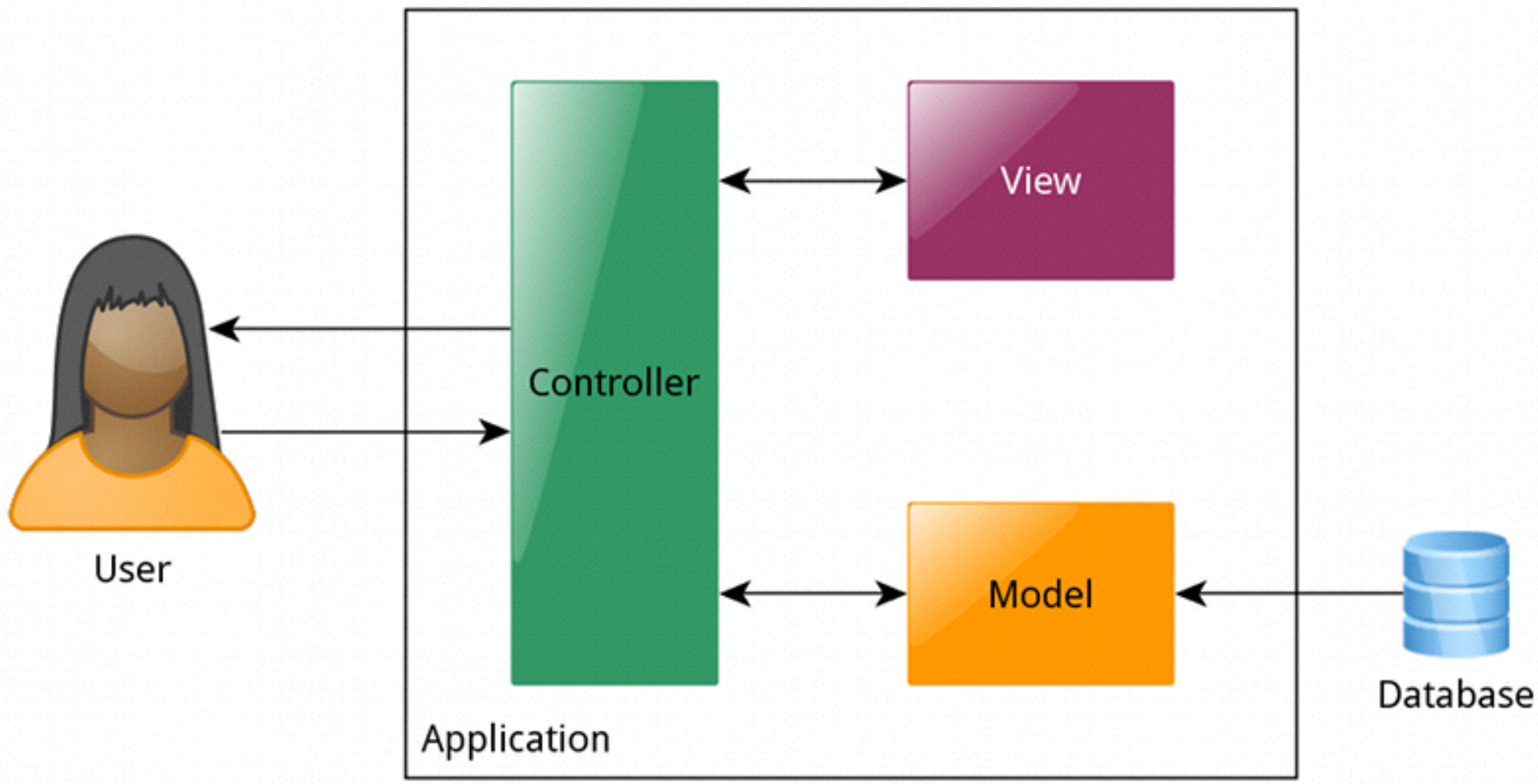
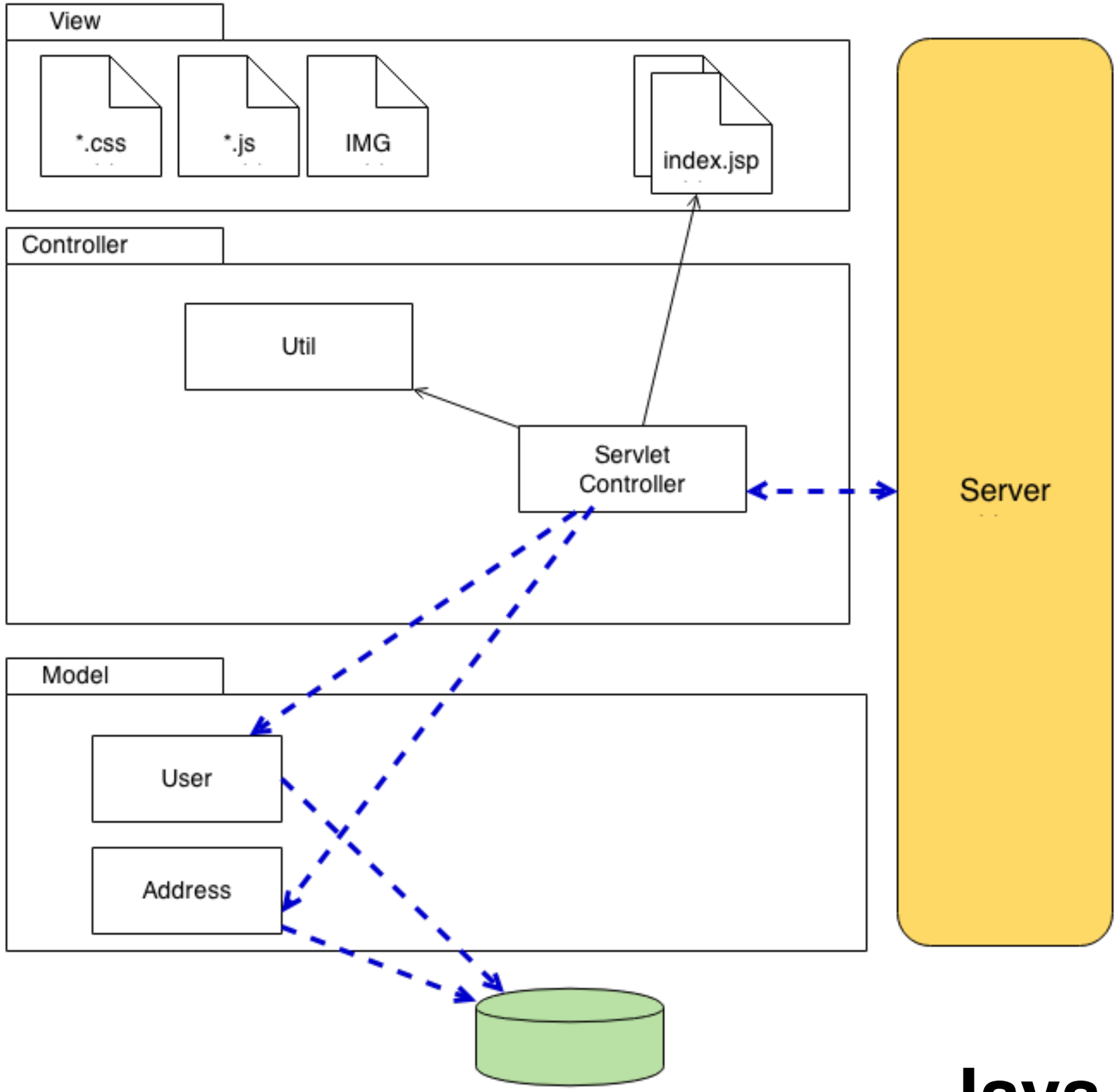
# Spring MVC
## brief tutorial

Andrea Caracciolo
http://scg.unibe.ch/staff/caracciolo

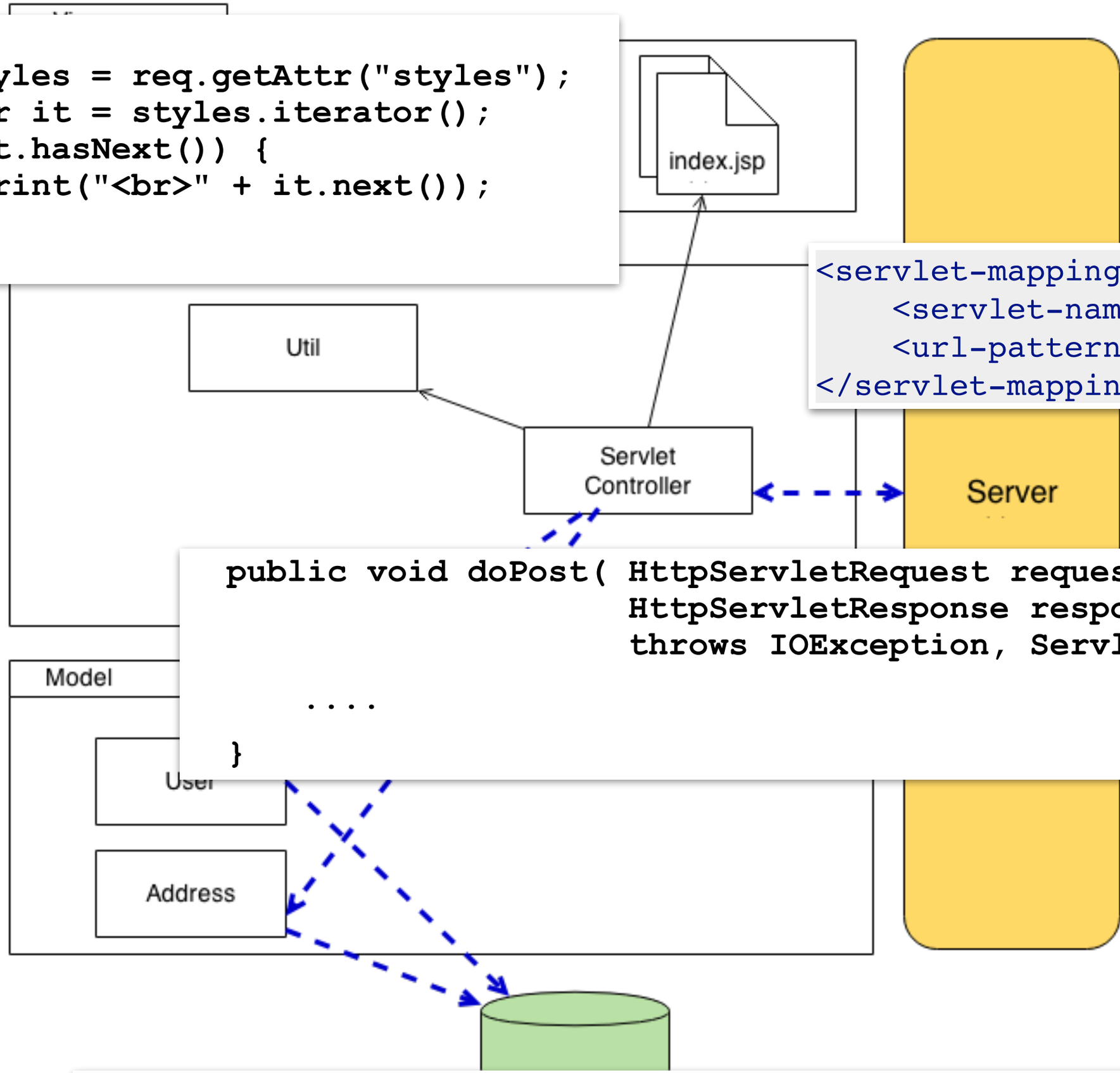
# MVC






# Java Servlet

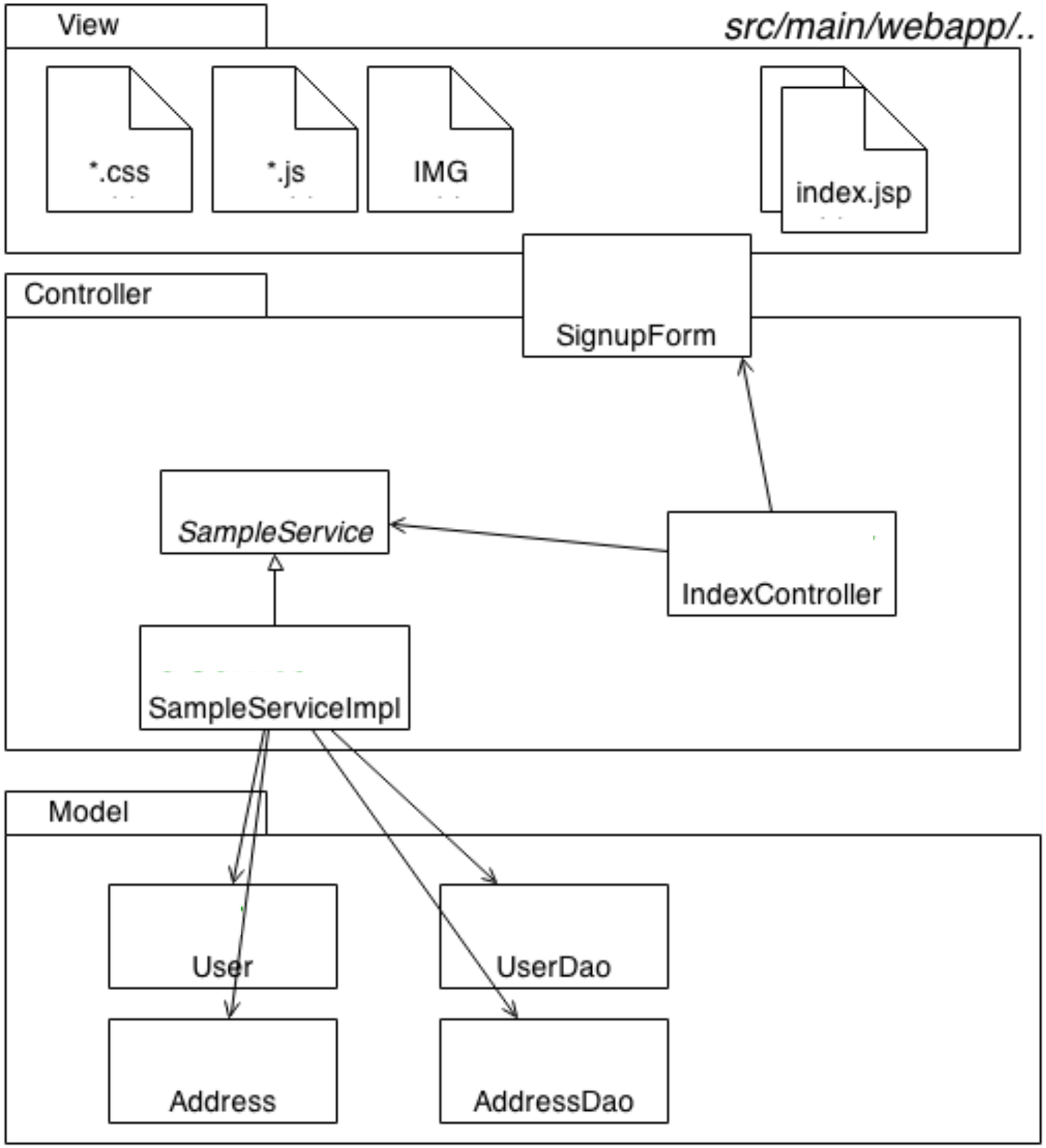
```
<%
  List styles = req.getAttr("styles");
  Iterator it = styles.iterator();
  while(it.hasNext()) {
    out.print("<br>" + it.next());
  }
%>
```



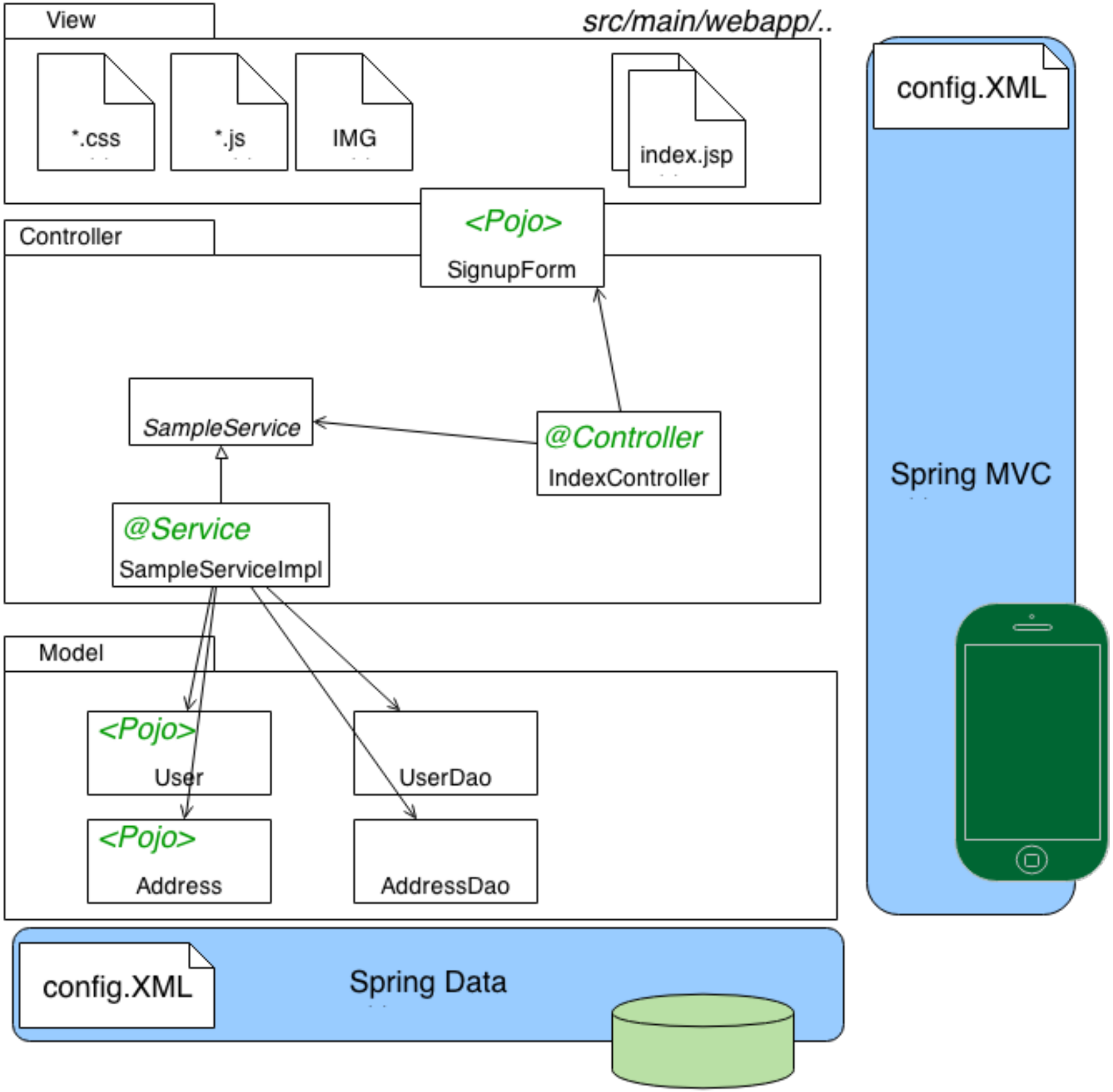

```
<servlet-mapping>
    <servlet-name>MyPage</..>
    <url-pattern>/MyPage</..>
</servlet-mapping>
```

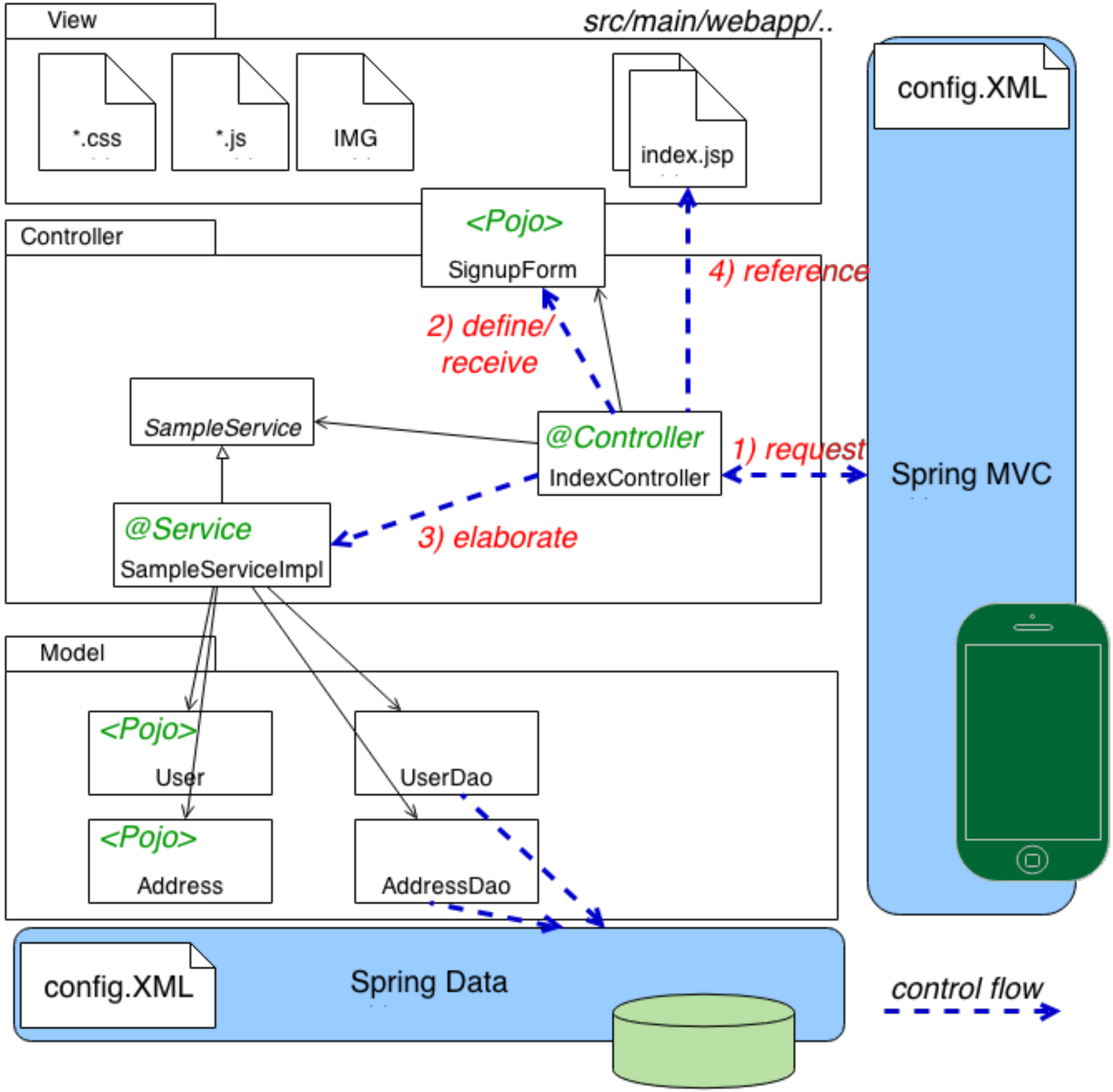
```
public void doPost( HttpServletRequest request,
                    HttpServletResponse response)
                    throws IOException, ServletException {
    ....
}
```

```
insert into myTable(x,y,z) values('foo', 'bar', 'baz')
```


View
src/main/webapp/..
*.css
*.js
IMG
index.jsp
Controller
SignupForm
SampleService
IndexController
SampleServiceImpl
Model
User
UserDao
Address
AddressDao

# Spring/J2EE


View
src/main/webapp/..
*.css
*.js
IMG
index.jsp
config.XML
Controller
<Pojo>
SignupForm
SampleService
@Controller
IndexController
Spring MVC
@Service
SampleServiceImpl
Model
<Pojo>
User
UserDao
<Pojo>
Address
AddressDao
config.XML
Spring Data


View
src/main/webapp/..
*.css
*.js
IMG
index.jsp
config.XML
Controller
<Pojo>
SignupForm
4) reference
2) define/
receive
SampleService
@Controller
IndexController
1) request
Spring MVC
@Service
SampleServiceImpl
3) elaborate
Model
<Pojo>
User
UserDao
<Pojo>
Address
AddressDao
config.XML
Spring Data
control flow

# Spring

- **loose-coupling (DI)**
  - example: *@Controller, @Autowired*
  - dependencies defined in config / annotation
  - pros: testability, readability, maintainability

- **declarative programming (AOP)**
  - example: *@Transactional*
  - what you want to do vs. how to do it
  - avoid boilerplate code

*http://docs.spring.io/spring-framework/docs/current/spring-framework-reference/html/*

# ESE Skeleton project

## - XML

- **pom.xml**: required libraries + build (maven)
- **web.xml**: java webapp deployment descriptor
  - **springMVC.xml**
  - **springData.xml**
  - **springSecurity.xml**

## - MVC

- **VIEW:** webapp/pages/**.jsp
- **CONTROLLER:** java/org/sample/controller/**.java
- **MODEL:** java/org/sample/model/**.java

*https://github.com/ese-unibe-ch/ese2014-wiki/tree/master/Skeleton*

# Request

```
@Controller
public class IndexController {

    @Autowired
    SampleService sampleService;

    @RequestMapping(value = "/", method = RequestMethod.GET)
    public ModelAndView index() {
        ModelAndView model = new ModelAndView("index");
        model.addObject("signupForm", new SignupForm());
        return model;
    }
}
```

map to request

define required parameters

handle request

return an answer model

**java/org/sample/controller/IndexController.java**

# Request

```
@Controller
public class IndexController {

    @Autowired
    SampleService sampleService;


    @RequestMapping(value = "/", method = RequestMethod.GET)
    public ModelAndView index() {
      ModelAndView model = new ModelAndView("index");
      model.addObject("signupForm", new SignupForm());
        return model;
    }
}
```

## URI Template Patterns

```
@RequestMapping(value="/owners/{ownerId}", method=RequestMethod.GET)
public String findOwner(@PathVariable String ownerId, Model model) {
```

```
@RequestMapping("/spring-web/{symbolicName:[a-z-]}-{version:\\d\\.\\d\\.\\d}{extension:\\.[a-z]
public void handle(@PathVariable String version, @PathVariable String extension) {
```

## Consumable Media Types

```
@RequestMapping(value = "/pets", method = RequestMethod.POST, consumes="application/json")
public void addPet(@RequestBody Pet pet, Model model) {
```

http://docs.spring.io/spring/docs/current/spring-framework-reference/html/mvc.html#mvc-ann-requestmapping

http://docs.spring.io/spring/docs/current/spring-framework-reference/html/mvc.html#mvc-ann-requestmapping-uri-templates

http://docs.spring.io/spring/docs/current/spring-framework-reference/html/mvc.html#mvc-ann-requestmapping-consumes

# Request

```
@Controller
public class IndexController {

    @Autowired
    SampleService sampleService;


    @RequestMapping(value = "/", method = RequestMethod.GET)
    public ModelAndView index() {
      ModelAndView model = new ModelAndView("index");
      model.addObject("signupForm", new SignupForm());
        return model;
    }
}
```

*"An `@RequestMapping` handler method can have a very flexible signatures"*

**Method signature**

*xx.jsp?userId=10*

```
public ModelAndView setupForm(@RequestParam("userId") int petId, ModelMap model)
public ModelAndView handle(@RequestBody String body, Writer writer) throws IOException
public ModelAndView displayHeaderInfo(@CookieValue("JSESSIONID") String cookie)
public ModelAndView displayHeaderInfo(@RequestHeader("Accept-Encoding") String encoding,
                                      @RequestHeader("Keep-Alive") long keepAlive)
public ModelAndView processSubmit(@Valid Pet pet, BindingResult result)
```

http://docs.spring.io/spring/docs/current/spring-framework-reference/html/mvc.html#mvc-ann-methods

# Handling

pojo validation result

```
    @RequestMapping(value = "/create", method = RequestMethod.POST)
    public ModelAndView create(@Valid SignupForm signupForm, BindingResult
result, RedirectAttributes redirectAttributes) {
      ModelAndView model;
      if (!result.hasErrors()) {
            try {
             sampleService.saveFrom(signupForm);
             model = new ModelAndView("show");
            } catch (InvalidUserException e) {
             model = new ModelAndView("index");
             model.addObject("page_error", e.getMessage());
            }
      } else {
        model = new ModelAndView("index");
      }
      return model;
}
```

use services to interact with the model

initialize a model

bind user model attributes

return model

**java/org/sample/controller/IndexController.java**

# View

```
model.addObject("signupForm", new SignupForm());
```

user model attribute (logical name)

target page

```
<form:form method="post" modelAttribute="signupForm" action="create" id="signupForm"
cssClass="form-horizontal" autocomplete="off">
    <fieldset>
        <legend>Enter Your Information</legend>

        <c:set var="emailErrors"><form:errors path="email"/></c:set>
        <div class="control-group<c:if test="${not empty emailErrors}"> error</c:if>">
            <label class="control-label" for="field-email">Email</label>

        <div class="controls">
            <form:input path="email" id="field-email" tabindex="1" maxlength="45"
                    placeholder="Email"/>
            <form:errors path="email" cssClass="help-inline" element="span"/>
        </div>
    </div>
```

form error

form input field

**webapp/pages/index.jsp**

# Model attribute - Pojo

```
public class SignupForm {
    private Long id;

    @NotNull
    @Pattern(regexp = "[a-z0-9!#$%&'*+/=?^_`{|}~-]+(?:\\.[a-z0-9!#$%&'*+/=?^_`{|}~-]+)*@(?:[a-z0-9](?:[a-z0-9-]*[a-z0-9])?\\.)+[a-z0-9](?:[a-z0-9-]*[a-z0-9])?", message = "Must be valid email address")
    private String email;

    ..

    public String getEmail() {
        return email;
    }

    public void setEmail(String email) {
        this.email = email;
    }

}
```

validation rules **

pojo field

getter/setter

** http://docs.jboss.org/hibernate/validator/4.0.1/reference/en/html/validator-usingvalidator.html#table-builtin-constraints

**java/org/sample/controller/pojos/SignupForm.java**

# Spring Security

- Use it for authentication and authorization

**http://projects.spring.io/spring-security/**

**http://krams915.blogspot.ch/2010/12/spring-security-3-mvc-using-simple-user.html**

**http://docs.spring.io/spring-security/site/docs/3.2.5.RELEASE/reference/htmlsingle/**

# Documentation



**http://docs.spring.io/spring/docs/current/spring-framework-reference/html/mvc.html**