MVC 1.0 - Hands-on LAB

Christian Kaltepoth
Senior Software Developer, ingenit
@chkal

Ivar Grimstad
Principal Consultant, Cybercom Sweden
@ivar_grimstad

Part 1
Introduction

⬛ No Change to Plan
🟧 Propose to Drop
🟦 Propose to Add

Revised Java EE 8 Proposal

CDI 2.0 (JSR 365)
- Bootstrap API for Java SE
- Async events
- Observer ordering

JSON-B 1.0 (JSR 367)
- JSON <-> object mapping

JMS 2.1 (JSR 368)
- Flexible JMS MDBs
- Improved XA support

Servlet 4.0 (JSR 369)
- HTTP/2 support

JAX-RS 2.1 (JSR 370)
- Reactive enhancements
- Server-sent events
- Non-blocking I/O
- Client-side circuit breakers

MVC 1.0 (JSR 371)
- Action-based MVC framework

JSF 2.3 (JSR 372)
- Small-scale new features
- Community-driven improvements

Management 2.0 (JSR 373)
- REST-based APIs

JSON-P 1.1 (JSR 374)
- JSON Pointer and Patch
- Java Lambda support

Security 1.0 (JSR 375)
- Authentication/authorization APIs
- OAuth, OpenID support
- Secret management

Bean Validation 2.0 (JSR 380)
- Collection constraints
- Date/Time support
- Community-requested features

Configuration
- Standard for externalizing application configuration

Health Checking
- Standard for client-side health reporting

JavaOne ORACLE

Rationale for Proposed Changes

New Functionality

- Cloud apps make many remote REST calls. Need a **client-side circuit breaker** added to JAX-RS
- Need a **secret vault** because there's no way to do this today using **standards**
- **Need OAuth and OpenID** support because those technologies have rapidly emerged as standards
- Need **externalized configuration store** to make applications retargetable across environments
- Need basic **multi-tenancy** support to accommodate needs of more complex apps and offer higher density
- **Need standard way of health checking** Java-based apps

Dropped Functionality

- **JMS** is no longer very relevant in cloud. Proposed to stay at JMS 2.0 standard (vs. upgrading to JMS 2.1).
- Cloud apps often ship headless, making **MVC** largely irrelevant
- Current **Management** JSR not widely used

JavaOne ORACLE

JSR #371
Model-View-Controller (MVC 1.0) Specification
Transfer Ballot

Ballot duration: 2017-01-17 to: 2017-01-30

Special Vote Instructions:

Santiago Pericasgeertsen, and Manfred Riem, Co-Sped. Leads, representing Oracle, request a JSR Transfer Ballot of JSR 371: Model-View-Controller (MVC 1.0) Specification, to Ivar Grimstad, an individual.

@Controller
to the Community !

History

August 2014	First Proposal
September 2014	Approved by JCP Executive Committee
September 2014	Expert Group starts discussions
March 2015	EDR 1
October 2015	EDR 2
October 2016	JavaOne
January 2017	Transfer Ballot

History

January 2017
Transfer
Ballot

April 2017
Transfer Complete

May 2017
Christian
co-spec lead

May 2017
Apache License 2.0

June 2017
Infrastructure Complete

October 2017
eclipse

Recent Activities

Transfer approved by the Executive Committee ✓

Thanks!

Finalize Transfer ✓

Done!

New Infrastructure Setup ✓

Done!

Licensing ✓

Done!

Ongoing Activities

Formally move infrastructure from java.net √
Done!

Bring in Christian Kaltepoth as co spec-lead √
Done!

Revise the Schedule √

Adopt-a-JSR

Apache License 2.0

Adopt-a-JSR

Write Code!

Give us Feedback

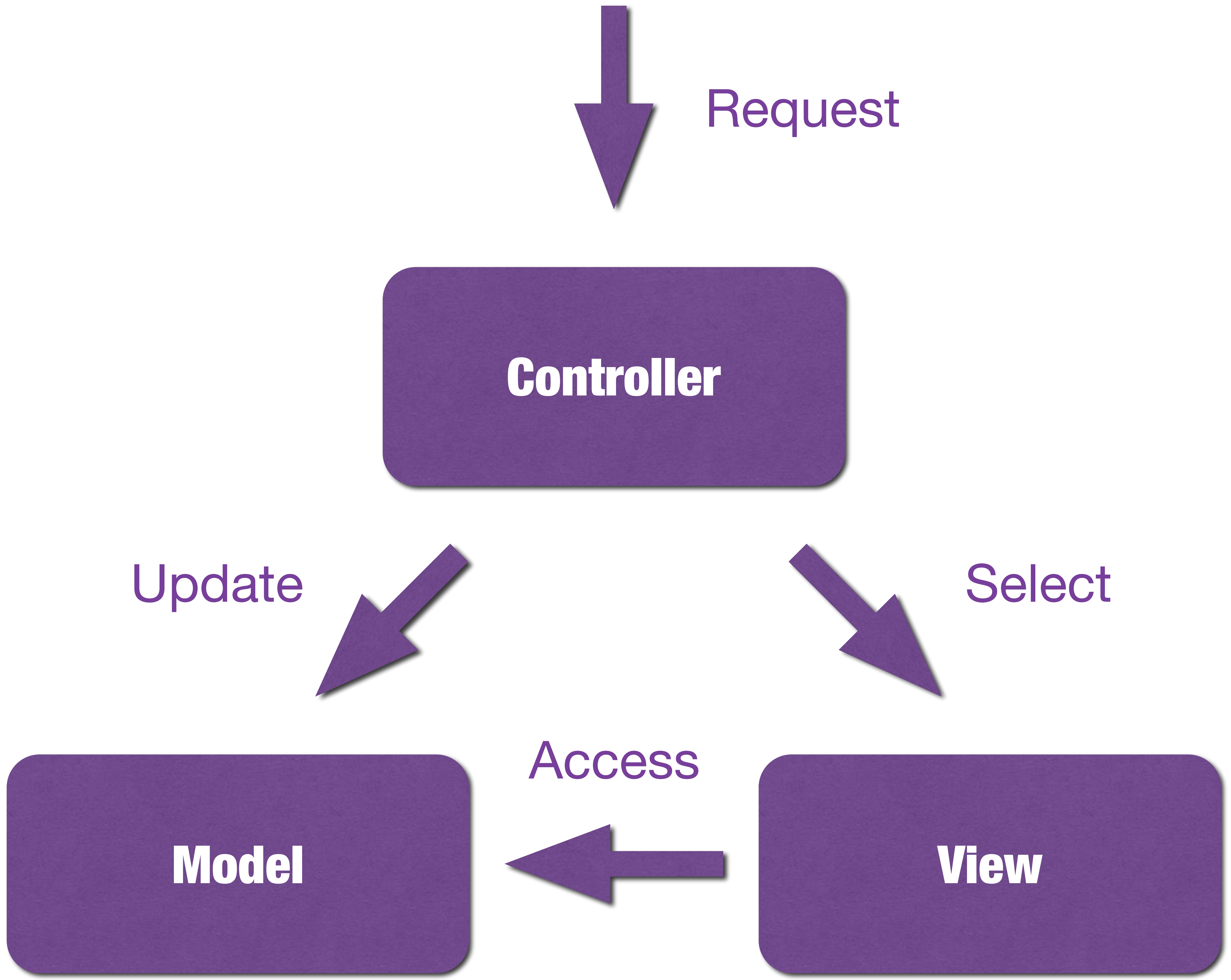
Blog

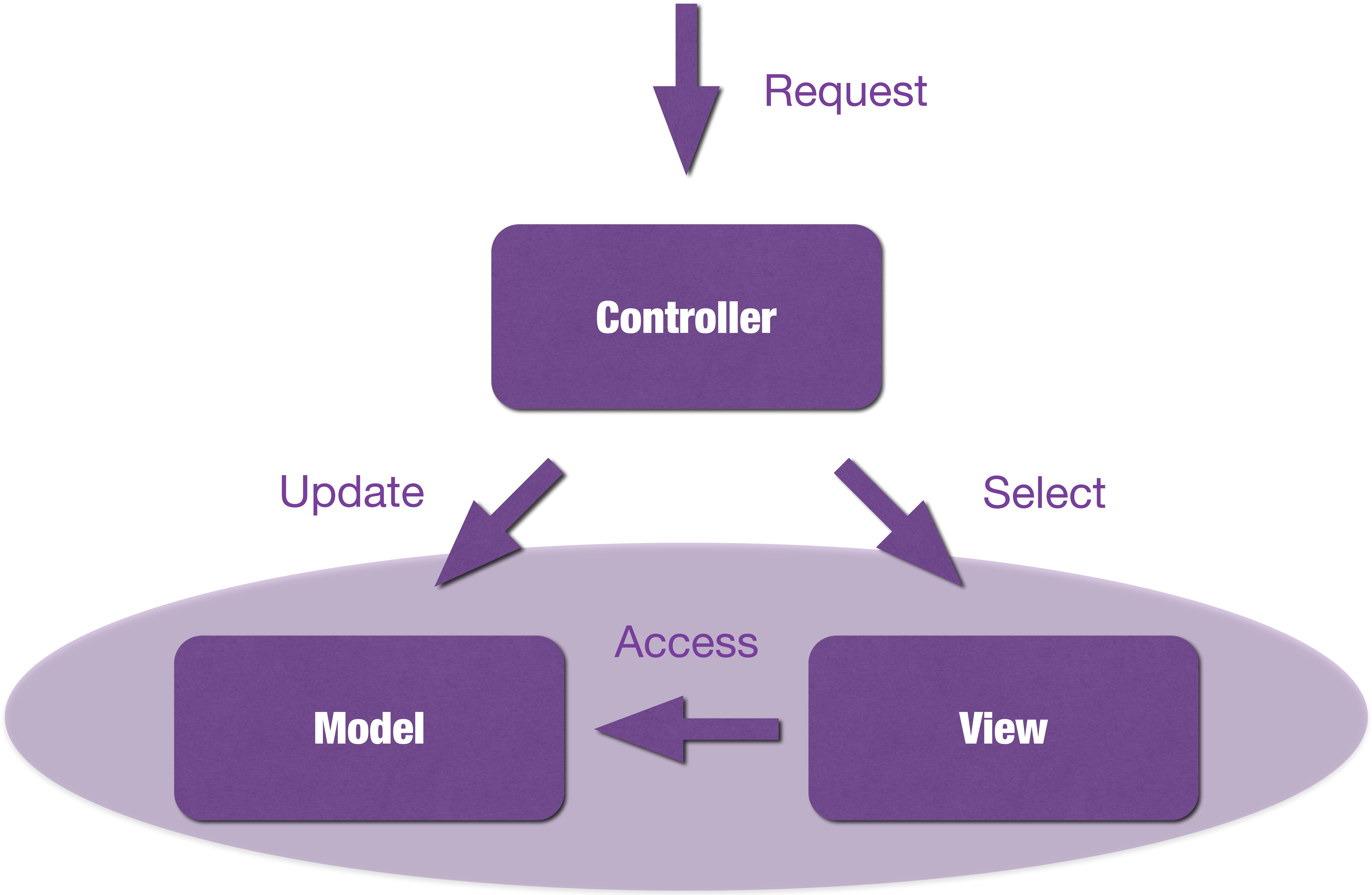
Tweet

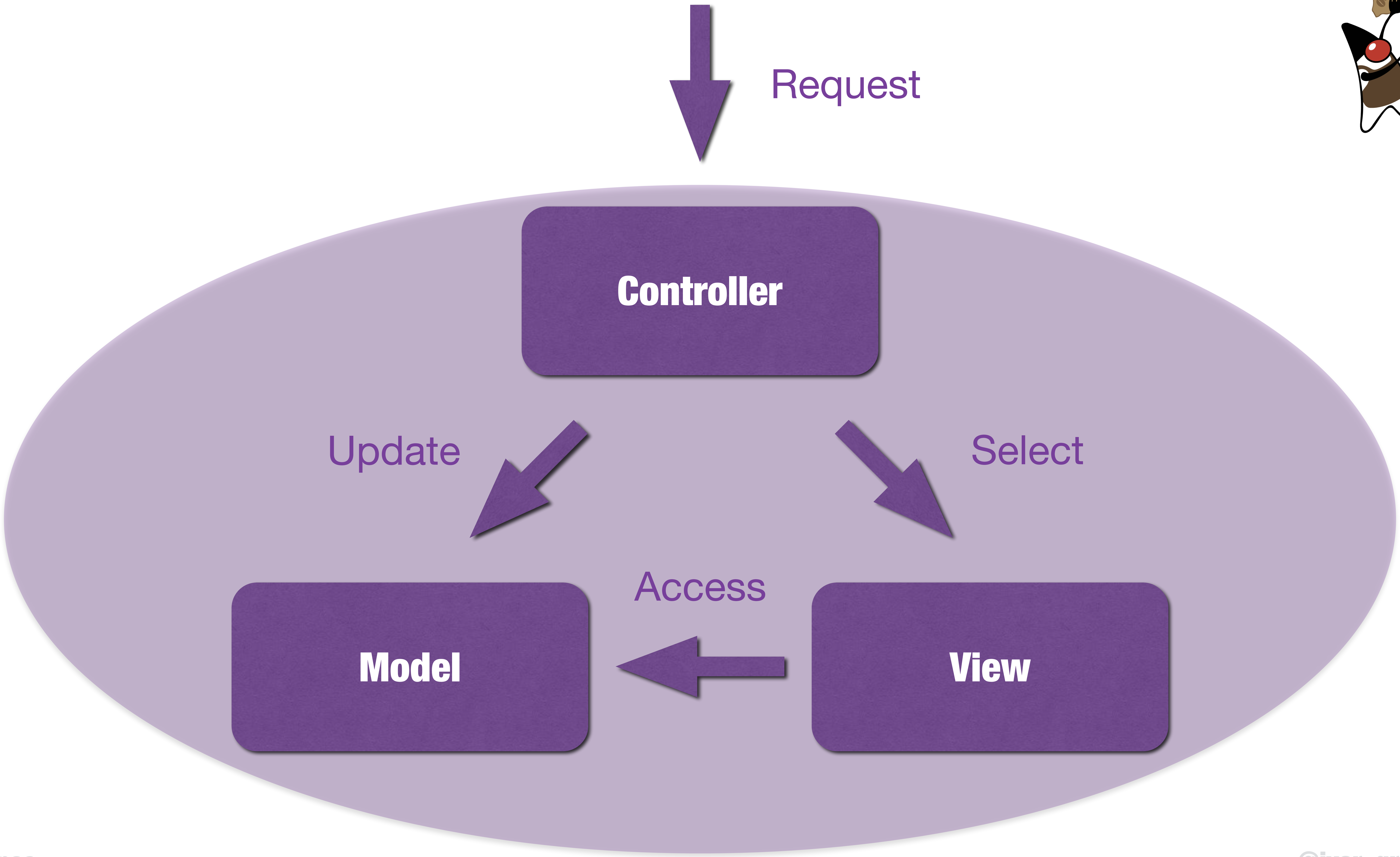
Create a Logotype √

MVC 1.0 - The Basics

Action-based MVC


Request
Controller
Update
Select
Access
Model
View


Request
Controller
Update
Select
Access
Model
View


Request
Controller
Update
Select
Access
Model
View

Existing Java EE Technologies

Key Decisions

Key Decision

Build MVC 1.0 on top of JAX-RS

Controllers

Controller

```
public class HelloController {




}
```

Controller

```
@Path("hello")
public class HelloController {






}
```

Controller

```
@Controller
@Path(“hello”)
public class HelloController {









}
```

Views

View

```
@Controller
@Path("hello")
public class HelloController {






}
```

View

```
@Controller
@Path("hello")
public class HelloController {




   @GET
   public String hello() {
      return "hello.jsp";
   }
}
```

View

```
@Controller
@Path(“hello”)
public class HelloController {




   @GET
   public Response hello() {
      return Response.status(OK).entity(“hello.jsp”).build();
   }
}
```

View

```
@Controller
@Path("hello")
public class HelloController {



   @View("hello.jsp")
   @GET
   public void hello() {

   }
}
```

View

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





   @GET
   public void hello() {

   }
}
```

Models

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 Models model;

   @GET
   public void hello() {
     model.put("message", "Hello Cologne!");
   }
}
```

Model

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

Part 1

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

CYBERCOM GROUP

cybercom.com

@mvc_spec

@ivar_grimstad