List<GcQuotes> findOrderedByIdGcQuotes(Integer idGcQuotes) {
        CriteriaBuilder cb = em.getCriteriaBuilder();
        CriteriaQuery<GcQuotes> criteria = cb.createQuery(GcQuotes.class);
        Root<GcQuotes> quote = criteria.from(GcQuotes.class);
        criteria.select(quote).where(cb.equal(quote.get("idGcQuotes"),
                idGcQuotes)).orderBy(cb.asc(quote.get("idGcQuotes")));
        return em.createQuery(criteria).getResultList();
    }

    public List<GcQuotes> findAllOrderedByIdGcQuotes() {
            CriteriaBuilder cb = em.getCriteriaBuilder();
            CriteriaQuery<GcQuotes> criteria = cb.createQuery(GcQuotes.class);
            Root<GcQuotes> quote = criteria.from(GcQuotes.class);
            criteria.select(quote).orderBy(cb.asc(quote.get("idGcQuotes")));
            return em.createQuery(criteria).getResultList();
    }

    public List<GcQuotes> findAllOrderedByQuote() {
            CriteriaBuilder cb = em.getCriteriaBuilder();
            CriteriaQuery<GcQuotes> criteria = cb.createQuery(GcQuotes.class);
            Root<GcQuotes> quote = criteria.from(GcQuotes.class);
            criteria.select(quote).orderBy(cb.asc(quote.get("spruchText")));
            return em.createQuery(criteria).getResultList();
    }

    public List<GcQuotes> findByQuoteType(String quoteType) {
            CriteriaBuilder cb = em.getCriteriaBuilder();
            CriteriaQuery<GcQuotes> criteria = cb.createQuery(GcQuotes.class);
            Root<GcQuotes> quote = criteria.from(GcQuotes.class);
            criteria.select(quote).where(cb.equal(quote.get("spruchType"), quoteType));
            return em.createQuery(criteria).getResultList();
    }
}
```

Und hier der **QuoteRegistration-Service (**als **SLSB)**:

```
package de.binaris.greetingcards.service;
import de.binaris.greetingcards.model.GcQuotes;

import javax.ejb.Stateless;
import javax.enterprise.event.Event;
import javax.inject.Inject;
import javax.persistence.EntityManager;

import java.util.logging.Logger;

// No need for transaction demarcation
@Stateless
public class QuoteRegistration {

    @Inject
    private Logger log;

    @Inject
    private EntityManager em;

    @Inject
    private Event<GcQuotes> quoteEventSrc;

    public GcQuotes register(GcQuotes quote) throws Exception {
        log.info("Registering Quote ");
        em.persist(quote);
        log.info("Persisted ID=" + quote.getIdGcQuotes());
        quoteEventSrc.fire(quote);
        return quote;
    }
}
```

b) REST Webservice:

Über den Einsatz von JBoss RESTEasy für RESTful Webservices für SPAs gibt es bereits Beispiele in diesem Blog. Die Aktivierung des RESTEasy-Webservices geschieht dort über die web.xml, auf die im Beispiel-Projekt durch die Verwendung von JAX-RS/Java EE 6 verzichtet wird.

Hier beispielhaft die klassische Verwendung der web.xml (Servlet 2.3 oder auch Servlet 2.5) zur Aktivierung von RESTEasy Webservices:

```
<?xml version="1.0"?>
<!DOCTYPE web-app PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
      "http://java.sun.com/dtd/web-app_2_3.dtd">

<web-app>
  <display-name>Archetype Created Web Application</display-name>
  <context-param>
    <param-name>javax.ws.rs.core.Application</param-name>
    <param-value>de.binaris.rest.samples.service.SumOfMultiplesApplication</param-value>
  </context-param>

  <context-param>
    <param-name>resteasy.servlet.mapping.prefix</param-name>
    <param-value>/resteasy</param-value>
  </context-param>

  <listener>
    <listener-class>
      org.jboss.resteasy.plugins.server.servlet.ResteasyBootstrap
    </listener-class>
  </listener>

  <servlet>
    <servlet-name>Resteasy</servlet-name>
    <servlet-class>
      org.jboss.resteasy.plugins.server.servlet.HttpServletDispatcher
    </servlet-class>
  </servlet>

  <servlet-mapping>
    <servlet-name>Resteasy</servlet-name>
    <url-pattern>/resteasy/*</url-pattern>
  </servlet-mapping>
</web-app>
```

Stattdessen wird nun die folgende Klasse JaxRsActivator verwendet, die von Application abgeleitet ist und mit der **@ApplicationPath** Annotation annotiert ist, da seit **servlet-3.0** keine web.xml mehr benötigt wird, sondern alles über Annotationen deklarierbar ist:

```
package de.binaris.greetingcards.rest;

import javax.ws.rs.ApplicationPath;
import javax.ws.rs.core.Application;

@ApplicationPath("/rest")
public class JaxRsActivator extends Application {
   /* neither extended class attributes nor methods required */
}
```

Hier der Beispiel REST-Webservice zum Eintragen/Auslesen der für die Grußkarten verwendeten Quotes:

```
package de.binaris.greetingcards.rest;

import java.util.HashMap;
import java.util.HashSet;
import java.util.List;
import java.util.Map;
import java.util.Set;
import java.util.logging.Logger;

import javax.ejb.Stateful;
import javax.enterprise.context.RequestScoped;
import javax.inject.Inject;
import javax.persistence.NoResultException;
import javax.validation.ConstraintViolation;
import javax.validation.ConstraintViolationException;
import javax.validation.ValidationException;
import javax.validation.Validator;
```

```
import javax.ws.rs.Consumes;
import javax.ws.rs.GET;
import javax.ws.rs.POST;
import javax.ws.rs.Path;
import javax.ws.rs.PathParam;
import javax.ws.rs.Produces;
import javax.ws.rs.WebApplicationException;
import javax.ws.rs.core.MediaType;
import javax.ws.rs.core.Response;

import de.binaris.greetingcards.data.QuoteRepository;
import de.binaris.greetingcards.model.GcQuotes;
import de.binaris.greetingcards.service.QuoteRegistration;

/**
 * JAX-RS Quote Service
 * <p/>
 * This class produces a RESTful service to read/write the quotes of the
 * gc_quotes table.
 */
@Path("/quotes")
@RequestScoped
@Stateful
public class QuoteService {
    @Inject
    private Logger log;

    @Inject
    private Validator validator;

    @Inject
    private QuoteRepository repository;

    @Inject
    QuoteRegistration registration;

    @GET
    @Produces(MediaType.APPLICATION_JSON)
    public List<GcQuotes> listAllQuotes() {
        return repository.findAllOrderedByIdGcQuotes();
    }

    @GET
    @Path("/{id:[0-9][0-9]*}")
    @Produces(MediaType.APPLICATION_JSON)
    public GcQuotes lookupQuoteById(@PathParam("id") long id) {
            GcQuotes quote = repository.findById(id);
        if (quote == null) {
            throw new WebApplicationException(Response.Status.NOT_FOUND);
        }
        return quote;
    }

    /**
     * Creates a new quote from the values provided. Performs validation, and
     * will return a JAX-RS response with either 200 ok, or with a map of
     * fields, and related errors.
     */
    @POST
    @Consumes(MediaType.APPLICATION_JSON)
    @Produces(MediaType.APPLICATION_JSON)
    public Response createQuote(GcQuotes quote) {

        Response.ResponseBuilder builder = null;

        try {
            // Validates quote using bean validation
            validateQuote(quote);

            quote = registration.register(quote);

            // Create an "ok" response
            builder = Response.ok(quote);
        } catch (ConstraintViolationException ce) {
            // Handle bean validation issues
            builder = createViolationResponse(ce.getConstraintViolations());
        } catch (ValidationException e) {
```

```
            // Handle the unique constraint violation
            Map<String, String> responseObj = new HashMap<String, String>();
            responseObj.put("idGcQuotes", "idGcQuotes taken");
            builder = Response.status(Response.Status.CONFLICT).entity(
                    responseObj);
        } catch (Exception e) {
            // Handle generic exceptions
            Map<String, String> responseObj = new HashMap<String, String>();
            responseObj.put("error", e.getMessage());
            builder = Response.status(Response.Status.BAD_REQUEST).entity(
                    responseObj);
        }
        return builder.build();
    }

    /**
     * <p>
     * Validates the given quote variable and throws validation exceptions
     * based on the type of error. If the error is standard bean validation
     * errors then it will throw a ConstraintValidationException with the set of
     * the constraints violated.
     * </p>
     * <p>
     * If the error is caused it throws a regular validation exception,
     * which can be interpreted separately.
     * </p>
     *
     * @param quote
     *            GcQuotes to be validated
     * @throws ConstraintViolationException
     *             If Bean Validation errors exist
     * @throws ValidationException
     *             If quote with the same idGcQuotes already exists
     */
    private void validateQuote(GcQuotes quote)
        throws ConstraintViolationException, ValidationException {
        // Create a bean validator and check for issues.
        // […] validation goes here
    }

    /**
     * Creates a JAX-RS "Bad Request" response including a map of all violation
     * fields, and their message. This can then be used by clients to show
     * violations.
     *
     * @param violations
     *            A set of violations that needs to be reported
     * @return JAX-RS response containing all violations
     */
    private Response.ResponseBuilder createViolationResponse(
            Set<ConstraintViolation<?>> violations) {
        log.fine("Validation completed. violations found: " + violations.size());

        Map<String, String> responseObj = new HashMap<String, String>();

        for (ConstraintViolation<?> violation : violations) {
            responseObj.put(violation.getPropertyPath().toString(),
                    violation.getMessage());
        }
        return Response.status(Response.Status.BAD_REQUEST).entity(responseObj);
    }
}
```

c) Die Datenbank: SQL-Export

Hier auch der mittels Heidi SQL erstellte **Export der 'greetingcards'** MySQL 5 - Datenbank.

```
/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET NAMES utf8 */;
/*!40014 SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0 */;
/*!40101 SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='NO_AUTO_VALUE_ON_ZERO' */;

-- Exportiere Datenbank Struktur für greetingcards
CREATE DATABASE IF NOT EXISTS `greetingcards` /*!40100 DEFAULT CHARACTER SET utf8 COLLATE
utf8_unicode_ci */;
USE `greetingcards`;
```

```
-- Exportiere Struktur von Tabelle greetingcards.gc_card_info
CREATE TABLE IF NOT EXISTS `gc_card_info` (
  `ID_GC_CARD_INFO` bigint(20) NOT NULL AUTO_INCREMENT,
  `ID_MEDIA` int(11) NOT NULL,
  `ID_IMAGE_BG` int(11) DEFAULT NULL,
  `ID_GC_SUBCATEGORY` int(11) NOT NULL,
  `ID_GC_CATEGORY` int(11) NOT NULL,
  `MESSAGE` text,
  `SEND_DATE` date DEFAULT NULL,
  `KONFUZ_TEXT` varchar(250) DEFAULT NULL,
  `FLIRT_TEXT` varchar(250) DEFAULT NULL,
  `JOKE_TEXT` varchar(250) DEFAULT NULL,
  `ID_GC_CUSTOMER` int(11) DEFAULT NULL,
  `FONT_COLOR` varchar(10) DEFAULT NULL,
  `FONT_FAMILY` varchar(150) DEFAULT NULL,
  `FONT_SIZE` varchar(2) DEFAULT NULL,
  `CARD_LAYOUT` int(2) DEFAULT NULL,
  `THUMB_PATH` varchar(80) DEFAULT NULL,
  PRIMARY KEY (`ID_GC_CARD_INFO`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

-- Exportiere Struktur von Tabelle greetingcards.gc_card_sent
CREATE TABLE IF NOT EXISTS `gc_card_sent` (
  `ID_GC_CARD_SENT` bigint(20) NOT NULL AUTO_INCREMENT,
  `ID_GC_CARD_INFO` bigint(20) NOT NULL,
  `ID_MEDIA` int(11) NOT NULL,
  `TO_NAME` varchar(60) DEFAULT NULL,
  `TO_EMAIL` text,
  `TO_VORNAME` varchar(60) DEFAULT NULL,
  `FROM_NAME` varchar(60) DEFAULT NULL,
  `FROM_EMAIL` text,
  `FROM_VORNAME` varchar(60) DEFAULT NULL,
  `CREATE_DATE` date DEFAULT NULL,
  `EXPIRE_ARCHIV` date DEFAULT NULL,
  `SEND_ON_PICKUP` varchar(10) DEFAULT NULL,
  `IS_SENT` tinyint(2) DEFAULT NULL,
  `SEEN_DATE` date DEFAULT NULL,
  `EXPIRE_DATE` date DEFAULT NULL,
  `SEEN_COUNT` tinyint(4) DEFAULT NULL,
  PRIMARY KEY (`ID_GC_CARD_SENT`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

-- Exportiere Struktur von Tabelle greetingcards.gc_category
CREATE TABLE IF NOT EXISTS `gc_category` (
  `ID_GC_CATEGORY` int(11) NOT NULL AUTO_INCREMENT,
  `CATEGORY_ID` int(11) NOT NULL DEFAULT '0',
  `APPROVED` tinyint(1) DEFAULT '1',
  `TEXT_COLOR` varchar(8) DEFAULT NULL,
  `COMMERCIAL` varchar(150) DEFAULT NULL,
  `HTML_TITLE` blob,
  `FONT_SIZE` varchar(2) DEFAULT NULL,
  `HTML_KEYWORD` blob,
  `HTML_DESCRIPTION` blob,
  `DESCRIPTION` varchar(200) DEFAULT NULL,
  `TEXT_LINK` varchar(200) DEFAULT NULL,
  `CATEGORY_DESCRIPTION` blob,
  PRIMARY KEY (`ID_GC_CATEGORY`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 ROW_FORMAT=COMPACT;

-- Exportiere Struktur von Tabelle greetingcards.gc_flash
CREATE TABLE IF NOT EXISTS `gc_flash` (
  `ID_GC_FLASH` int(11) NOT NULL AUTO_INCREMENT,
  `ID_MEDIA` int(11) NOT NULL DEFAULT '0',
  `MEDIA_TYPE_ID` int(11) DEFAULT '1',
  `NAME` varchar(40) DEFAULT NULL,
  `URL` varchar(150) DEFAULT NULL,
  `FULL_URL` varchar(150) DEFAULT NULL,
  `LINK` varchar(255) DEFAULT NULL,
  `APPROVED` tinyint(1) DEFAULT '1',
  `PIC_NAME` varchar(100) DEFAULT NULL,
  `EXT_ID` int(11) DEFAULT NULL,
  `EXT_NR` tinyint(3) DEFAULT NULL,
  `EXT_RUBRIK` tinyint(1) DEFAULT '0',
  `POOL_ID` tinyint(3) DEFAULT NULL,
  PRIMARY KEY (`ID_GC_FLASH`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 ROW_FORMAT=COMPACT;
```

```
-- Exportiere Struktur von Tabelle greetingcards.gc_image
CREATE TABLE IF NOT EXISTS `gc_image` (
  `ID_GC_IMAGE` int(11) NOT NULL AUTO_INCREMENT,
  `ID_MEDIA` int(11) NOT NULL DEFAULT '0',
  `MEDIA_TYPE_ID` int(11) DEFAULT '1',
  `NAME` varchar(40) DEFAULT NULL,
  `URL` varchar(150) DEFAULT NULL,
  `FULL_URL` varchar(150) DEFAULT NULL,
  `LINK` varchar(255) DEFAULT NULL,
  `APPROVED` tinyint(1) DEFAULT '1',
  `PIC_NAME` varchar(100) DEFAULT NULL,
  `EXT_ID` int(11) DEFAULT NULL,
  `EXT_NR` tinyint(3) DEFAULT NULL,
  `EXT_RUBRIK` tinyint(1) DEFAULT '0',
  `POOL_ID` tinyint(3) DEFAULT NULL,
  PRIMARY KEY (`ID_GC_IMAGE`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 ROW_FORMAT=COMPACT;

-- Exportiere Struktur von Tabelle greetingcards.gc_media_type
CREATE TABLE IF NOT EXISTS `gc_media_type` (
  `ID_GC_MEDIA_TYPE` int(11) NOT NULL AUTO_INCREMENT,
  `NAME` varchar(40) DEFAULT NULL,
  `MEDIA_TYPE_ID` int(11) DEFAULT NULL,
  PRIMARY KEY (`ID_GC_MEDIA_TYPE`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 ROW_FORMAT=COMPACT;

-- Exportiere Struktur von Tabelle greetingcards.gc_quotes
CREATE TABLE IF NOT EXISTS `gc_quotes` (
  `ID_GC_QUOTES` bigint(20) NOT NULL AUTO_INCREMENT,
  `SPRUCH_TEXT` varchar(250) DEFAULT NULL,
  `SPRUCH_TYPE` varchar(2) DEFAULT NULL,
  PRIMARY KEY (`ID_GC_QUOTES`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 ROW_FORMAT=COMPACT;

-- Exportiere Struktur von Tabelle greetingcards.gc_subcategory
CREATE TABLE IF NOT EXISTS `gc_subcategory` (
  `ID_GC_SUBCATEGORY` int(11) NOT NULL AUTO_INCREMENT,
  `SUBCATEGORY_ID` int(11) NOT NULL DEFAULT '0',
  `CATEGORY_ID` int(11) NOT NULL DEFAULT '0',
  `APPROVED` tinyint(1) DEFAULT '1',
  `TEXT_COLOR` varchar(8) DEFAULT NULL,
  `COMMERCIAL` varchar(150) DEFAULT NULL,
  `HTML_TITLE` blob,
  `FONT_SIZE` varchar(2) DEFAULT NULL,
  `HTML_KEYWORD` blob,
  `HTML_DESCRIPTION` blob,
  `DESCRIPTION` varchar(200) DEFAULT NULL,
  `TEXT_LINK` varchar(200) DEFAULT NULL,
  `CATEGORY_DESCRIPTION` blob,
  PRIMARY KEY (`ID_GC_SUBCATEGORY`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 ROW_FORMAT=COMPACT;

-- Exportiere Struktur von Tabelle greetingcards.gc_subcategory_media
CREATE TABLE IF NOT EXISTS `gc_subcategory_media` (
  `ID_GC_SUBCATEGORY_MEDIA` int(11) NOT NULL AUTO_INCREMENT,
  `ID_MEDIA` int(11) NOT NULL DEFAULT '0',
  `ID_FLASH` int(11) NOT NULL DEFAULT '0',
  `MEDIA_TYPE_ID` int(11) NOT NULL DEFAULT '0',
  `SUBCATEGORY_ID` int(11) NOT NULL DEFAULT '0',
  `RFOLGE` tinyint(4) DEFAULT '0',
  PRIMARY KEY (`ID_GC_SUBCATEGORY_MEDIA`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 ROW_FORMAT=COMPACT;

/*!40101 SET SQL_MODE=IFNULL(@OLD_SQL_MODE, '') */;
/*!40014 SET FOREIGN_KEY_CHECKS=IF(@OLD_FOREIGN_KEY_CHECKS IS NULL, 1,
@OLD_FOREIGN_KEY_CHECKS) */;
/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
```

Die Testdaten aus der **import.sql:**

```
insert into Gc_Card_Info (idGcCardInfo, idMedia, idGcCategory, idGcSubcategory) values (4, 18, 5, 10)
insert into Gc_Quotes (idGcQuotes, spruchText, spruchType) values (7, 'Konfuzius sagt: Nicht mit Kanonen auf
Spatzen ballern', 'k')
```

Die persistence.xml aus dem /META-INF Unterverzeichnis zur Aktivierung von JPA 2 sieht für die Verwendung der HSQL Test-Datenbank folgendermaßen aus:

```
<?xml version="1.0" encoding="UTF-8"?>
<persistence version="2.0"
  xmlns="http://java.sun.com/xml/ns/persistence"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://java.sun.com/xml/ns/persistence
      http://java.sun.com/xml/ns/persistence/persistence_2_0.xsd">
  <persistence-unit name="greetingcards">
    <jta-data-source>java:jboss/datasources/GreetingcardsDS</jta-data-source>
    <properties>
      <property name="hibernate.hbm2ddl.auto" value="create-drop" />
      <property name="hibernate.show_sql" value="false" />
    </properties>
  </persistence-unit>
</persistence>
```

Die Verwendung von JPA 2 ist am folgenden <persistence …>-Element sehr gut zu erkennen. Denn ohne dass hier die version="2.0" eingetragen ist, fährt man gezwungenermaßen stets nur persistence 1.0, d.h. z.B. JPA 1.

```
<persistence version="2.0"
  xmlns="http://java.sun.com/xml/ns/persistence"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://java.sun.com/xml/ns/persistence
      http://java.sun.com/xml/ns/persistence/persistence_2_0.xsd">
```

Für eine produktive Applikation und Datenbank sollte die Datasource in der standalone.xml entsprechend konfiguriert werden.

Für die Test-Applikation soll die Verwendung der HSQL "in memory"-Datenbank genügen.
Hierfür wird im /WEB-INF-Verzeichnis neben der beans.xml (CDI-Aktivierung) auch die folgende **greetingcards-ds.xml** hinterlegt:

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- unmanaged datasource for testing purposes only uses H2,
      an in memory database that ships with JBoss AS. -->
<datasources xmlns="http://www.jboss.org/ironjacamar/schema"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="http://www.jboss.org/ironjacamar/schema
   http://docs.jboss.org/ironjacamar/schema/datasources_1_0.xsd">

   <!-- The datasource is bound into JNDI at this location. We reference
      this in META-INF/persistence.xml -->
   <datasource jndi-name="java:jboss/datasources/GreetingcardsDS"
      pool-name="greetingcards" enabled="true"
      use-java-context="true">
      <connection-url>
          jdbc:h2:mem:greetingcards;DB_CLOSE_ON_EXIT=FALSE;DB_CLOSE_DELAY=-1
      </connection-url>
      <driver>h2</driver>
      <security>
         <user-name>admin</user-name>
         <password>banana_joe</password>
      </security>
   </datasource>
</datasources>
```

d) die Benutzerschnittstelle:

Hier das vom Backbone.js und Underscore.js verwendete'**quote.tmpl**':

```
<script type="text/template" id="quote-tmpl">
<% var addHeader = true;
_.each(quotes, function(quote) {
   if ( Modernizr.mq( "only all and (max-width: 640px)" ) ) {
      addHeader = true;
   }
   %>
<div class="row quote">
                  <div class="col"><% if ( addHeader ) { %><div class="head">Id</div><% } %><div
class="data"><%=quote.idGcQuotes%></div></div>
                  <div class="col"><% if ( addHeader ) { %><div class="head">Quote</div><% } %><div
class="data"><%=quote.spruchText%></div></div>
                  <div class="col"><% if ( addHeader ) { %><div class="head">Category</div><% } %><div
class="data"><%=quote.spruchType%></div></div>
                  <div class="col"><% if ( addHeader ) { %><div class="head">REST URL</div><% } %><div
class="data"><a href="rest/quotes/<%=quote.idGcQuotes%>" rel="external" target="_blank" class="resturl ui-
link">JSON</a></div></div>
</div>
```

```
<% addHeader = false;
}); %>
</script>
```

Die **app.js** mit der Backbone-View, dem Backbone-Model und den fürs "Behaviour-Driven Development" (**BDD**) mittels jQuery implementierten Callbacks, sowie den (Action, EventHandling)-Mappings für cardinfo und quotes.

Der prinzipielle Aufbau eines darin verwendeten **JavaScript-Models** und der Verwendung von ***localStorage*** wird in einem Blog-Eintrag erklärt werden und das Speichern und Caching von Web- und Mobile Applikationsdaten am Beispiel der Indexed DB werden dann ebenfalls erklärt:

```
/*
 * Core JavaScript functionality for the application.  Performs the required
 * RESTful calls, validates return values, and populates the quote and cardinfo table.
 *
 * @Author: uwe
 */

/* Builds the updated table for the cardinfo and quote list */
// Load the application once the DOM is ready, using `jQuery.ready`:
$(function() {
	// Our basic **Cardinfo** model
	window.Cardinfo = Backbone.Model.extend({
		//Intentionally left empty
	});

	window.CardinfoList = Backbone.Collection.extend({
		// Specify the base url to target the rest-easy service
		url : 'rest/cardinfos',

		// Reference to this collection's model.
		model : Cardinfo
	});

	// Create our global collection of **Cardinfos**.
	window.Cardinfos = new CardinfoList;

	window.CardinfoView = Backbone.View.extend({

		// The HTML that gets created will be inserted into a parent element defined here.
		// The default is 'div' and not mandatory to be listed explicitly.
//		el : "div",
		className : "row cardinfo body",

		// Cache the template function for a single item.
		template : _.template($('#cardinfo-Body-tmpl').html()),

		// The CardinfoView listens for changes to its model, re-rendering.
		initialize : function() {
//			console.log("CardinfoView - initialize() - start");
			_.bindAll(this, 'render');

			// Listen to model changes
			this.model.on('change', this.render, this);
		},

		// Re-render the contents of the cardinfo item.
		render : function() {
//			console.log("CardinfoView - render() - start");
			$(this.el).html(this.template(this.model.toJSON()));
			return this;
		},

	});

	// Our basic **Quote** model
	window.Quote = Backbone.Model.extend({
		//Intentionally left empty
	});

	window.QuoteList = Backbone.Collection.extend({
		// Specify the base url to target the rest-easy service
		url : 'rest/quotes',

		// Reference to this collection's model.
		model : Quote
	});
```

```
// Create our global collection of **Quotes**.
window.Quotes = new QuoteList;

window.QuoteView = Backbone.View.extend({

	// The HTML that gets created will be inserted into a parent element defined here.
	// The default is 'div' and not mandatory to be listed explicitly.
//	el : "div",
	className : "row quote body",

	// Cache the template function for a single item.
	template : _.template($('#quote-Body-tmpl').html()),

	// The QuoteView listens for changes to its model, re-rendering.
	initialize : function() {
//		console.log("QuoteView - initialize() - start");
		_.bindAll(this, 'render');

		// Listen to model changes
		this.model.on('change', this.render, this);
	},

	// Re-render the contents of the quote item.
	render : function() {
//		console.log("QuoteView - render() - start");
		$(this.el).html(this.template(this.model.toJSON()));
		return this;
	},

});

// The App
// Our overall **AppView** is the top-level piece of UI.
window.AppView = Backbone.View.extend({

	// Instead of generating a new element, bind to the existing skeleton of
	// the App already present in the HTML.
	el : $("#container"),

	events : {
		"click #refreshButtonCI1": "updateCardinfoTable",
		"click #refreshButtonCI2": "updateCardinfoTable",
		"click #refreshButtonCI3": "updateQuoteTable",
		"click #refreshButtonCI4": "updateQuoteTable",
		"click #cancel" : "cancelRegistration",
		"pagebeforeshow #register-card" : "clearMessages",
		"pagebeforeshow #register-quote" : "clearMessages",
	},

	// bind to the relevant events on the `Cardinfo` and `Quotes`
	// collection, when items are added or changed, with possibility to
	// load any preexisting cardinfos, or quotes that might be saved in
	// *localStorage*.
	initialize : function() {
//		console.log("AppView - initialize() - start");
		//refer to the parent function. JS is block scoped.
		var self = this;

		Cardinfos.on('add', this.addOneCardinfo, this);
		Cardinfos.on('reset', this.addAllCardinfos, this);
		Cardinfos.on('all', this.render, this);

		$('#cardinfos').html(self.templateHeaderRowCardinfo);
		this.updateCardinfoTable();
		this.submitRegistrationCardinfo();

//		Cardinfos.fetch();

		Quotes.on('add', this.addOneQuote, this);
		Quotes.on('reset', this.addAllQuotes, this);
		Quotes.on('all', this.render, this);

		$('#quotes').html(self.templateHeaderRowQuote);
		this.updateQuoteTable();
		this.submitRegistrationQuote();
```

```
//                      Quotes.fetch();
                },

                // Re-rendering the App just means refreshing the statistics
                // the rest of the app doesn't change.
                render: function() {
                        console.log("AppView - render() - start");
                },

                // Add a single cardinfo item to the list by creating a view for it, and
                // appending its element to the `<ul>`.
                addOneCardinfo : function(cardinfo) {
                        console.log("AppView - addOneCardinfo() - start");
                        var view = new CardinfoView({
                                model : cardinfo
                        });

                        this.$("#cardinfos").append(view.render().el);
                },

                // Add a single quote item to the list by creating a view for it, and
                // appending its element to the `<ul>`.
                addOneQuote : function(cardinfo) {
                        console.log("AppView - addOneQuote() - start");
                        var view = new QuoteView({
                                model : cardinfo
                        });

                        this.$("#quotes").append(view.render().el);
                },

                // Add all items in the **Cardinfos** collection at once.
                addAllCardinfos : function() {
                        console.log("AppView - addAllCardinfos() - start");
                        Cardinfos.each(this.addOneCardinfo);
                },

                // Add all items in the **Quotes** collection at once.
                addAllQuotes : function() {
                        console.log("AppView - addAllQuotes() - start");
                        Quotes.each(this.addOneQuote);
                },

                //Create a template of the Header Row to be inserted in the AJAX call below.
                templateHeaderRowCardinfo : _.template($('#cardinfo-Header-Row-tmpl').html()),

                //Create a template of the Header Row to be inserted in the AJAX call below.
                templateHeaderRowQuote : _.template($('#quote-Header-Row-tmpl').html()),

                //Remove the table or else we will have more then one.
                $('.body').remove();
                /* Uses JAX-RS GET to retrieve current cardinfo list */
                updateCardinfoTable : function () {
                        console.log("AppView - Update Cardinfo - start");
                        //refer to the parent function. JS is block scoped.
                        var self = this;
                        //Remove the table or else we will have more then one.
                        $('.body').remove();
                        //Clear the Collection or it will try to load every Cardinfo again.
//                      Cardinfos.reset();

                        var jqxhr = $.ajax({
                                url: 'rest/cardinfos',
                                cache: false,
                                type: "GET"
                        }).done(function(data, textStatus, jqXHR) {
//                              console.log("AppView - Update Cardinfo - succes on ajax call");
                                Cardinfos.add(data);
                        }).fail(function(jqXHR, textStatus, errorThrown) {
                          console.log("AppView - Update Cardinfo - error updating table -" + error.status);
                        });
                },

                /* Uses JAX-RS GET to retrieve current quotes list */
                updateQuoteTable : function () {
                        console.log("AppView - Update Quote - start");
                        //refer to the parent function. JS is block scoped.
```

```
                        var self = this;
                        //Remove the table or else we will have more then one.
                        $('.body').remove();
                        //Clear the Collection or it will try to load every Quote again.
//                      Quotes.reset();

                        var jqxhr = $.ajax({
                                url: 'rest/quotes',
                                cache: false,
                                type: "GET"
                        }).done(function(data, textStatus, jqXHR) {
//                              console.log("AppView - Update Quote - succes on ajax call");
                                Quotes.add(data);
                        }).fail(function(jqXHR, textStatus, errorThrown) {
                          console.log("AppView - Update Quote - error updating table -" + error.status);
                        });
                },

                //Clear registration and error messages on page change
                clearMessages : function(event) {
                        console.log("AppView - clearMessages() - start");
                        $('#formMsgsCard').empty();
                        $('#formMsgsQuote').empty();
                        $('span.invalid').remove();
                },


        /**
         * Attempts to register a new cardinfo using a JAX-RS POST. The callback to refresh the cardinfo table, or
         * process JAX-RS response codes to update the validation errors.
         */
        submitRegistrationCardinfo : function(event) {
                console.log("AppView - submitRegistrationCardInfo - start");
                //refer to the parent function. JS is block scoped.
                var outerSelf = this;

                $("form#regcard").submit(function(event) {
                        console.log("AppView - submitRegistrationCardinfo - submit event - started");
                        var self = outerSelf;
                        event.preventDefault();
                        //Transform the form fields into JSON.
                        var cardinfoData = JSON.stringify(self.$("form#regcard").serializeObject());

                        var jqxhr = $.ajax({
                                url: 'rest/cardinfos',
                                contentType: "application/json",
                                dataType: "json",
                                data: cardinfoData,
                                type: "POST"
                        }).done(function(data, textStatus, jqXHR) {
                                console.log("AppView - submitRegistrationCardinfo - ajax done");
                                //clear existing msgs
                                $('span.invalid').remove();
                                $('span.success').remove();
                                //clear input fields
                                $('#regcard')[0].reset();

                                //mark success on the registration form
                                $('#formMsgsCard').append($('<span class="success">Cardinfo Registered</span>'));

                                //There is no need to clear the Collection or the 'table' as add() will add to the existing
                                // Collection which will invoke the View and add to the 'table'.
                                Cardinfos.add(data);
                                console.log("AppView - submitRegistrationCardinfo - ajax done - after Cardinfos.add call");
                        }).fail(function(jqXHR, textStatus, errorThrown) {
                                //clear existing  msgs
                                $('span.success').remove();

                                if ((jqXHR.status === 409) || (jqXHR.status === 400)) {
                                console.log("AppView - submitRegistrationCardinfo - error in ajax - Validation error
registering user! "
                                                + jqXHR.status);
                                        //clear existing msgs so that when the new message is display you don't have 2 of
them.
                                        $('span.invalid').remove();
                                        var errorMsg = $.parseJSON(jqXHR.responseText);
```

```
                                $.each(errorMsg, function(index, val) {
                                        $('<span class="invalid">' + val + '</span>').insertAfter($('#' + index));
                                });
                        } else if (jqXHR.status === 200) {
                                //It should not reach this error as long as the Server method returns data.
                                console.log("AppView - submitRegistrationCardinfo - ajax error on 200 with error
message: "
                                        + errorThrown.message);
                                console.log("AppView - submitRegistrationCardinfo - ajax error because the REST
service doesn't return" +
                                        "any data and this app expects data.  Fix the REST app.");
                                //clear existing  msgs
                                $('span.invalid').remove();
                                //clear input fields
                                $('#regcard')[0].reset();
                                //Clear the Collection or it will try to load every Cardinfo again.
                                Cardinfos.reset();
                                self.updateCardinfoTable();
                                console.log("AppView - submitRegistrationCardinfo - ajax error on 200 - after
updateCardinfoTable() call");
                        } else {
                                console.log("AppView - submitRegistrationCardinfo - error in ajax - " +
                                        "jqXHR = " + jqXHR.status +
                                        " - textStatus = " + textStatus +
                                        " - errorThrown = " + errorThrown);
                                //clear existing  msgs
                                $('span.invalid').remove();
                                $('#formMsgsCard').append($('<span class="invalid">Please try again</span>'));
                        }
                });
        });
    },

    /**
     * Attempts to register a new quote using a JAX-RS POST. The callback to refresh the quote table, or
     * process JAX-RS response codes to update the validation errors.
     */
    submitRegistrationQuote : function(event) {
        console.log("AppView - submitRegistrationQuote - start");
        //refer to the parent function. JS is block scoped.
        var outerSelf = this;

        $("form#regquote").submit(function(event) {
                console.log("AppView - submitRegistrationQuote - submit event - started");
                var self = outerSelf;
                event.preventDefault();
                //Transform the form fields into JSON.
                var quoteData = JSON.stringify(self.$("form#regquote").serializeObject());

                var jqxhr = $.ajax({
                        url: 'rest/quotes',
                        contentType: "application/json",
                        dataType: "json",
                        data: quoteData,
                        type: "POST"
                }).done(function(data, textStatus, jqXHR) {
                        console.log("AppView - submitRegistrationQuote - ajax done");
                        //clear existing msgs
                        $('span.invalid').remove();
                        $('span.success').remove();
                        //clear input fields
                        $('#regquote')[0].reset();

                        //mark success on the registration form
                        $('#formMsgsQuote').append($('<span class="success">Quote Registered</span>'));

                        //There is no need to clear the Collection or the 'table' as this add() will add to the existing
                        // Collection which will invoke the View and add to the 'table'.
                        Quotes.add(data);
                        console.log("AppView - submitRegistrationQuote - ajax done - after Quotes.add call");
                }).fail(function(jqXHR, textStatus, errorThrown) {
                        //clear existing  msgs
                        $('span.success').remove();

                        if ((jqXHR.status === 409) || (jqXHR.status === 400)) {
                                console.log("AppView - submitRegistrationQuote - error in ajax - Validation error
registering user! "
```

```
                                        + jqXHR.status);
                                //clear existing msgs so that when the new message is display you don't have 2 of
them.
                                $('span.invalid').remove();
                                var errorMsg = $.parseJSON(jqXHR.responseText);

                                $.each(errorMsg, function(index, val) {
                                        $('<span class="invalid">' + val + '</span>').insertAfter($('#' + index));
                                });
                        } else if (jqXHR.status === 200) {
                                //It should not reach this error as long as the Server method returns data.
                                console.log("AppView - submitRegistrationQuote - ajax error on 200 with error
message: "
                                        + errorThrown.message);
                                console.log("AppView - submitRegistrationQuote - ajax error because the REST
service doesn't return" +
                                "any data and this app expects data.  Fix the REST app.");
                                //clear existing  msgs
                                $('span.invalid').remove();
                                //clear input fields
                                $('#regquote')[0].reset();
                                //Clear the Collection or it will try to load every Quote again.
                                Quotes.reset();
                                self.updateQuoteTable();
                                console.log("AppView - submitRegistrationQuote - ajax error on 200 - after
updateQuoteTable() call");
                        } else {
                                console.log("AppView - submitRegistrationQuote - error in ajax - " +
                                        "jqXHR = " + jqXHR.status +
                                        " - textStatus = " + textStatus +
                                        " - errorThrown = " + errorThrown);
                                //clear existing  msgs
                                $('span.invalid').remove();
                                $('#formMsgsQuote').append($('<span class="invalid">Please try again</span>'));
                        }
                });
        });
    },

    //Register the cancel listener
    cancelRegistration : function(event) {
        console.log("AppView - start cancelRegistration");
        //clear input fields
       $('#regcard')[0].reset();
       $('#regquote')[0].reset();

       //clear existing msgs
       $('span.invalid').remove();
       $('span.success').remove();
    }

    });

    // Finally creating the **App**.
    window.App = new AppView;
});

/* Get the quote template */
function getQuoteTemplate() {
   $.ajax({
      url: "tmpl/quote.tmpl",
      dataType: "html",
      success: function( data ) {
         $( "head" ).append( data );
         updateQuoteTable();
      }
   });
}

/* Get the cardinfo template */
function getCardinfoTemplate() {
        $.ajax({
                url: "tmpl/cardinfo.tmpl",
                dataType: "html",
                success: function( data ) {
                        $( "head" ).append( data );
                        updateCardinfoTable();
```

```
                }
        });
}

/* Builds the updated table for the cardinfo list */
function buildCardinfoRows(cardinfos) {
        return _.template( $( "#cardinfo-tmpl" ).html(), {"cardinfos": cardinfos});
}

/* Builds the updated table for the quote list */
function buildQuoteRows(cardinfos) {
        return _.template( $( "#quote-tmpl" ).html(), {"quotes": quotes});
}

/* Uses JAX-RS GET to retrieve current cardinfo list */
function updateCardinfoTable() {
        $.ajax({
                url: "rest/cardinfos",
                cache: false,
                success: function(data) {
                        $('#cardinfos').empty().append(buildCardinfoRows(data));
                },
                error: function(error) {
                        console.log("error updating table -" + error.status);
                }
        });
}

/* Uses JAX-RS GET to retrieve current quote list */
function updateQuoteTable() {
        $.ajax({
                url: "rest/quotes",
                cache: false,
                success: function(data) {
                        $('#quotes').empty().append(buildQuoteRows(data));
                },
                error: function(error) {
                        console.log("error updating table -" + error.status);
                }
        });
}

/*
Attempts to register a new cardinfo using a JAX-RS POST.  The callbacks
the refresh the cardinfo table, or process JAX-RS response codes to update
the validation errors.
 */
function registerCardinfo(cardinfoData) {
        //clear existing  msgs
        $('span.invalid').remove();
        $('span.success').remove();

        $.ajax({
                url: 'rest/cardinfos',
                contentType: "application/json",
                dataType: "json",
                type: "POST",
                data: JSON.stringify(cardinfoData),
                success: function(data) {
                        //console.log("Cardinfo registered");

                        //clear input fields
                        $('#regcard')[0].reset();

                        //mark success on the registration form
                        $('#formMsgsCard').append($('<span class="success">Cardinfo Registered</span>'));

                        updateCardinfoTable();
                },
                error: function(error) {
                        if ((error.status == 409) || (error.status == 400)) {
                                //console.log("Validation error registering Cardinfo!");

                                var errorMsg = $.parseJSON(error.responseText);

                                $.each(errorMsg, function(index, val) {
                                        $('<span class="invalid">' + val + '</span>').insertAfter($('#' + index));
```

```
                                        });
                                } else {
                                        //console.log("error - unknown server issue");
                                        $('#formMsgsCard').append($('<span class="invalid">Unknown server
error</span>'));
                                }
                        }
                });
}

/*
Attempts to register a new quote using a JAX-RS POST. The callback
to refresh the quote table, or process JAX-RS response codes to update
the validation errors.
 */
function registerQuote(quoteData) {
        //clear existing msgs
        $('span.invalid').remove();
        $('span.success').remove();

        $.ajax({
                url: 'rest/quotes',
                contentType: "application/json",
                dataType: "json",
                type: "POST",
                data: JSON.stringify(quoteData),
                success: function(data) {
                        //console.log("Quote registered");

                        //clear input fields
                        $('#regquote')[0].reset();

                        //mark success on the registration form
                        $('#formMsgsQuote').append($('<span class="success">Quote Registered</span>'));

                        updateQuoteTable();
                },
                error: function(error) {
                        if ((error.status == 409) || (error.status == 400)) {
                                //console.log("Validation error registering Quote!");

                                var errorMsg = $.parseJSON(error.responseText);

                                $.each(errorMsg, function(index, val) {
                                        $('<span class="invalid">' + val + '</span>').insertAfter($('#' + index));
                                });
                        } else {
                                //console.log("error - unknown server issue");
                                $('#formMsgsQuote').append($('<span class="invalid">Unknown server
error</span>'));
                        }
                }
        });
}
```

Die einzige (single) HTML5-Seite der Applikation (**SPA**): **index.html**

```
<!DOCTYPE html>
<html>
<head>
   <title>Greetingcards Admin</title>
   <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>

   <!-- Set the tab, home page, shortcut icons -->
        <!-- <link rel="Shortcut Icon" href="img/"> -->
        <!-- <link rel="apple-touch-icon" href="img/"/> -->

   <!-- For minification, comment out this link -->
   <!-- Shared styles -->
   <link rel="stylesheet" href="css/screen.css"/>

   <!-- Set viewport scaling and zoom features -->
   <meta name="viewport" content='width=device-width, initial-scale=1">

   <script type="text/javascript" src="js/libs/modernizr-2.6.2.min.js"></script>

   <!-- For minification, comment out this file -->
```

```
<script type="text/javascript" src="js/libs/jquery-1.9.1.js"></script>
<!-- For minification, uncomment out this file -->
<!--<script type="text/javascript" src="js/libs/jquery-1.9.1.min.js"></script>-->

<!-- For minification, comment out this file -->
<script type="text/javascript" src="js/libs/lodash-1.3.1.js"></script>
<!-- For minification, uncomment out this file -->
<!--<script type="text/javascript" src="js/libs/lodash-1.3.1.min.js"></script>-->

<!-- For minification, comment out this file -->
<script type="text/javascript" src="js/libs/backbone-1.0.0.js"></script>
<!-- For minification, uncomment out this file -->
<!-- <script type="text/javascript" src="js/libs/backbone-1.0.0.min.js"></script> -->

<!-- For minification, comment out this file -->
<script type="text/javascript" src="js/app.js"></script>
<!-- For minification, uncomment out this file -->
<!-- <script type="text/javascript" src="js/app.min.js"></script> -->

<script type="text/javascript">
        // Detect if the browser viewport is small enough for a mobile or tablet
        // We shall consider both portrait and landscape modes for detection
        if (Modernizr.mq("only all and (max-width: 800px ) and (orientation: portrait)") ||
        Modernizr.mq("only all and (max-width: 1280px ) and (orientation: landscape)")) {
        // Store the result so that we can modify how the list headers are displayed
                window.loadJQM = true;
        }

        if(window.loadJQM) {
    // Minification - See Readme for details
    // For minification comment out this line
    $('head').append('<link rel="stylesheet" href="css/jquery.mobile-1.3.2.css"/>');
    // For minification uncomment out this line
    // $('head').append('<link rel="stylesheet" href="css/jquery.mobile-1.3.2.min.css"/>');

    // For minification comment out this line
    $('head').append('<link rel="stylesheet" href="css/m.screen.css"/>');
    // For minification uncomment out this line
    // $('head').append('<link rel="stylesheet" href="css/m.screen.min.css"/>');

    // For minification comment out this line
    $('head').append('<script type="text/javascript" src="js/libs/jquery.mobile-1.3.2.js">');
    // For minification uncomment out this line
    // $('head').append('<script type="text/javascript" src="js/libs/jquery.mobile-1.3.2.min.js">');
        } else {
    // For minification comment out this line
    $('head').append('<link rel="stylesheet" href="css/d.screen.css"/>');
    // For minification uncomment out this line
    // $('head').append('<link rel="stylesheet" href="css/d.screen.min.css"/>');

                $( document ).ready( function() {
                    //here we are moving content around based on the browser/device.
                    $( "aside" ).insertAfter( $( "section" ) );
                });
        }

    $( document ).ready( function() {
        //setup the app after all scripts have loaded

        $( "#container" ).show();
    });

    $.fn.serializeObject = function() {
        var o = {};
        var a = this.serializeArray();
        $.each(a, function() {
            if (o[this.name]) {
                if (!o[this.name].push) {
                    o[this.name] = [o[this.name]];
                }
                o[this.name].push(this.value || '');
            } else {
                o[this.name] = this.value || '';
            }
        });
        return o;
    };
```

```
    </script>
</head>
<body>
        <div id="container" style="display:none">
            <div class="dualbrand">
                <img src="img/rhjb_eap_logo.png"/>
            </div>
            <section>
                <h1>Welcome to Greetingcards Admin</h1>
                <article id="intro-art" data-role="page" data-theme="a">
                    <!-- Header -->
                    <div data-role="header" class="header">
                        <h3>Technologies: Backbone, jQuery, QUnit, Selenium, REST, JPA, JBoss EAP</h3>
                        <a href="#info-aside" data-role="button" data-icon="info" class="ui-btn-right" data-
iconpos="notext" data-rel="dialog"></a>
                    </div>
                    <!-- /Header -->

                    <div data-role="content">
<!--                    <img src="img/" class="mobileicon"/> -->
                        <p>Demonstrates use of Mobile, HTML5, CSS3, JavaScript and JavaEE 6 techniques.</p>
                        <div class="highlights">
                            <ul>
                                <li>HTML5 using a Backbone.js MVC</li>
                                <li>HTML5 using Underscore.js</li>
                                <li>HTML5 Client using JAX-RS REST endpoints</li>
                                <li>JBoss EAP with JPA backend and REST services</li>
                            </ul>
                        </div>
                        <div class="highlights">
                            <ul>
                                <li>jQuery Mobile Integration</li>
                                <li>QUnit test suite for TDD with JavaScript</li>
                            </ul>
                        </div>
                        <ul id="features">
                            <li class="feature">Pure HTML client</li>
                            <li class="feature">Backbone.js MVC</li>
                            <li class="feature">JAX-RS GET &amp;amp; POST end points</li>
                            <li class="feature">HTML5 based page structure</li>
                            <li class="feature">HTML5 form element &amp;amp; validation</li>
                            <li class="feature">CSS3 selectors for styling</li>
                            <li class="feature">JAX-RS validation handling</li>
                            <li class="feature">jQuery Mobile integration</li>
                            <li class="feature">QUnit test suite for TDD with JavaScript</li>
                        </ul>
                    </div>

                    <!-- Footer -->
                    <div class="footer" data-role="footer" data-position="fixed">
                        <div class="footer_left">
                            <a href="#intro-art" data-role="button" data-icon="home">Home</a>
                        </div>
                        <div class="footer_right">
                        <a href="/greetingcards/test/qunit/index.html" data-role="button" data-inline="true" data-icon="grid"
target="_self" rel="external">QUnit Tests</a>
                            <span class="footer_txt">Cards:</span>
                            <a href="#register-card" data-role="button" data-inline="true" data-icon="plus">Add</a>
                            <a href="#cardinfo-art" data-role="button" data-inline="true" data-icon="grid">List</a>
                            <span class="footer_txt">Quotes:</span>
                            <a href="#register-quote" data-role="button" data-inline="true" data-icon="plus">Add</a>
                            <a href="#quote-art" data-role="button" data-inline="true" data-icon="grid">List</a>
                        </div>
                    </div>
                    <!-- /Footer -->
                </article>

                <!-- New HTML5 article tag -->
                <article id="register-card" data-role="page" data-theme="a">
                    <!-- Header -->
                    <div data-role="header" class="header">
                        <h3>Register Card</h3>
                        <a href="#info-aside" data-role="button" data-icon="info" class="ui-btn-right" data-
iconpos="notext" data-rel="dialog"></a>
                    </div>
                    <!-- /Header -->
```

```
                <div data-role="content">
                    <!-- For now mapping bean validation constraints from server side model is a manual task -->
                    <form name="regcard" id="regcard" data-ajax="false">
                        <fieldset>
                            <legend>Register a card:</legend>
                            <div>
                                <label for="idMedia">Image Id:</label>
                                <input type="text" name="idMedia" id="idMedia" placeholder="Image Id" required
autofocus/>
                            </div>
                            <div>
                                <label for="idGcCategory">Category:</label>
                                <input type="text" name="idGcCategory" id="idGcCategory" placeholder="Card Category"
required autofocus/>
                            </div>
                            <div>
                                <label for="idGcSubcategory">Subcategory:</label>
                                <input type="text" name="idGcSubcategory" id="idGcSubcategory" placeholder="Card
Subcategory" required/>
                            </div>
                            <div id="formMsgsCard"></div>
                            <div data-role="controlgroup" data-type="horizontal">
                                <input type="submit" id="register" value="Register"/>
                                <input type="button" name="cancel" id="cancel" value="Cancel"/>
                            </div>
                        </fieldset>
                    </form>
                </div>

                <!-- Footer -->
                <div class="footer" data-role="footer" data-position="fixed">
                    <div class="footer_left">
                        <a href="#intro-art" data-role="button" data-icon="home">Home</a>
                    </div>
                    <div class="footer_right">
                    <a href="/greetingcards/test/qunit/index.html" data-role="button" data-inline="true" data-icon="grid"
target="_self" rel="external">QUnit Tests</a>
                        <span class="footer_txt">Cards:</span>
                        <a href="#register-card" data-role="button" data-inline="true" data-icon="plus">Add</a>
                        <a href="#cardinfo-art" data-role="button" data-inline="true" data-icon="grid">List</a>
                        <span class="footer_txt">Quotes:</span>
                        <a href="#register-quote" data-role="button" data-inline="true" data-icon="plus">Add</a>
                        <a href="#quote-art" data-role="button" data-inline="true" data-icon="grid">List</a>
                    </div>
                </div>
                <!-- /Footer -->
            </article>

            <!-- New HTML5 article tag -->
            <article id="cardinfo-art" data-role="page" data-theme="a">
                <!-- Header -->
                <div data-role="header" class="header" data-position="fixed">
                    <h3>Cards</h3>
                    <a href="#info-aside" data-role="button" data-icon="info" class="ui-btn-right" data-
iconpos="notext" data-rel="dialog"></a>
                </div>
                <!-- /Header -->
                <h2>Current Cards</h2>

                <div data-role="content">
                    <div id="cardinfos">
                                                <!-- This is where the list of cards gets populated by the templates. This
Header row is only
                                                        here as an example of the code that will be inserted. --
>
                    </div>
                    <div class="cardinfo-foot">
                        <!-- Sets the JAX-RS URLs to retrieve all greeting cards either as XML or JSON data.-->
                        REST URL for all greetingcards:
                        <a href="rest/cardinfos" target="_blank" rel="external">JSON</a>
                        <!-- <button id="refreshButtonCI2">Refresh Greetingcards</button> -->
                    </div>
                    <div class="quote-foot">
                        <!-- Sets the JAX-RS URLs to retrieve all quotes either as XML or JSON data.-->
                        REST URL for all quotes:
                        <a href="rest/quotes" target="_blank" rel="external">JSON</a>
                        <!-- <button id="refreshButtonD4">Refresh Quotes</button> -->
```

```
            </div>
        </div>

        <!-- Footer -->
        <div class="footer" data-role="footer" data-position="fixed">
            <div class="footer_left">
                <a href="#intro-art" data-role="button" data-icon="home">Home</a>
            </div>
            <div class="footer_right">
            <a href="/greetingcards/test/qunit/index.html" data-role="button" data-inline="true" data-icon="grid"
target="_self" rel="external">QUnit Tests</a>
                <span class="footer_txt">Cards:</span>
                <a href="#register-card" data-role="button" data-inline="true" data-icon="plus">Add</a>
                <a href="#cardinfo-art" data-role="button" data-inline="true" data-icon="grid">List</a>
                <span class="footer_txt">Quotes:</span>
                <a href="#register-quote" data-role="button" data-inline="true" data-icon="plus">Add</a>
                <a href="#quote-art" data-role="button" data-inline="true" data-icon="grid">List</a>
            </div>
        </div>
        <!-- /Footer -->
    </article>

    <!-- New HTML5 article tag -->
    <article id="register-quote" data-role="page" data-theme="a">
        <!-- Header -->
        <div data-role="header" class="header">
            <h3>Register Quote</h3>
            <a href="#info-aside" data-role="button" data-icon="info" class="ui-btn-right" data-
iconpos="notext" data-rel="dialog"></a>
        </div>
        <!-- /Header -->

        <div data-role="content">
            <!-- For now mapping bean validation constraints from server side model is a manual task -->
            <form name="regquote" id="regquote" data-ajax="false">
                <fieldset>
                    <legend>Register a Quote:</legend>
                    <div>
                        <label for="spruchText">Quote:</label>
                        <input type="text" name="spruchText" id="spruchText" placeholder="Quote Text" required
autofocus/>
                    </div>
                    <div>
                        <label for="spruchType">Type:</label>
                        <input type="text" name="spruchType" id="spruchType" placeholder="Type of Quote"
required/>
                    </div>
                    <div id="formMsgsQuote"></div>
                    <div data-role="controlgroup" data-type="horizontal">
                        <input type="submit" id="register" value="Register"/>
                        <input type="button" name="cancel" id="cancel" value="Cancel"/>
                    </div>
                </fieldset>
            </form>
        </div>

        <!-- Footer -->
        <div class="footer" data-role="footer" data-position="fixed">
            <div class="footer_left">
                <a href="#intro-art" data-role="button" data-icon="home">Home</a>
            </div>
            <div class="footer_right">
            <a href="/greetingcards/test/qunit/index.html" data-role="button" data-inline="true" data-icon="grid"
target="_self" rel="external">QUnit Tests</a>
                <span class="footer_txt">Cards:</span>
                <a href="#register-card" data-role="button" data-inline="true" data-icon="plus">Add</a>
                <a href="#cardinfo-art" data-role="button" data-inline="true" data-icon="grid">List</a>
                <span class="footer_txt">Quotes:</span>
                <a href="#register-quote" data-role="button" data-inline="true" data-icon="plus">Add</a>
                <a href="#quote-art" data-role="button" data-inline="true" data-icon="grid">List</a>
            </div>
        </div>
        <!-- /Footer -->
    </article>

    <!-- New HTML5 article tag -->
    <article id="quote-art" data-role="page" data-theme="a">
```

```
                <!-- Header -->
                <div data-role="header" class="header" data-position="fixed">
                  <h3>Quotes</h3>
                  <a href="#info-aside" data-role="button" data-icon="info" class="ui-btn-right" data-
iconpos="notext" data-rel="dialog"></a>
                </div>
                <!-- /Header -->
                <h2>Current Quotes</h2>

                <div data-role="content">
                  <div id="quotes">
                                                <!-- This is where the list of quotes gets populated by the templates.
This Header row is only
                                                                here as an example of the code that will be inserted. --
>
                  </div>
                  <div class="cardinfo-foot">
                    <!-- Sets the JAX-RS URLs to retrieve all greeting cards either as XML or JSON data.-->
                    REST URL for all greetingcards:
                    <a href="rest/cardinfos" target="_blank" rel="external">JSON</a>
                    <!-- <button id="refreshButtonCI2">Refresh Greetingcards</button> -->
                  </div>
                  <div class="quote-foot">
                    <!-- Sets the JAX-RS URLs to retrieve all greeting cards either as XML or JSON data.-->
                    REST URL for all quotes:
                    <a href="rest/quotes" target="_blank" rel="external">JSON</a>
                    <!-- <button id="refreshButtonD3">Refresh Quotes</button> -->
                  </div>
                </div>

                <!-- Footer -->
                <div class="footer" data-role="footer" data-position="fixed">
                  <div class="footer_left">
                    <a href="#intro-art" data-role="button" data-icon="home">Home</a>
                  </div>
                  <div class="footer_right">
                  <a href="/greetingcards/test/qunit/index.html" data-role="button" data-inline="true" data-icon="grid"
target="_self" rel="external">QUnit Tests</a>
                    <span class="footer_txt">Cards:</span>
                    <a href="#register-card" data-role="button" data-inline="true" data-icon="plus">Add</a>
                    <a href="#cardinfo-art" data-role="button" data-inline="true" data-icon="grid">List</a>
                    <span class="footer_txt">Quotes:</span>
                    <a href="#register-quote" data-role="button" data-inline="true" data-icon="plus">Add</a>
                    <a href="#quote-art" data-role="button" data-inline="true" data-icon="grid">List</a>
                  </div>
                </div>
                <!-- /Footer -->
              </article>

              <!-- New HTML5 aside tag -->
              <aside id="info-aside" data-role="page" data-theme="a">
                <!-- Header -->
                <div data-role="header">
                  <h3>More Info</h3>
                </div>
                <!-- /Header -->

                <div data-role="content">
                  <p>JBoss Enterprise Application Platform JEE6.</p>
                  <ul>
                    <li><a
href="https://access.redhat.com/site/documentation/JBoss_Enterprise_Application_Platform/"
target="_blank">Documentation</a></li>
                    <li><a
href="https://access.redhat.com/site/documentation/JBoss_Enterprise_Application_Platform/"
target="_blank">Download JBoss EAP 6.2 (JBoss AS Vers. 7.4)</a></li>
                  </ul>
                  <p>HTML5/Mobile RESTful development on JBoss.</p>
                  <ul>
                    <li><a href="http://aerogear.org" target="_blank">More about JPA, EJB</a></li>
                    <li><a href="http://aerogear.org/docs/guides/HTML5RESTApps" target="_blank">More on
<strong>HTML5 + REST</strong></a></li>
                    <li><a href="https://github.com/organizations/aerogear" target="_blank">Greetingcards test
project on 4shared.com</a></li>
                  </ul>
                  <p>Backbone, Underscore, jQuery development</p>
                  <ul>
```

```
                    <li><a href="http://backbonejs.org/" target="_blank">Backbone.js</a></li>
                    <li><a href="http://underscorejs.org/" target="_blank">Underscore.js</a></li>
                    <li><a href="http://jquery.com/" target="_blank">jQuery.js</a></li>
                  </ul>
                </div>
              </aside>
            </section>

            <!-- New HTML5 footer tag -->
            <footer>
              <p>This Maven project is to be deployed on JBoss EAP.<br/></p>
            </footer>
          </div>

  <!-- This is the Header row for the browser view.  If the user is on a mobile devise do not use this header,
    instead add a header to each row. -->
          <script type="text/template" id="cardinfo-Header-Row-tmpl">
          <%
                  if (!window.loadJQM) {
          %>
                          <div class="row cardinfo">
                                  <div class="col">
                                          <div class="head">Id</div>
                                  </div>
                                  <div class="col">
                                          <div class="head">ImageId</div>
                                  </div>
                                  <div class="col">
                                          <div class="head">Category</div>
                                  </div>
                                  <div class="col">
                                          <div class="head">Subcategory</div>
                                  </div>
                                  <div class="col">
                                          <div class="head">REST URL</div>
                                  </div>
                          </div>
          <%
          }
          %>
          </script>
          <script type="text/template" id="cardinfo-Body-tmpl">
          <% var addHeader = false;
          if (window.loadJQM) {
                  addHeader = true;
          }
          %>
                  <div class="col">
                          <% if ( addHeader ) { %><div class="head">Id</div><% } %>
        <div class="data"><%=idGcCardInfo%></div>
                  </div>
                  <div class="col">
                          <% if ( addHeader ) { %><div class="head">ImageId</div><% } %>
        <div class="data"><%=idMedia%></div>
                  </div>
                  <div class="col">
                          <% if ( addHeader ) { %><div class="head">Category</div><% } %>
        <div class="data"><%=idGcCategory%></div>
                  </div>
                  <div class="col">
                          <% if ( addHeader ) { %><div class="head">Subcategory</div><% } %>
        <div class="data"><%=idGcSubcategory%></div>
                  </div>
                  <div class="col">
                          <% if ( addHeader ) { %><div class="head">REST URL</div><% } %>
        <div class="data">
                                  <a href="rest/cardinfos/<%=idGcCardInfo%>" rel="external" target="_blank"
class="resturl ui-link">JSON</a>
                          </div>
                  </div>
      <% addHeader = false;%>
          </script>

          <script type="text/template" id="quote-Header-Row-tmpl">
          <%
                  if (!window.loadJQM) {
          %>
```

```
                        <div class="row quote">
                                <div class="col">
                                        <div class="head">Id</div>
                                </div>
                                <div class="col">
                                        <div class="head">Quote</div>
                                </div>
                                <div class="col">
                                        <div class="head">Type</div>
                                </div>
                                <div class="col">
                                        <div class="head">REST URL</div>
                                </div>
                        </div>
        <%
                }
        %>
        </script>
        <script type="text/template" id="quote-Body-tmpl">
        <% var addHeader = false;
        if (window.loadJQM) {
                addHeader = true;
        }
        %>
                <div class="col">
                        <% if ( addHeader ) { %><div class="head">Id</div><% } %>
      <div class="data"><%=idGcQuotes%></div>
                </div>
                <div class="col">
                        <% if ( addHeader ) { %><div class="head">Quote</div><% } %>
      <div class="data"><%=spruchText%></div>
                </div>
                <div class="col">
                        <% if ( addHeader ) { %><div class="head">Type</div><% } %>
      <div class="data"><%=spruchType%></div>
                </div>
                <div class="col">
                        <% if ( addHeader ) { %><div class="head">REST URL</div><% } %>
      <div class="data">
                <a href="rest/quotes/<%=idGcQuotes%>" rel="external" target="_blank" class="resturl ui-
link">JSON</a>
                </div>
                </div>
        <% addHeader = false;%>
        </script>
</body>
</html>
```

Die CSS3-Styles:

d.css:

```
/* Core styles for the page */
body {
    background-color: #F1F1F1;
    font-family: "Lucida Sans Unicode", "Lucida Grande", sans-serif;
    color: #363636;
    text-align: center;
}
#container {
    text-align: left;
    margin: 0 auto;
    padding: 0 20px 10px 20px;
    border-top: 5px solid #000000;
    border-left: 5px solid #8c8f91;
    border-right: 5px solid #8c8f91;
    border-bottom: 25px solid #8c8f91;
    width: 865px; /* subtract 40px from banner width for padding */
    background: #FFFFFF;
    background-image: url(../img/headerbkg.png);
    background-repeat: repeat-x;
    padding-top: 30px;
    box-shadow: 3px 3px 15px #d5d5d5;
}

section {
    clear: both;
```

```
    float: left;
    width: 500px;
    margin: 20px;
}

article {
    clear: both;
}

aside {
    font-size: 0.9em;
    width: 245px;
    float: right;
    margin: 20px 0;
    border: 1px solid #D5D5D5;
    background: #F1F1F1;
    background-image: url(../img/asidebkg.png);
    background-repeat: repeat-x;
    padding: 20px;
}

aside ul {
    padding-left: 20px;
}

.dualbrand {
    float: right;
    padding-right: 10px;
    margin-bottom: 10px;
}

.logo {
    float: left;
    padding-left: 10px;
}

.highlights {
    width: 240px;
    float: left;
}

.highlights ul {
    margin-top: 0;
}

footer {
    clear: both;
    text-align: center;
    color: #666666;
    font-size: 0.85em;
}

a {
    color: #4a5d75;
}

h1 {
    color: #243446;
    font-size: 2.25em;
    margin: 0 0 10px 0;
}

h2 {
    font-size: 1em;
}

h3 {
    color: #243446;
}

/* registration styles */

fieldset {
    padding: 1em;
    font: 80%/1 sans-serif;
    width: 375px;
    border: 1px solid #D5D5D5;
```

```
}

label {
    float: left;
    width: 15%;
    margin-left: 20px;
    margin-right: 0.5em;
    padding-top: 0.2em;
    text-align: right;
    font-weight: bold;
    color: #363636;
}

input {
    margin-bottom: 8px;
}

legend {
    padding: 0.2em 0.5em;
    color: #4a5d75;
    font-size: 1.5em;
}

#register {
    float: left;
    margin-left: 85px;
}

/* Cardinfo table styles */
#cardinfos .row .col:nth-child(1) {
    width: 30px;
}

#cardinfos .row .col:nth-child(4) {
    width: 100px;
}

#cardinfos .row .col:nth-child(5) {
    width: 75px;
}

/* Cardinfo table styles */
#quotes .row .col:nth-child(1) {
    width: 30px;
}

#quotes .row .col:nth-child(4) {
    width: 100px;
}

#quotes .row .col:nth-child(5) {
    width: 75px;
}

/* Responsive Styles */
@media only screen and (max-width: 875px) {
    #container {
        width: 575px;
    }

    aside {
        clear: both;
        float: none;
        width: 510px;
        margin: 20px;
    }
}

/* Using new CSS3 selectors for styling*/
#cardinfos > div:nth-child(odd) {
    background: #f4f3f3;
}

#cardinfos > div:nth-child(even) {
    background: #ffffff;
}
```

```
#quotes > div:nth-child(odd) {
    background: #f4f3f3;
}

#quotes > div:nth-child(even) {
    background: #ffffff;
}

/* Hide Mobile Site Elements */
.header, .footer, .mobileicon, #refreshButtonM, #refreshButtonD, #refreshButtonCl, #features {
    display: none;
}
```

m.screen.css:

```
/* Core styles for the mobile version of the application */
body {
  font-family: Verdana, sans-serif;
}

p {
  margin-left: 8px;
  margin-right: 8px;
}

a {
  color: #369;
  text-decoration: none;
}

/* Header and footer styles */

.header_logo{
  float: left;
  margin-top: 3px;
  margin-left: 8px;
}

.footer_left {
  float: left;
  margin-left: 8px;
}

.footer_right{
  float: right;
  margin-right: 8px;
}

.footer a:hover{
  text-decoration: none;
}

.mobileicon {
    float: right;
    padding: 0 10px 0 10px;
}

/*- utils -*/
.max-width-100{
    max-width: 100%;
}


/* registration styles */

/* override default jquery mobile styles
 * This was necessary because the current jQuery Mobile Themeroller seems to
 * have a bug that will not allow editing the default theme and any theme it
 * generates is not compatible with 1.0.1
 */
.ui-bar-a{
  background: #71A1D0;
  background-image: -webkit-gradient(linear, left top, left bottom, from( #81B9EF), to( #6088B0));
  background-image: -webkit-linear-gradient( #81B9EF, #6088B0);
  background-image: -moz-linear-gradient( #81B9EF, #6088B0);
  background-image: -ms-linear-gradient( #81B9EF, #6088B0);
  background-image: -o-linear-gradient( #81B9EF, #6088B0);
```

```
  background-image: linear-gradient( #81B9EF, #6088B0);
}

.ui-body-a .ui-link {
  color: #E25027;
  font-weight: bold;
}

.ui-body-a .ui-link:visited, .ui-body-a .ui-link:hover {
  color: #E25027;
}

.ui-btn-active {
  border: 1px solid #155678;
  background: #71A1D0;
  background-image: -webkit-gradient(linear, left top, left bottom, from( #71A1D0), to( #71A1D0));
  background-image: -webkit-linear-gradient( #71A1D0, #71A1D0);
  background-image: -moz-linear-gradient( #71A1D0, #71A1D0);
  background-image: -ms-linear-gradient( #71A1D0, #71A1D0);
  background-image: -o-linear-gradient( #71A1D0, #71A1D0);
  background-image: linear-gradient( #71A1D0, #71A1D0);
}

.ui-btn-up-a {
  border: 1px solid #155678;
  background: #3E73A8;
  background-image: -webkit-gradient(linear, left top, left bottom, from( #3E73A8), to( #3E73A8));
  background-image: -webkit-linear-gradient( #3E73A8, #3E73A8);
  background-image: -moz-linear-gradient( #3E73A8, #3E73A8);
  background-image: -ms-linear-gradient( #3E73A8, #3E73A8);
  background-image: -o-linear-gradient( #3E73A8, #3E73A8);
  background-image: linear-gradient( #3E73A8, #3E73A8);
}

.ui-btn-hover-a {
  border: 1px solid #155678;
  background: #E25027;
  background-image: -webkit-gradient(linear, left top, left bottom, from( #E25027), to( #E25027));
  background-image: -webkit-linear-gradient( #E25027, #E25027);
  background-image: -moz-linear-gradient( #E25027, #E25027);
  background-image: -ms-linear-gradient( #E25027, #E25027);
  background-image: -o-linear-gradient( #E25027, #E25027);
  background-image: linear-gradient( #E25027, #E25027);
}

.ui-btn-hover-a, .ui-btn-hover-a .ui-btn-inner .ui-btn-text {
  color: #FFF;
}

.ui-body-a, .ui-dialog.ui-overlay-a {
  color: #333;
  text-shadow: none;
  background: #F0F0F0;
  background-image: -webkit-gradient(linear, left top, left bottom, from( #EEE), to( #DDD));
  background-image: -webkit-linear-gradient( #EEE, #DDD);
  background-image: -moz-linear-gradient( #EEE, #DDD);
  background-image: -ms-linear-gradient( #EEE, #DDD);
  background-image: -o-linear-gradient( #EEE, #DDD);
  background-image: linear-gradient( #EEE, #DDD);
}

.ui-mobile fieldset{
  padding-left: 5px;
  padding-right: 5px;
}

/* Cardinfo table styles */
#cardinfos .row .col .head {
   float: left;
   width: 70px;
   border: none;
   padding: 4px 5px;
}

#cardinfos .row .col {
   width: 100%;
   float: left;
```

```
    clear: left;
    background-color: #fff;
    border: 1px dotted #ccc;
}

#cardinfos .row .col .data {
    padding: 4px 4px 4px 94px;
}

#cardinfos .col:nth-child(5) {
    margin-bottom: 20px;
}

#cardinfo-art {
    padding-top: 30px!important;
}

#cardinfo-art .ui-content .ui-btn {
    float: right;
}

/* Cardinfo table styles */
#quotes .row .col .head {
    float: left;
    width: 70px;
    border: none;
    padding: 4px 5px;
}

#quotes .row .col {
    width: 100%;
    float: left;
    clear: left;
    background-color: #fff;
    border: 1px dotted #ccc;
}

#quotes .row .col .data {
    padding: 4px 4px 4px 94px;
}

#quotes .col:nth-child(5) {
    margin-bottom: 20px;
}

#quote-art {
    padding-top: 30px!important;
}

#quote-art .ui-content .ui-btn {
    float: right;
}

/* Hide column header rows if present */
.columnNames {
    display: none;
}

/* Hide Desktop Items */
section > h1, article > h2, aside > h3, footer, .cardinfo-foot .quote-foot .ui-btn, .highlights, .logo, .dualbrand {
            display: none;
}
```

screen.css:

```
/* Core styles for the page */
body {
    margin: 0;
    padding: 0;
    font-size: 0.8em;
    color: #363636;
}

code {
    font-size: 1.1em;
}
```

```
a {
    text-decoration: none;
}

a:hover {
    color: #369;
    text-decoration: underline;
}

/* registration styles */
span.invalid {
    padding-left: 3px;
    color: red;
}

span.success {
   padding-left: 3px;
   color: green;
}

/* Cardinfo table styles */
#cardinfos {
    display: table;
    width: 100%;
}

#cardinfos .row {
    display: table-row;
}

#cardinfos .row .col {
    display: table-cell;
    width: auto;
    border: 1px solid #ccc;
}

#cardinfos .head {
    color: #FFFFFF;
    background-color: #000000;
    padding: 1px 5px;
    border-bottom: 1px solid #ccc;
}

#cardinfos .data {
    padding: 1px 5px;
    font-size: 11px;
}

.cardinfo-foot {
    clear: both;
    height: 20px;
    font-weight: bold;
}

/* Quotes table styles */
#quotes {
    display: table;
    width: 100%;
}

#quotes .row {
    display: table-row;
}

#quotes .row .col {
    display: table-cell;
    width: auto;
    border: 1px solid #ccc;
}

#quotes .head {
    color: #FFFFFF;
    background-color: #000000;
    padding: 1px 5px;
    border-bottom: 1px solid #ccc;
}
```

```
#quotes .data{
   padding: 1px 5px;
   font-size: 11px;
}

.quote-foot {
   clear: both;
   height: 20px;
   font-weight: bold;
}
```

Die Test-Frameworks und die Tests

a) QUnit Tests werden folgendermaßen aufgerufen: z.B. über den Link in der index.html

http://localhost:8080/greetingcards/test/qunit/index.html

Das /test-Verzeichnis neben dem /WEB-INF-Verzeichnis des **greetingcards.war** hat hierfür folgende Struktur:

```
- test
   - qunit
      index.html
      - qunit
         qunit.css
         qunit.js
      - test
         test.js
```

Die test.js enthält folgende Test-Suite:

```
/*
   Unit tests that cover basic functionality of app.js, using service calls
 */
module('Test Suite: Cardinfo and Quote asynch RESTful calls');

asyncTest('Request quotes json, cardinfos json', function() {

   expect(1);

   $('#results-rest-quotes').load('http://localhost:8080/greetingcards/rest/quotes', function(response,
   status, xhr)
   {
      if (status !== "error") {
            document.getElementById('results-restful-quotes-status-ok').style.visibility = "visible";
      } else {
            document.getElementById('results-restful-quotes-status-error').style.visibility = "visible";
      }
      $('#results-rest-cardinfos').load('http://localhost:8080/greetingcards/rest/cardinfos',
      function(response, status, xhr) {
            if (status !== "error") {
                  document.getElementById('results-restful-cardinfos-status-ok').style.visibility = "visible";
            } else {
                  document.getElementById('results-restful-cardinfos-status-error').style.visibility = "visible";
            }
      });
   });
});
```

Für die QUnit Tests sieht die **index.html** im /test-Verzeichnis folgendermaßen aus:

```
<!DOCTYPE html>
<!--
Base test file for running quit tests.  Load this into any browser to execute
the tests for the HTML5/Mobile application.
-->
<html>
<head>
            <meta charset="UTF-8" />
            <title>HTML5 Test Suite</title>

   <link rel="stylesheet" href="qunit/qunit.css" type="text/css" media="screen">
   <script type="text/javascript" src="../../js/libs/jquery-1.9.1.min.js"></script>
   <script type="text/javascript" src="../../js/libs/backbone-1.0.0.js"></script>
   <script type="text/javascript" src="../../js/libs/underscore-min.js"></script>
   <script type="text/javascript" src="qunit/qunit.js"></script>

   <!-- The js file to be tested -->
```

```
    <!-- The js file that contains the tests -->
    <script type="text/javascript" src="test/test.js"></script>

        <!-- Underscore.js Template for building cardinfos (simplified for testing) -->
        <script type="text/template" id="cardinfo-tmpl">
<% _.each(cardinfos, function(cardinfo) { %><div><%=cardinfo.idGcCardInfo%></div><% }); %></script>
        <!-- Underscore.js Template for building quotes (simplified for testing) -->
        <script type="text/template" id="quote-tmpl">
<% _.each(quotes, function(quote) { %><div><%=quote.idGcQuotes%></div><% }); %></script>

</head>
<body>
        <h1 id="qunit-header">HTML5 Test Suite  
        <a href="/greetingcards/test/qunit/index.html">Run</a>  
        <a href="../../index.html">Home</a>
        </h1>

        <h2 id="qunit-banner"></h2>
        <div id="qunit-testrunner-toolbar"></div>
        <h2 id="qunit-userAgent"></h2>
        <ol id="qunit-tests"></ol>
        <div id="qunit-fixture">QUnit Tests</div>

        Quotes JSON result:
        <div id="results-rest-quotes"></div>
        <div id="results-restful-quotes-status-ok" style="background-color:green;font-
weight:bold;visibility:hidden">Test quotes ok.</div>
        <div id="results-restful-quotes-status-error" style="background-color:red;font-
weight:bold;visibility:hidden">Test quotes in error.</div>

        Cardinfos JSON result:
        <div id="results-rest-cardinfos"></div>
        <div id="results-restful-cardinfos-status-ok" style="background-color:green;font-
weight:bold;visibility:hidden">Test cardinfos ok.</div>
        <div id="results-restful-cardinfos-status-error" style="background-color:red;font-
weight:bold;visibility:hidden">Test cardinfos in error.</div>
</body>
</html>
```

b) Selenium Tests:

Installation: In der Test-Browser Instanz die folgende Installations-Url aufrufen und der Installation zustimmen:
http://release.seleniumhq.org/selenium-ide/2.9.0/selenium-ide-2.9.0.xpi

Selenium-Blog:
http://blog.reallysimplethoughts.com/2015/03/09/selenium-ide-scheduler-has-arrived-part-1/

Selenium Download:
http://docs.seleniumhq.org/download/

Selenium Documentation:
http://docs.seleniumhq.org/docs/
https://seleniumhq.github.io/docs/
http://seleniumhq.github.io/selenium/docs/api/java/index.html

Die **Tests** werden folgendermaßen angelegt, nachdem das Selenium IDE PlugIn installiert wurde. Dabei sollten folgende Grundregeln beachtet werden:

- Tests sollten immer eine definierte Start-Url haben, z.B. hier: **http://localhost:8080/greetingcards/**
- Tests sollten nicht von anderen Tests abhängig sein.
- Tests sollten immer nur eine Sache/einen kleinen Workflow je Test Case testen,
  z.B. die Registrierung eines Quotes oder genau einer Card.
- Test Cases können in einer Test Suite zusammengefasst werden, um den Automatisierungsgrad zu erhöhen.

Bei dieser Gelegenheit darf auch auf das sehr effektive und interessante Seminar

"**TDD mit Java**" **hier** und **hier** von Binaris Informatik hingewiesen werden.

Hier nun die zwei Test-Suites, die mit dem Selenium IDE PlugIn durchgeführt wurden und automatisiert ausgeführt werden können:

1. greetingcards_link_test_suite_selenium enthält folgende Test Cases:

- **selenium_test_link_Home.html**:
  <?xml version="1.0" encoding="UTF-8"?>

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<head profile="http://selenium-ide.openqa.org/profiles/test-case">
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    <link rel="selenium.base" href="http://localhost:8080/greetingcards/" />
    <title>selenium_test_link_Home</title>
</head>

<body>
<table cellpadding="1" cellspacing="1" border="1">
<thead>
<tr>
    <td rowspan="1" colspan="3">selenium_test_link_Home</td>
</tr>
</thead>
<tbody>
<tr>
    <td>open</td>
    <td>/greetingcards/</td>
    <td></td>
</tr>
<tr>
    <td>click</td>
    <td>//article[@id='intro-art']/div[3]/div/a/span/span</td>
    <td></td>
</tr>
</tbody>
</table>
</body>
</html>
```

- selenium_test_link_Info_Open.html:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<head profile="http://selenium-ide.openqa.org/profiles/test-case">
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    <link rel="selenium.base" href="http://localhost:8080/greetingcards/" />
    <title>selenium_test_link_Info_Open</title>
</head>

<body>
<table cellpadding="1" cellspacing="1" border="1">
<thead>
<tr>
    <td rowspan="1" colspan="3">selenium_test_link_Info_Open</td>
</tr>
</thead>
<tbody>
<tr>
    <td>open</td>
    <td>/greetingcards/</td>
    <td></td>
</tr>
<tr>
    <td>click</td>
    <td>//article[@id='intro-art']/div/a/span/span[2]</td>
    <td></td>
</tr>
<tr>
    <td>click</td>
    <td>//aside[@id='info-aside']/div/div/a/span/span[2]</td>
    <td></td>
</tr>
</tbody>
</table>
</body>
</html>
```

- selenium_test_link_QUnit_Tests.html:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<head profile="http://selenium-ide.openqa.org/profiles/test-case">
```

```
		<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
		<link rel="selenium.base" href="http://localhost:8080/greetingcards/" />
		<title>selenium_test_link_QUnit_Tests</title>
	</head>
	<body>
	<table cellpadding="1" cellspacing="1" border="1">
	<thead>
	<tr>
		<td rowspan="1" colspan="3">selenium_test_link_QUnit_Tests</td>
	</tr>
	</thead>
	<tbody>
	<tr>
		<td>open</td>
		<td>/greetingcards/</td>
		<td></td>
	</tr>
	<tr>
		<td>clickAndWait</td>
		<td>//article[@id='intro-art']/div[3]/div[2]/a/span/span</td>
		<td></td>
	</tr>
	</tbody>
	</table>
	</body>
	</html>
```

- selenium_test_list_card.html:

```
	<?xml version="1.0" encoding="UTF-8"?>
	<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
	strict.dtd">
	<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
	<head profile="http://selenium-ide.openqa.org/profiles/test-case">
		<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
		<link rel="selenium.base" href="http://localhost:8080/greetingcards/" />
		<title>selenium_test_list_card</title>
	</head>
	<body>
	<table cellpadding="1" cellspacing="1" border="1">
	<thead>
	<tr>
		<td rowspan="1" colspan="3">selenium_test_list_card</td>
	</tr>
	</thead>
	<tbody>
	<tr>
		<td>open</td>
		<td>/greetingcards/</td>
		<td></td>
	</tr>
	<tr>
		<td>click</td>
		<td>//article[@id='intro-art']/div[3]/div[2]/a[5]/span/span</td>
		<td></td>
	</tr>
	</tbody>
	</table>
	</body>
	</html>
```

- selenium_test_list_quote.html:

```
	<?xml version="1.0" encoding="UTF-8"?>
	<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
	strict.dtd">
	<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
	<head profile="http://selenium-ide.openqa.org/profiles/test-case">
		<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
		<link rel="selenium.base" href="http://localhost:8080/greetingcards/" />
		<title>selenium_test_list_quote</title>
	</head>
	<body>
	<table cellpadding="1" cellspacing="1" border="1">
	<thead>
	<tr>
		<td rowspan="1" colspan="3">selenium_test_list_quote</td>
	</tr>
	</thead>
```

```
<tbody>
<tr>
	<td>open</td>
	<td>/greetingcards/</td>
	<td></td>
</tr>
<tr>
	<td>click</td>
	<td>//article[@id='intro-art']/div[3]/div[2]/a[7]/span</td>
	<td></td>
</tr>
</tbody>
</table>
</body>
</html>
```

2. greetingcards_register_test_suite_selenium enthält folgende Test Cases:

- selenium_test_register_card.html:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<head profile="http://selenium-ide.openqa.org/profiles/test-case">
	<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
	<link rel="selenium.base" href="http://localhost:8080/greetingcards/" />
	<title>selenium_test_register_card</title>
</head>
<body>
<table cellpadding="1" cellspacing="1" border="1">
<thead>
<tr>
    <td rowspan="1" colspan="3">selenium_test_register_card</td>
</tr>
</thead>
<tbody>
<tr>
	<td>open</td>
	<td>/greetingcards/</td>
	<td></td>
</tr>
<tr>
	<td>click</td>
	<td>//article[@id='intro-art']/div[3]/div[2]/a[4]/span/span</td>
	<td></td>
</tr>
<tr>
	<td>type</td>
	<td>id=idMedia</td>
	<td>123</td>
</tr>
<tr>
	<td>type</td>
	<td>id=idGcCategory</td>
	<td>5</td>
</tr>
<tr>
	<td>type</td>
	<td>id=idGcSubcategory</td>
	<td>10</td>
</tr>
<tr>
	<td>click</td>
	<td>id=register</td>
	<td></td>
</tr>
<tr>
	<td>click</td>
	<td>link=QUnit Tests</td>
	<td></td>
</tr>
</tbody>
</table>
</body>
</html>
```

- selenium_test_register_quote.html:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<head profile="http://selenium-ide.openqa.org/profiles/test-case">
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
        <link rel="selenium.base" href="http://localhost:8080/greetingcards/" />
        <title>selenium_test_register_quote</title>
</head>
<body>
<table cellpadding="1" cellspacing="1" border="1">
<thead>
<tr>
    <td rowspan="1" colspan="3">selenium_test_register_quote</td>
</tr>
</thead>
<tbody>
<tr>
        <td>open</td>
        <td>/greetingcards/</td>
        <td></td>
</tr>
<tr>
        <td>click</td>
        <td>//article[@id='intro-art']/div[3]/div[2]/a[6]/span/span</td>
        <td></td>
</tr>
<tr>
        <td>type</td>
        <td>id=spruchText</td>
        <td>Man soll den Tag nicht vor dem Abend loben.</td>
</tr>
<tr>
        <td>type</td>
        <td>id=spruchType</td>
        <td>k</td>
</tr>
<tr>
        <td>click</td>
        <td>xpath=(//input[@id='register'])[2]</td>
        <td></td>
</tr>
<tr>
        <td>click</td>
        <td>link=QUnit Tests</td>
        <td></td>
</tr>
</tbody></table>
</body>
</html>
```

Hier nun auch das komplette Web-Archiv **greetingcards.war** zum Download:
http://www.4shared.com/file/ioRJrzycba/greetingcards.html

Und hier die kompletten Selenium Tests für die Beispiel-Applikation:
http://www.4shared.com/zip/ajxRguQVba/selenium_tests.html

Hier den **JBoss EAP 6.3 (JBoss AS 7.4)** als Jar-Installer zum Download.
http://www.jboss.org/download-manager/file/jboss-eap-6.3.0.GA-installer.jar

Installation/Start der Beispiel-Applikation:
- Das JBoss Zip-Archiv entpacken
- JAVA_HOME, JBOSS_HOME entsprechend setzen und
  in der standalone.bat oder standalone.sh verwenden.
- greetingcard.war ins Server-Unterverzeichnis /standalone/deployments speichern
- Neben greetingcard.war eine leere Textdatei namens greetingcards.war.dodeploy speichen.
- Server starten per standalone.bat oder standalone.sh, http://localhost:8080/greetingcards/ aufrufen
- Die Tests starten:
  - QUnit Tests per Schaltfläche 'QUnit Tests'
  - Selenium Tests per Selenium IDE PlugIn