MATHEMA campus 2016

MVC – Nachwuchs im Hause Java EE

Florian Hirsch / adorsys

florian.hirsch@adorsys.de
twitter.com/lefloh / github.com/lefloh

Let‘s build a Java EE Web Application!

Servlets and JSPs?

Portlets?

JSF?

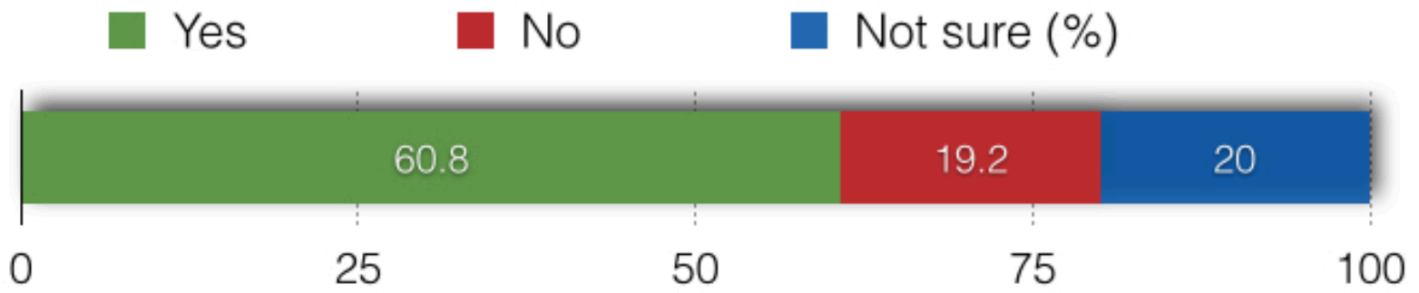
JAX-RS?

RESTful Websites?

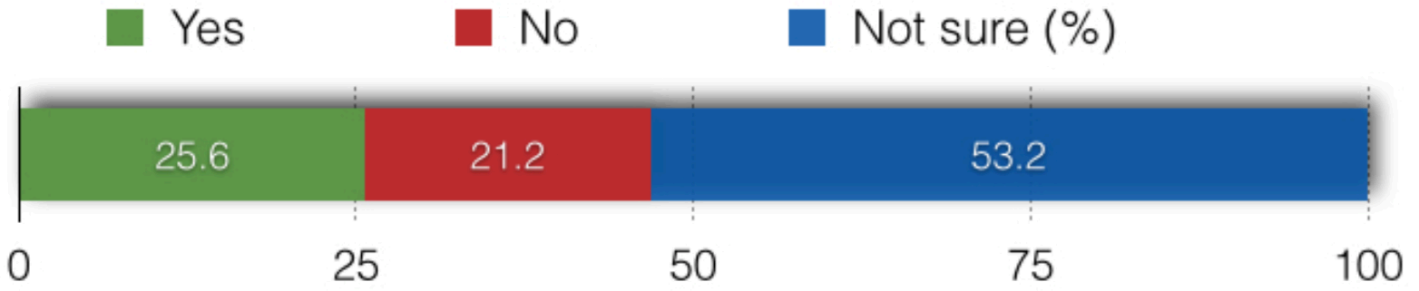
Java EE 8 Community Survey

MVC Support

Should Java EE provide support for MVC, alongside JSF?



Is there any one de-facto standard technology in this space to which we should look for inspiration?



[Results from the Java EE 8 Community Survey]

Goals

- Leverage existing Java EE technologies
- Integrate with CDI and Bean Validation
- Define a solid core to build MVC applications without necessarily supporting all the features in its first version
- Explore layering on top of JAX-RS for the purpose of re-using its matching and binding layers
- Provide built-in support for JSPs and Facelets view languages

MVC Specification (Early Draft Review 2)

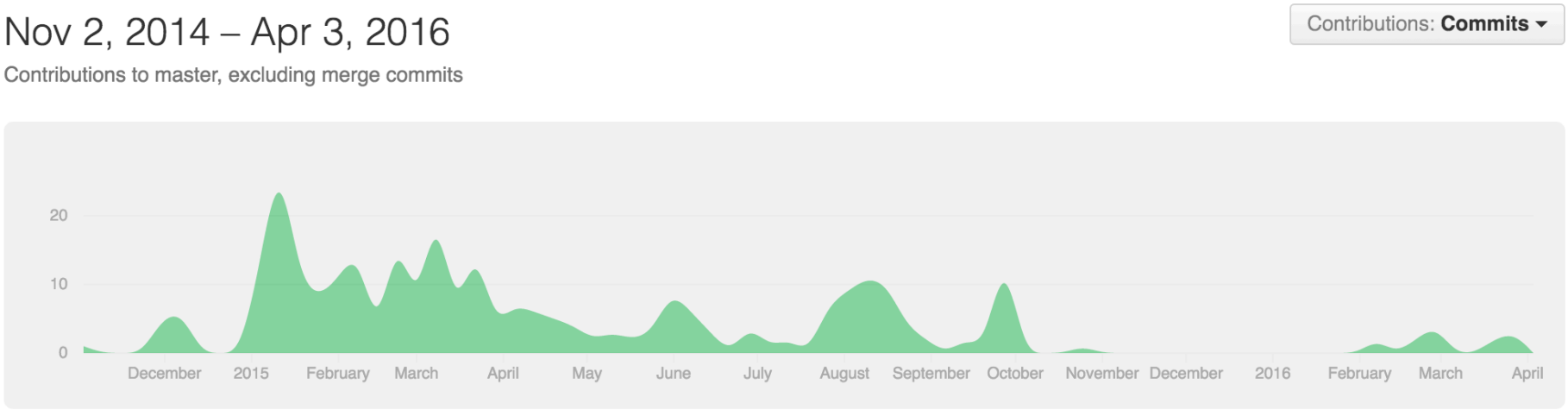
Non Goals

- Define a new view (template) language and processor
- Support standalone implementations of MVC running outside of Java EE
- Support REST services not based on JAX-RS
- Provide built-in support for view languages that are not part of Java EE

MVC Specification (Early Draft Review 2)

Status

Commits to Ozark (Reference Implementation)



JSF Controller

```java
@Named
@RequestScoped
public class CustomerController {

   @Inject
   private CustomerVO vo;

   @Inject
   private CustomerDAO dao;

   public String showCustomer(long id) {
      Customer customer = dao.findCustomer(id);
      vo.setCustomer(customer);
      return "customer.xhtml";
   }

}
```

AngularJS Controller

```
app.controller('customerCtrl', function($scope, $http) {
  $scope.showCustomer = function(id) {
    $http.get('/api/customers/' + id).then(function(response) {
      $scope.customer = response.data;
    });
  };
});
```

AngularJS Controller + JAX-RS Resource

```
app.controller('customerCtrl', function($scope, $http) {
  $scope.showCustomer = function(id) {
    $http.get('/api/customers/' + id).then(function(response) {
      $scope.customer = response.data;
    });
  };
});
```

```
@Path("customers")
public class CustomerController {

   @Inject
   private CustomerDAO dao;

   @Path("{id}")
   public Customer showCustomer(@PathParam("id") Long id) {
      return dao.findCustomer(id);
   }

}
```

MVC Controller

```
@Controller
@Path("customers")
public class CustomerController {

    @Inject
    private Models models;

    @Inject
    private CustomerDAO dao;

    @Path("{id}")
    public String showCustomer(@PathParam("id") Long id) {
        Customer customer = dao.findCustomer(id);
        models.put("customer", customer);
        return "customer.jsp";
    }

}
```

MVC Controller

```
@Controller
@Path("customers")
public class CustomerController {

    @Inject
    private Models models;

    @Inject
    private CustomerDAO dao;

    @Path("{id}")
    public Response showCustomer(@PathParam("id") Long id) {
        Customer customer = dao.findCustomer(id);
        models.put("customer", customer);
        return Response.ok("customer.jsp").build();
    }

}
```

MVC Controller

```
@Controller
@Path("customers")
public class CustomerController {

    @Inject
    private Models models;

    @Inject
    private CustomerDAO dao;

    @Path("{id}")
    public Viewable showCustomer(@PathParam("id") Long id) {
        Customer customer = dao.findCustomer(id);
        models.put("customer", customer);
        return new Viewable("customer.jsp");
    }

}
```

MVC Controller

```
@Controller
@Path("customers")
public class CustomerController {

    @Inject
    private Models models;

    @Inject
    private CustomerDAO dao;

    @Path("{id}")
    @View("customer.jsp")
    public void showCustomer(@PathParam("id") Long id) {
        Customer customer = dao.findCustomer(id);
        models.put("customer", customer);
    }

}
```

MVC Controller (CDI)

```
@Controller
@Path("customers")
public class CustomerController {

   @Inject
   private Customer customer;

   @Path("{id}")
   public String showCustomer(@PathParam("id") Long id) {
      return "customer.jsp";
   }

}
```

JSF View

```
<ui:composition xmlns="http://www.w3.org/1999/xhtml"
  xmlns:ui="http://java.sun.com/jsf/facelets"
  xmlns:h="http://java.sun.com/jsf/html"
  xmlns:f="http://java.sun.com/jsf/core"
  template="/resources/mainTemplate.xhtml">

  <ui:param name="pageTitle" value="Customers" />

  <ui:define name="main">
   <h:dataTable id="customers" var="customer"
                value="#{customerController.allCustomers}">
    <h:column>
     <f:facet name="header">Name</f:facet>
     <h:commandLink action="#{customerController.showCustomer(customer.id)}">
      <h:outputText value="#{customer.name}" />
     </h:commandLink>
    </h:column>
   </h:dataTable>
  </ui:define>

</ui:composition>
```

AngularJS View

```
<!doctype html>
<html lang="en">
<head>
  <title>Customers</title>
</head>
<body ng-app="myApp">
  <table ng-controller="customerCtrl">
    <tr>
      <th>Name</th>
    </tr>
    <tr ng-repeat="customer in customers">
      <td ng-click="showCustomer(customer.id)">
        {{ customer.name }}
      </td>
    </tr>
  </table>
</body>
</html>
```

MVC View (JSP)

```
<%@ page contentType="text/html; charset=utf-8" language="java" %>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<!doctype html>
<html>
<head>
  <title>Customers</title>
</head>
<body>
<table>
  <tr>
    <th>Name</th>
  </tr>
  <c:forEach var="customer" items="${customers}">
    <tr>
      <td><a href="${mvc.basePath}/customers/${customer.id}">
        ${customer.name}
      </a></td>
    </tr>
  </c:forEach>
</table>
</body>
</html>
```

MVC View (Mustache)

```
<!doctype html>
<html>
<head>
  <title>Customers</title>
</head>
<body>
<body>
<table>
  <tr>
    <th>Name</th>
  </tr>
  {{#customers}}
    <tr>
      <td><a href="{{mvc.basePath}}/customers/{{id}}">{{ name }}</a></td>
    </tr>
  {{/customers}}
</table>
</body>
</html>
```

Other ViewEngines

- AsciiDoc
- Freemarker
- Handlebars
- Jade
- JSR 223
- Mustache
- StringTemplate
- Thymeleaf
- Velocity

linkTo Method

```
<%@ page contentType="text/html; charset=utf-8" language="java" %>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<!doctype html>
<html>
<head>
  <title>Customers</title>
</head>
<body>
<table>
  <tr>
    <th>Name</th>
  </tr>
  <c:forEach var="customer" items="${customers}">
    <tr>
      <td><a href="${mvc.linkTo('CustomerController', 'showCustomer',
                  {'id': customer.id})}">
        ${customer.name}
      </a></td>
    </tr>
  </c:forEach>
</table>
</body>
</html>
```

linkTo Controller

```
<%@ page contentType="text/html; charset=utf-8" language="java" %>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<!doctype html>
<html>
<head>
  <title>Customers</title>
</head>
<body>
<table>
  <tr>
    <th>Name</th>
  </tr>
  <c:forEach var="customer" items="${customers}">
    <tr>
      <td><a href="${mvc.linkTo('show-customer', {'id': customer.id})}">
        ${customer.name}
      </a></td>
    </tr>
  </c:forEach>
</table>
</body>
</html>
```

Content-Type

```java
@Controller
@Path("something")
@Produces("text/html")
public class SomeController {
    ...
}
```

```
<%@ page contentType="text/html; charset=utf-8" language="java" %>
```

Validation

```
@Controller
@Path("customers")
public class ValidationController {

   @Inject
   private BindingResult br;

   @Inject
   private Errors errors;

   @POST
   @ValidateOnExecution(type = ExecutableType.NONE)
   public Response newCustomer(@Valid @BeanParam CustomerForm form) {
      if (br.isFailed()) {
         for (ValidationError error : br.getAllValidationErrors()) {
            errors.add(new Error(error.getParamName(), error.getMessage()));
         }
         return Response.status(BAD_REQUEST)
                        .entity("customer-form.jsp").build();
      }
      return Response.status(OK).entity("customer-view.jsp").build();
   }

}
```

PR 1: ValidationError

```java
/**
 * <p>Represents a single validation error detected for a parameter.
 * A validation error always corresponds to exactly one
 * {@link ConstraintViolation}.</p>
 *
 * @author Christian Kaltepoth
 * @since 1.0
 */
public interface ValidationError {

    String getParamName();

    ConstraintViolation<?> getViolation();

    String getMessage();

}
```

https://github.com/mvc-spec/mvc-spec/pull/1

PR 2: LocaleResolver

```java
/**
 * <p>Locale resolvers are used to determine the locale of the current
 * request and are discovered using CDI.</p>
 *
 * <p>The MVC implementation is required to resolve the locale for each
 * request following this algorithm:</p>
 *
 * ...
 *
 * @author Christian Kaltepoth
 */
public interface LocaleResolver {

    String DEFAULT_LOCALE = "javax.mvc.locale.LocaleResolver.defaultLocale";

    Locale resolveLocale(LocaleResolverContext context);

}
```

https://github.com/mvc-spec/mvc-spec/pull/2

PR 3: Servlet Dependency

Remove Servlet dependencies from MVC API?

https://github.com/mvc-spec/mvc-spec/pull/3

PR 4: OSGI Module

Package MVC 1.0 as OSGI bundle?

https://github.com/mvc-spec/mvc-spec/pull/4

Fragen?

Vielen Dank!

florian.hirsch@adorsys.de