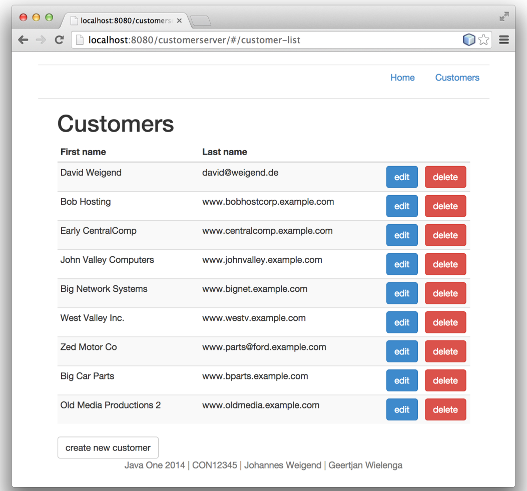
JEE + Angular - CRUD Demo

Java One 2014

Johannes Weigend, 21.09.2014



localhost:8080/customerserver/#/customer-list

Customers

First name	Last name		
David Weigend	david@weigend.de	edit	delete
Bob Hosting	www.bobhostcorp.example.com	edit	delete
Early CentralComp	www.centralcomp.example.com	edit	delete
John Valley Computers	www.johnvalley.example.com	edit	delete
Big Network Systems	www.bignet.example.com	edit	delete
West Valley Inc.	www.westv.example.com	edit	delete
Zed Motor Co	www.parts@ford.example.com	edit	delete
Big Car Parts	www.bparts.example.com	edit	delete
Old Media Productions 2	www.oldmedia.example.com	edit	delete

Java One 2014 | CON12345 | Johannes Weigend | Geertjan Wielenga

This demo shows a simple CRUD HTML5 Application developed with Netbeans 7, Glassfish 4 and AngularJS. We use a JEE REST Server Backend and a HTML/JS Frontend. The HTML Frontend communicates with REST Services. The Server is stateless.

In the demo I will show 3 steps.

1) I create a REST Backend with Database Access by using Netbeans, Maven and Glassfish
2) I create a HTML5 Frontend with the YEOMAN Angular Generator and show Debugging and Development of a Angular Single Page Application
3) Integrating everything together by adding the HTML5 Application as a Maven Ressource using the Maven WAR Plugin.

Step 1

1. Create a new maven based web application in Netbeans (Project->New->Maven->Web Application).
2. Use the „Create REST Services from Database“ Wizzard to create REST Services direct by connecting to the sample derby database.
3. Test the „Count“ Service by clicking the method in the Projects/Webservices Section
4. Create a „Cross Origin Resource Sharing Filter (CORS)“ in Netbeans to make sure the external HTML5 app can access our Webservice (After step 3 - the filter could be removed for production)

Step 2

1. We use the **yo** angular generator to generate a Angular HTML5 application.

```
$> yo angular
```

Prerequisite: Make sure **yeoman** is installed. After successful generation you can use grunt to show the generated sources in your browser. „grunt serve“ starts a simple NodeJS server which is a good first test.

```
$> grunt serve  #opens the browser and shows the generated
page
```

2. We are ready to open the generated project in Netbeans. We use the „HTML Project from existing sources“ type of project.

3. The Netbeans project has some errors because the generated structure is not direct supported. We can fix this easily be changing the directory in the .bowerrc file from bower-components to app/bower-components.

4. After successful editing of the bower component file we kann download the **bower** libraries by clicking „bower install“ direct to the app directory in the project tree.

5. Now we have professional template for our project. The template uses **grunt** as build automation tool. It uses bower as library download tool. It has the standard angular structure with unit tests and integration tests.

6. We now want to have two views. A list view to display the list of customers. A detail view to update or create a single customer. With the Yeoman generator we can create the two views inclusive the corresponding controllers. The reference to the controller code is automatically added to the index.html page.

```
$> yo angular:route customer-list
$> yo angular:route customer-detail
```

7. To make a first test we edit customer-list.html to loop over the generated array in the controller.

```
<div ng-repeat="thing in awesomeThings">
    {{ thing }}
</div>
```

8. We change the awesomeThings array of string to an array of customer objects which contains a name and an email property.

// controller - (Editor: `customers+TAB`)

```
$scope.customers = [{
              name: "Johannes Weigend",
              email: "johannes.weigend@weigend.de"
          }, {
              name: "Jonathan Weigend",
              email: "jonathan.weigend@weigend.de"
          }];
```

// view

```
<tr ng-repeat="customer in customers">
          <td>{{ customer.name }}</td>
          <td>{{ customer.email }}</td>
          …
```

9. The complete HTML Code looks like this - (Editor: `customertable + TAB`)

```
<div class="col-xs-12">
    <h1>Customers</h1>
    <table class="table table-striped table-condensed">
        <thead>
            <tr>
                <th style="min-width: 80px;">First name</th>
                <th style="min-width: 80px;">Last name</th>
                <th style="width:20px;">&nbsp;</th>
                <th style="width:20px;">&nbsp;</th>
            </tr>
        </thead>
        <tbody>
            <tr ng-repeat="customer in customers">
                <td>{{ customer.name }}</td>
                <td>{{ customer.email }}</td>
                <td><a class="btn btn-small btn-primary">edit</a></td>
                <td><a class="btn btn-small btn-danger">delete</a></td>
            </tr>
        </tbody>
    </table>
    <a class="btn btn-default">create new customer</a>
</div>
```

10. Now we are ready to connect to our REST service.

First create a customer-svc factory.

```
$> yo angular:factory customer-svc
```

We have to change the generated customer-svc.js file to return a angular resource object to access our webservice. (Editor: `customerServices + TAB`)

```
angular.module('customerServices', ['ngResource'])
        .factory('customerSvc', ['$resource',
            function ($resource) {
                return $resource(
                        'http://localhost:8080/customerserver/webresources/
com.javaone.customerserver.customer/:customerId:id',
                        {},
                        {}
                        });
            }]);
```

c) We also have to add the new module „customerServices“ to our list of module dependencies of our application. Otherwise the module will not be accessible in our controller.

```
angular
  .module('demoClientApp', [
    'ngAnimate',
    'ngCookies',
    'ngResource',
    'ngRoute',
    'ngSanitize',
    'ngTouch',
    'customerServices'
  ])
```

The complete code is available on github:

https://github.com/jweigend/JEEAngularCRUD

Step 3

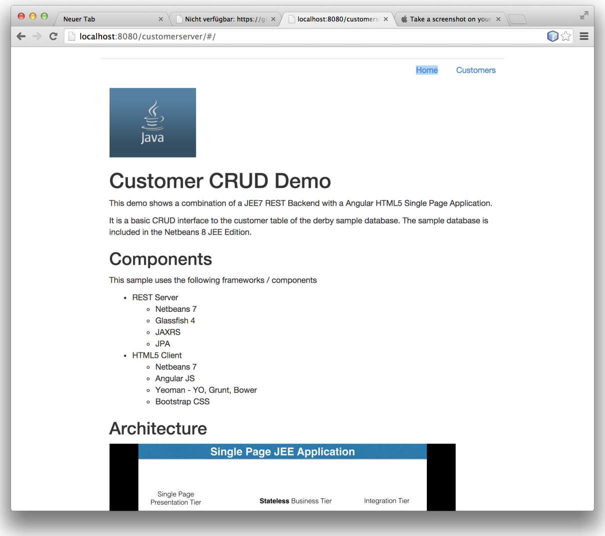
To include the HTML Project in our JEE Webapp we use the Maven WAR Plugin:

(Editor: `webresources+TAB`)

```
 <plugin>
     <groupId>org.apache.maven.plugins</groupId>
     <artifactId>maven-war-plugin</artifactId>
```

```
        <version>2.3</version>
        <configuration>
            <failOnMissingWebXml>false</failOnMissingWebXml>
            <webResources>
               <resource>
                   <!-- this is relative to the pom.xml directory -->
                   <directory>../CustomerClient/app</directory>
               </resource>
            </webResources>
        </configuration>
</plugin>
```

After new maven build and deployment the page shows up when you start the Glassfish Server:


localhost:8080/customerserver/#/

Home Customers

Customer CRUD Demo

This demo shows a combination of a JEE7 REST Backend with a Angular HTML5 Single Page Application.

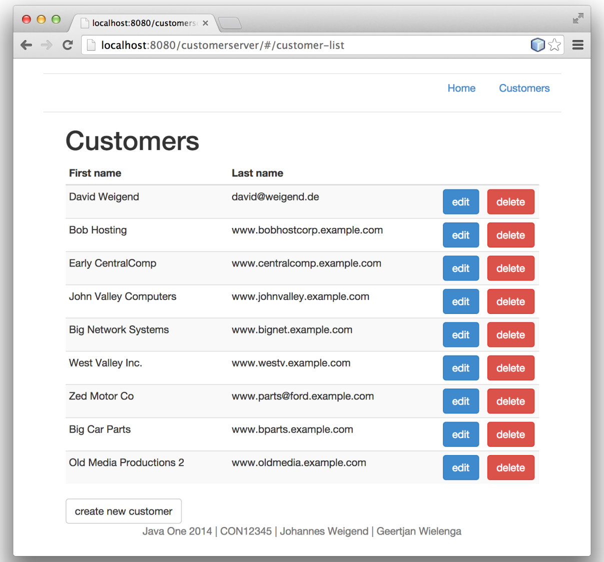
It is a basic CRUD interface to the customer table of the derby sample database. The sample database is included in the Netbeans 8 JEE Edition.

Components

This sample uses the following frameworks / components

- REST Server
  - Netbeans 7
  - Glassfish 4
  - JAXRS
  - JPA
- HTML5 Client
  - Netbeans 7
  - Angular JS
  - Yeoman - YO, Grunt, Bower
  - Bootstrap CSS

Architecture


localhost:8080/customerserver/#/customer-list

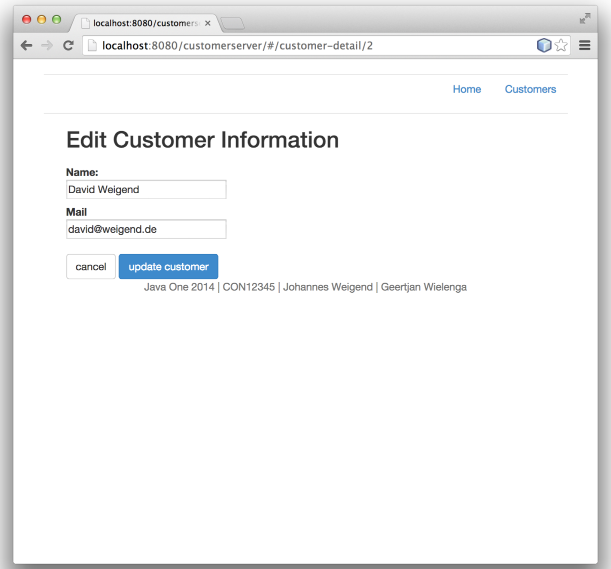
Home Customers

Customers

First name	Last name		
David Weigend	david@weigend.de	edit	delete
Bob Hosting	www.bobhostcorp.example.com	edit	delete
Early CentralComp	www.centralcomp.example.com	edit	delete
John Valley Computers	www.johnvalley.example.com	edit	delete
Big Network Systems	www.bignet.example.com	edit	delete
West Valley Inc.	www.westv.example.com	edit	delete
Zed Motor Co	www.parts@ford.example.com	edit	delete
Big Car Parts	www.bparts.example.com	edit	delete
Old Media Productions 2	www.oldmedia.example.com	edit	delete

create new customer

Java One 2014 | CON12345 | Johannes Weigend | Geertjan Wielenga


localhost:8080/customerserver/#/customer-detail/2

Home Customers

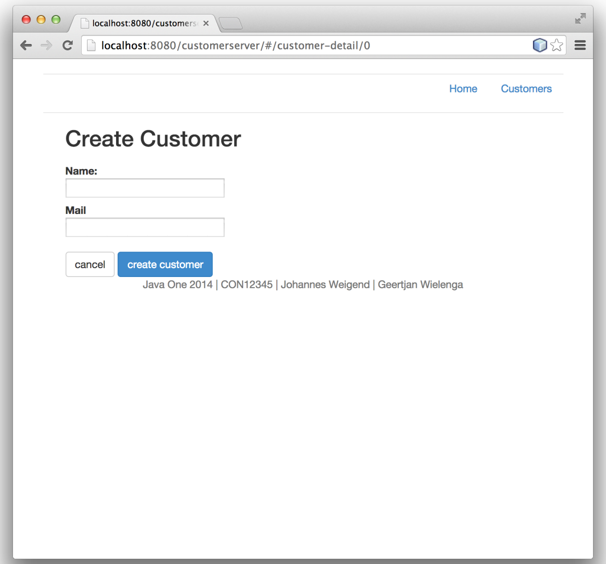
Edit Customer Information

Name: David Weigend

Mail david@weigend.de

cancel update customer

Java One 2014 | CON12345 | Johannes Weigend | Geertjan Wielenga


localhost:8080/customerserver/#/customer-detail/0

Home Customers

Create Customer

Name:

Mail

cancel create customer

Java One 2014 | CON12345 | Johannes Weigend | Geertjan Wielenga