



Introduction



Part 1 - The Basics



Part 2 - Core MVC 1.0 Features



Part 3 - Useful MVC 1.0 Features



Summary



Part 2 Core MVC 1.0 Features



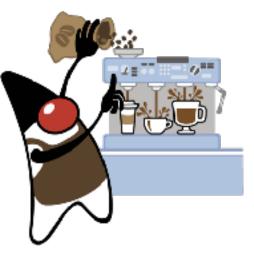
CDI Models

Mode



```
@Named("greeting")
@RequestScoped
public class Greeting {
   private String message;
   public void setMessage(String message) {
     this.message = message;
   public void getMessage() { return message; }
```

Model



```
@View("hello.jsp")
@Controller
@Path("hello")
public class HelloController {
   @GET
   public void hello() {
```

Model



```
@View("hello.jsp")
@Controller
@Path("hello")
public class HelloController {
   @Inject
   private Greeting greeting;
   @GET
   public void hello() {
     greeting.setMessage("Hello Cologne!");
```

Model



```
<%@page contentType="text/html"</pre>
pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
   <head>
     <title>MVC 1.0 Hello Demo</title>
   </head>
   <body>
      <h1>Hello ${greeting.message}</h1>
   </body>
```





```
@Controller
@Path("form")
public class FormController
                                 ConstraintValidationException
@POST
public Response formPost(@Valid @BeanParam FormDataBean f) {
                                                 @Min, @NotNull
                                                       etc.
      return Response.status(OK).entity("data.
```



```
public class FormViolationMapper implements
                  ExceptionMapper<ConstraintViolationException>
   public Response toResponse(ConstraintViolationException e) {
     Set<ConstraintViolation<?>>> s= e.getConstraintViolations();
     // process violations ...
     return Response.status(BAD_REQUEST)
                     .entity("error.jsp").build();
```

```
@Controller
                                           @MvcBinding allows MVC
public class FormController {
                                                error handling
  @Inject private BindingResult br;
  @POST
  public Response formPost(@Valid @BeanParam FormDataBean f) {
    if (br.isFailed()) {
        return Response.status(BAD_REQUEST)
                        .entity("error.jsp").build();
    return Response.status(OK).entity("data.jsp").build();
```

BindingResult	
---------------	--

All Methods Instance Methods	S Abstract Methods
Modifier and Type	Method and Description
Set <bindingerror></bindingerror>	getAllBindingErrors() Returns an immutable set of all binding errors detected while processing parameter bindings.
List <string></string>	getAllMessages() Returns an immutable list of all messages representing both binding errors and constraint violations.
Set <validationerror></validationerror>	getAllvalidationErrors() Returns an immutable set of all validation errors detected.
BindingError	getBindingError(String param) Returns the binding error for the binding specified by the given parameter name.
ValidationError	getValidationError(String param) Returns a single validation error detected for a parameter binding specified by the given parameter name.
Set <validationerror></validationerror>	getValidationErrors(String param) Returns an immutable set of all validation errors detected for a parameter binding specified by the given parameter name.
boolean	<pre>isFailed() Returns true if there is at least one binding error or constraint violation.</pre>



Known Issues



Disabling JAX-RS Validation on individual methods currently does not work in GlassFish.

Disabling it globally does...



Part 2

https://github.com/ivargrimstad/mvc-hol