

05. Продвинутые методики построения web-приложения

Лекции по информатике
для студентов второго курса Высшей школы ИТИС КФУ
2020

Ференец Александр Андреевич

старший преподаватель кафедры программной инженерии

С использованием материалов
к. т. н., доцента кафедры программной инженерии Абрамского М.М.

aferenets@it.kfu.ru

Design Pattern




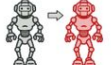

Зачем?

1. Общий язык, терминология
2. Вместе с описанием *Best Practices*

Шаблоны проектирования – это

1. Не защита от велосипедов
2. Не вся терминология в программировании
3. Не волшебный способ написать хороший код!
4. Могут иметь разные реализации

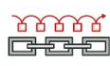



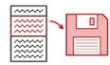

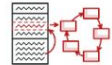
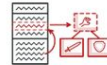


Порождающие

	
Factory Method	Abstract Factory
	
Builder	Prototype
	
Singleton	

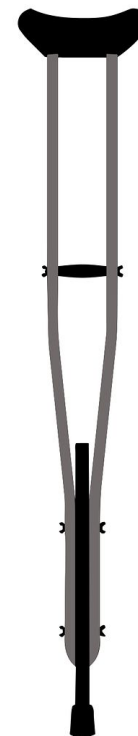
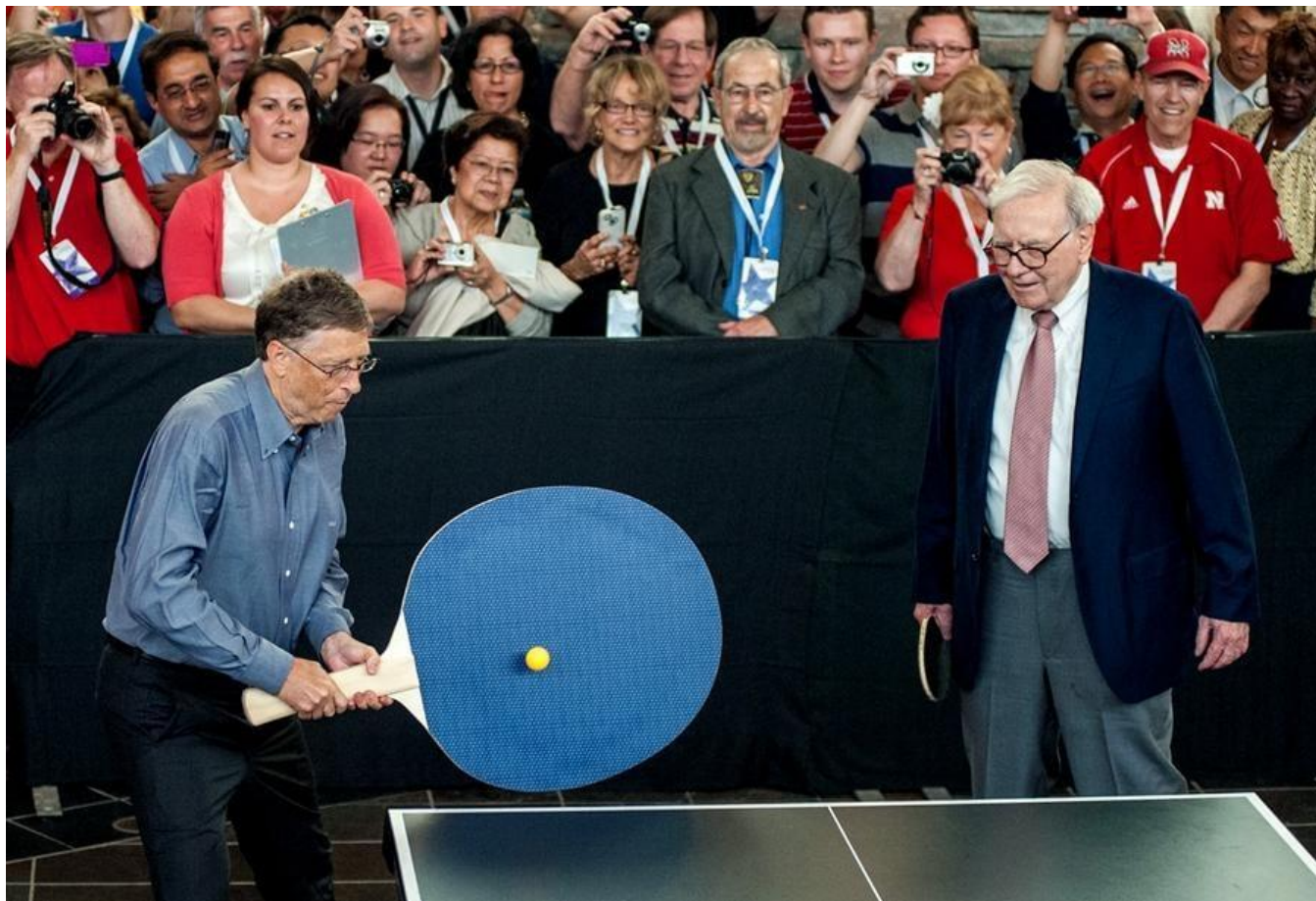
Структурные

	
Adapter	Bridge
	
Composite	Decorator
	
Facade	Flyweight

Поведенческие

			
Chain of Responsibility	Command	Iterator	Mediator
			
Memento	Observer	State	Strategy
			
Template Method	Visitor		

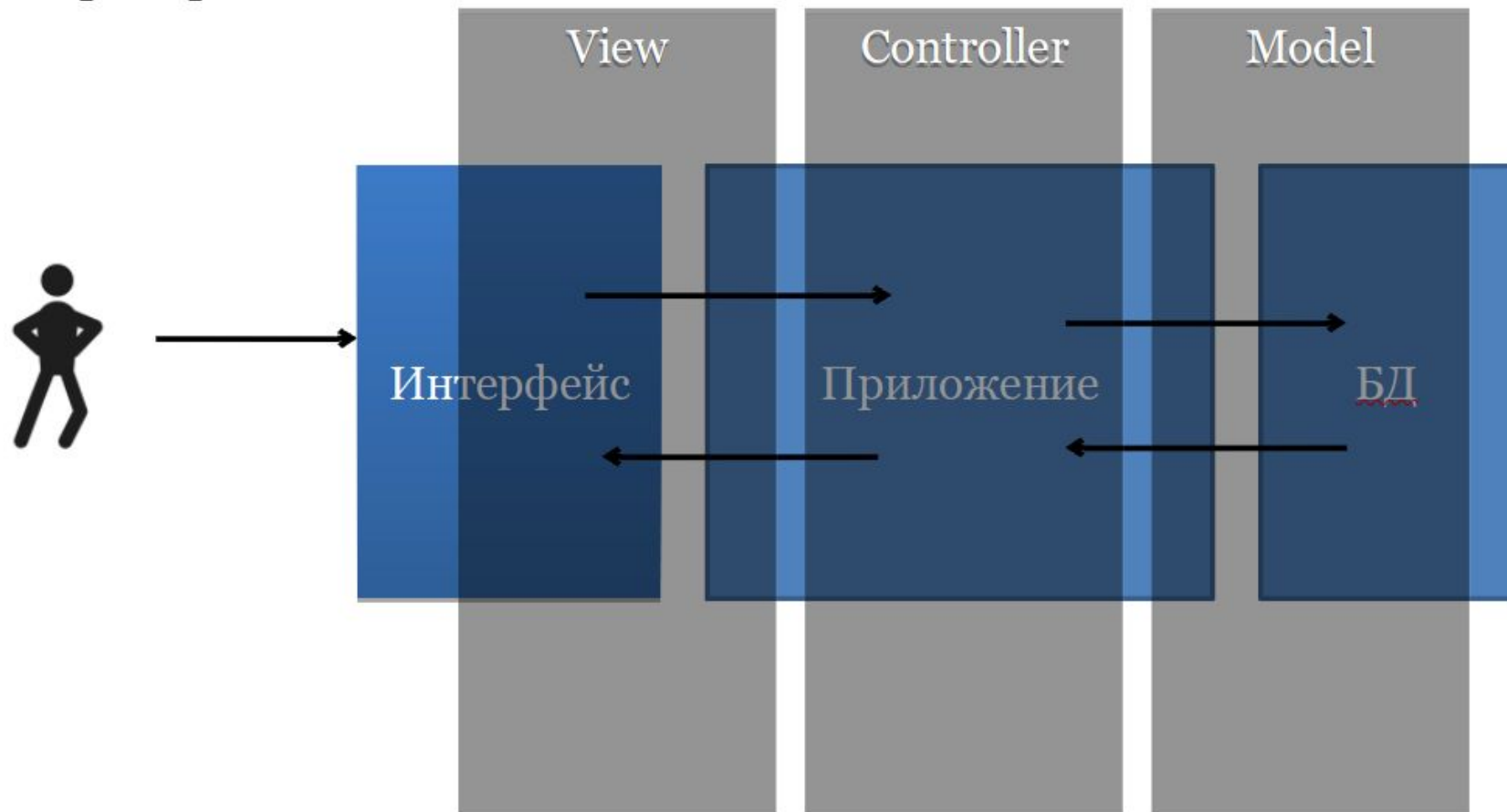
АРХИТЕКТУРА. Антипаттерны

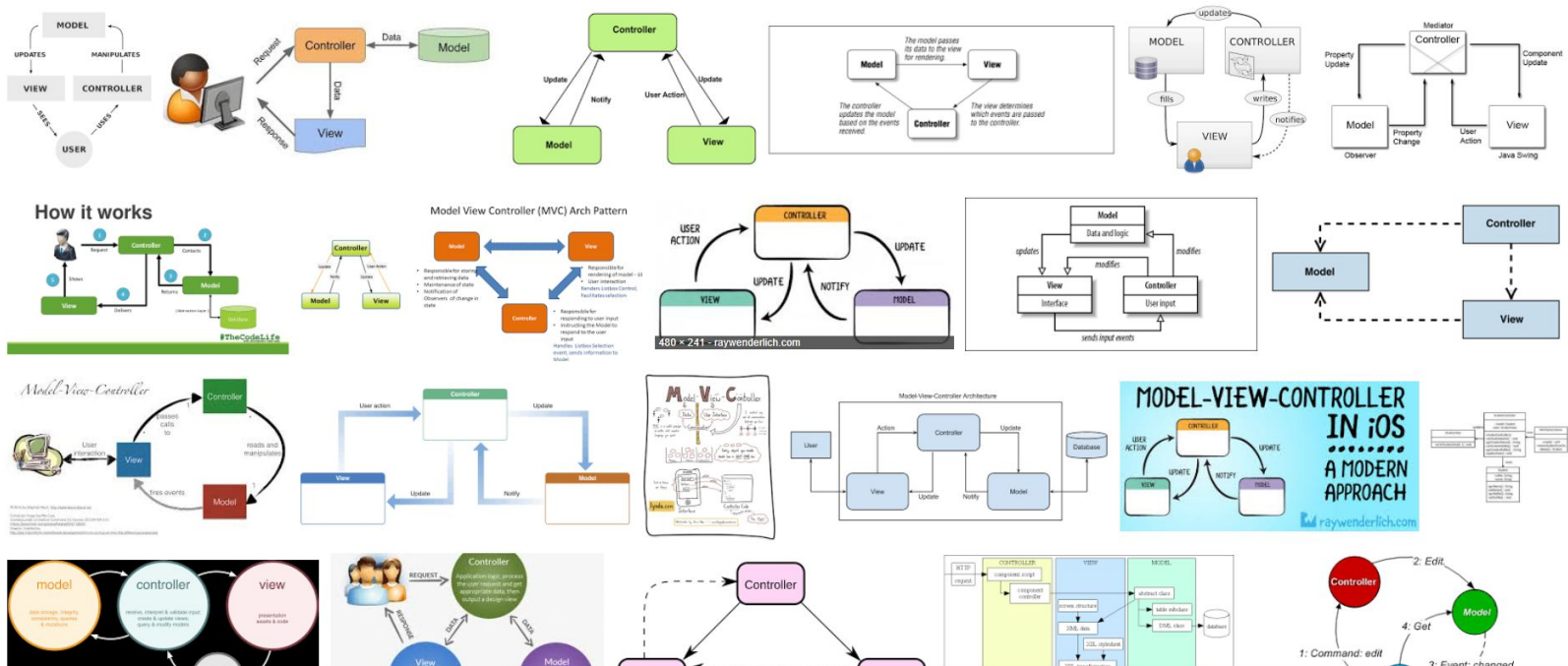


1. Data Mapper, Active Record
2. Listener
3. ...

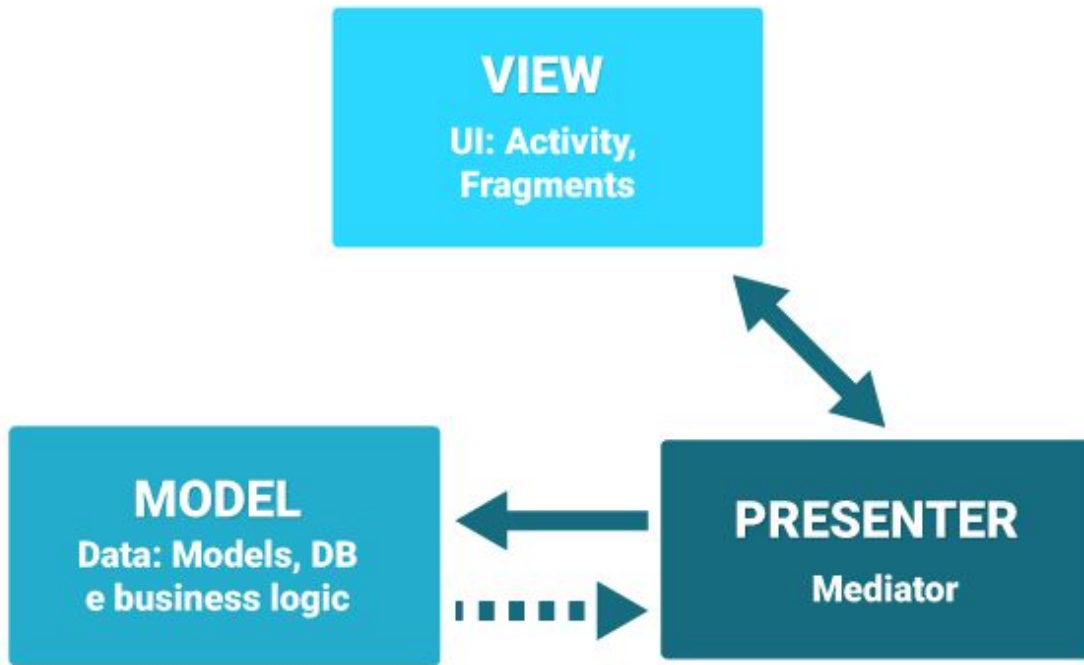
1. Контроллеры (сервлеты)
2. Сущности
3. Виды
4. DAO / Репозитории
5. Хелперы
6. Расширения для подключаемых библиотек (например, адаптеры)
7. Слушатели
8. Сервисы бизнес-логики

АРХИТЕКТУРА. Холивар MVC

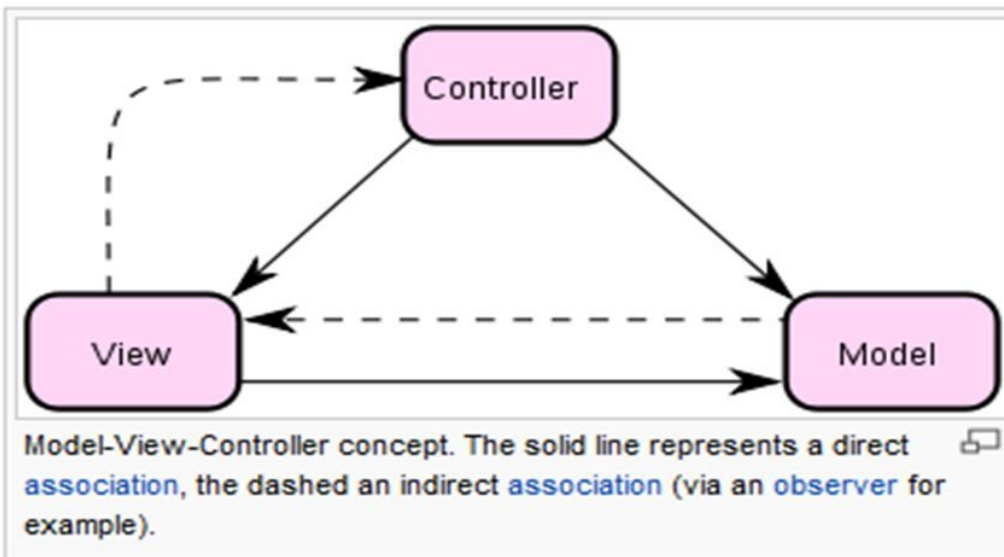




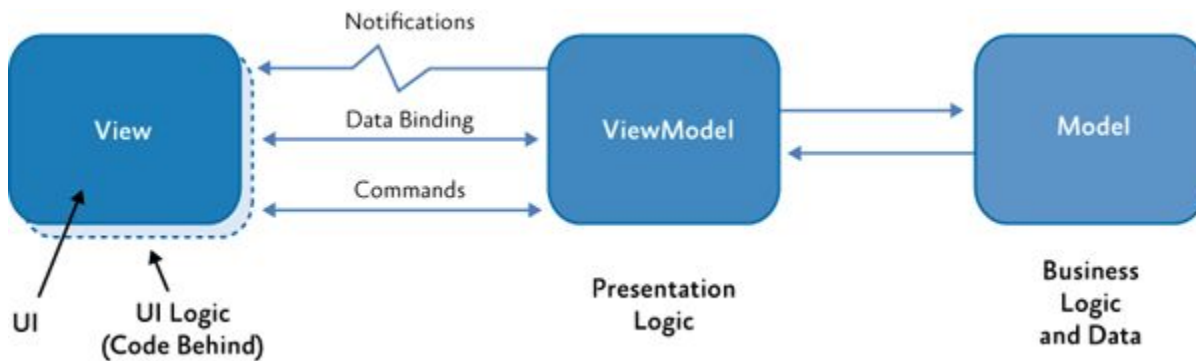
Model View Presenter



Reenskaug, Trygve. "MVC XEROX PARC 1978-79"

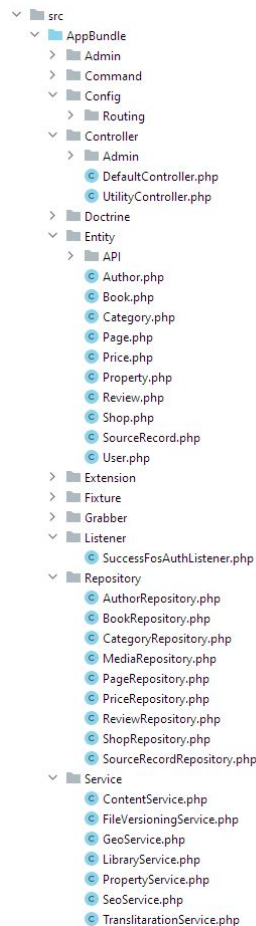
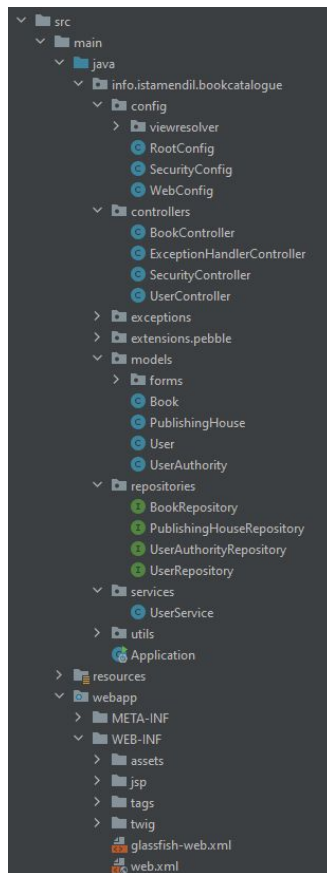
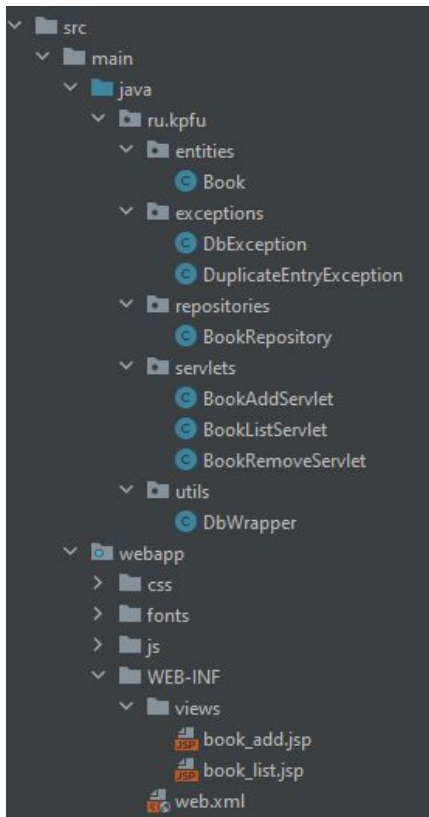


АРХИТЕКТУРА. Холивар MVC



1. Контроллеры (сервлеты)
2. Сущности
3. Виды
4. DAO / Репозитории
5. Хелперы
6. Расширения для подключаемых библиотек (например, адаптеры)
7. Слушатели
8. Сервисы бизнес-логики

АРХИТЕКТУРА. Организация кода веб-приложения



Уровень абстракции — один из способов сокрытия деталей реализации определенного набора функциональных возможностей. Применяется для управления сложностью проектируемой системы при декомпозиции, когда система представляется в виде иерархии уровней абстракции. (Wikipedia)

Если при проектировании возникает проблема сложности описания какой-то части системы, то можно ввести дополнительный слой абстракции. Но есть одна проблема, которую не решить дополнительным слоем абстракции...

Controller

Бизнес-логика

Доступ к данным

Помимо банальных CRUD-действий на сайте существуют сложные логики. Например, при добавлении поста на Image Board (Pikabu, Reddit):

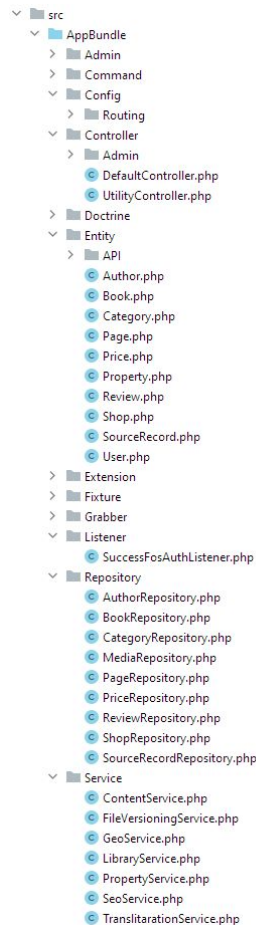
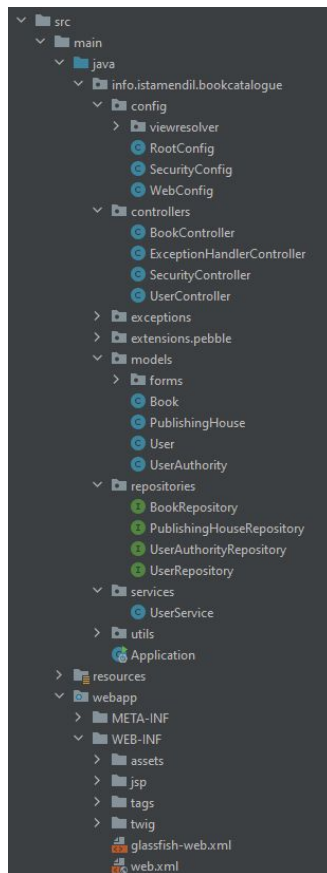
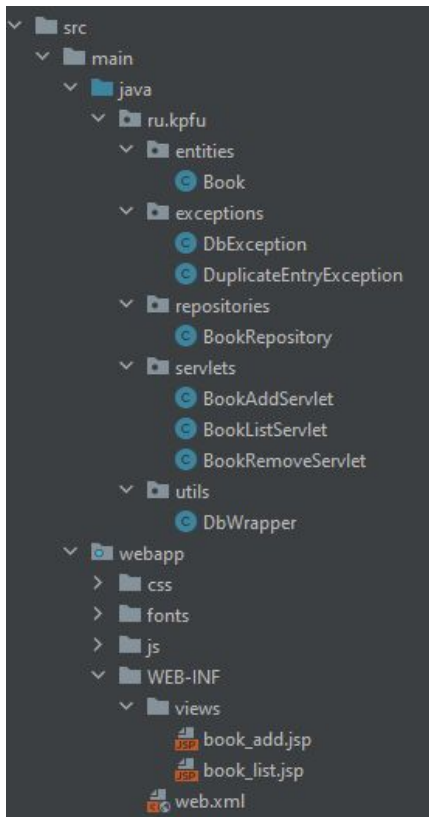
- ★ Добавить пост в личные ленты подписок.
- ★ Изменить рейтинг.
- ★ Уменьшить счётчик разрешённых постов за день.
- ★ Отправить на премодерацию?

Бизнес-логика

С точки зрения шаблонов проектирования,
сервисы бизнес-логики – это Модель

1. Контроллеры (сервлеты)
2. Сущности
3. Виды
4. DAO / Репозитории
5. Хелперы
6. Расширения для подключаемых библиотек (например, адаптеры)
7. Слушатели
8. **Сервисы бизнес-логики!!!**

АРХИТЕКТУРА. Слои абстракции и все-все-все



КОНТРОЛЛЕР. Типичный запрос на изменения в БД

```
@Override
protected void doPost(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {
    String email = req.getParameter("email");
    String password = req.getParameter("password");

    if(email != null && password != null){
        if(UserService.register(req, email, password)){
            resp.sendRedirect(getServletContext().getContextPath() + "/profile");
            return;
        }
    }
    req.setAttribute("email", req.getParameter("email"));
    getServletContext().getRequestDispatcher("/WEB-INF/views/profile.jsp").forward(req, resp);
}
```

```
if (result.hasErrors()) {
    return showForm(map);
} else {
    bookService.save(book);
    session.addFlashAttribute("message", "Book \"" + book.getName() + "\" has been saved successfully");
    session.addFlashAttribute("messageType", "success");
    return "redirect:" + generateUri("/book/add");
}
```

```
@Override
protected void doPost(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {
    String email = req.getParameter( s: "email");
    String password = req.getParameter( s: "password");

    if(email != null && password != null){
        if(UserService.register(req, email, password)){
            resp.sendRedirect( s: getServletContext().getContextPath() + "/profile");
            return;
        }
    }
    req.setAttribute( s: "email", req.getParameter( s: "email"));
    getServletContext().getRequestDispatcher( s: "/WEB-INF/views/profile.jsp").forward(req, resp);
}
```

Защита от повторной отправки формы

Подтвердите повторную отправку формы

На странице, которую вы ищете, использовалась введенная вами информация. При возврате на эту страницу может потребоваться повторить выполненные ранее действия. Продолжить?

Продолжить

Отмена

```
if ($form->isSubmitted() && $form->isValid()) {
    // do some sort of processing

    $this->addFlash(
        'notice',
        'Your changes were saved!'
    );
    // $this->addFlash() is equivalent to $request->getSession()->getFlashBag()->add()

    return $this->redirectToRoute(...);
}

return $this->render(...);
```

Сериализация – представление объекта в виде последовательности байт.
Строка – последовательность байт ;)

ПРЕДСТАВЛЕНИЕ ДАННЫХ. Оценка

1. Избыточность
2. Тип структуры (размерность?)
3. Читаемость
4. Легко сломать?
5. Типы данных

eXtensible Markup Language

```
<bookstore>
  <book category="cooking">
    <title lang="en">Everyday Italian</title>
    <author>Giada De Laurentiis</author>
    <year>2005</year>
    <price>30.00</price>
  </book>
  <book category="children">
    <title lang="en">Harry Potter</title>
    <author>J K. Rowling</author>
    <year>2005</year>
    <price>29.99</price>
  </book>
  <book category="web">
    <title lang="en">XQuery Kick Start</title>
    <author>James McGovern</author>
    <author>Per Bothner</author>
    <author>Kurt Cagle</author>
    <author>James Linn</author>
    <author>Vaidyanathan Nagarajan</author>
    <year>2003</year>
    <price>49.99</price>
  </book>
  <book category="web" cover="paperback">
    <title lang="en">Learning XML</title>
    <author>Erik T. Ray</author>
    <year>2003</year>
    <price>39.95</price>
  </book>
</bookstore>
```

1. Избыточность
2. Тип структуры (размерность?)
3. Читаемость
4. Легко сломать?
5. Типы данных

eXtensible Markup Language

1. Заглавная строка
`<?xml version="1.0" encoding="UTF-8"?>`
2. Вложенность
3. Есть ровно один корневой элемент
4. Все элементы состоят из открывающего и закрывающего тега + содержимое
Либо: `<element />`
5. У атрибутов обязательно указывается значение в кавычках (может быть пустой строкой)
6. UTF-8

1. Избыточность
2. Тип структуры (размерность?)
3. Читаемость
4. Легко сломать?
5. Типы данных

Comma Separated Values

```
Ivan, 1995, Kazan  
Peter, 1998, Ulyanovsk  
Kate, 1996, Moscow
```

1. Избыточность
2. Тип структуры (размерность?)
3. Читаемость
4. Легко сломать?
5. Типы данных

Comma Separated Values

1. Разделитель значений – запятая
2. Разделитель записей – символ переноса строки
3. Кавычки
4. Могут использоваться другие символы

1. Избыточность
2. Тип структуры (размерность?)
3. Читаемость
4. Легко сломать?
5. Типы данных

INItialization

```
[Main]
class=ru.kpfu.itis.App
args="42 abc 300"
```

```
[Database]
user=admin
password=qwerty
name=test
```

1. Избыточность
2. Тип структуры (размерность?)
3. Читаемость
4. Легко сломать?
5. Типы данных

INItialization

1. Пары ключ-значение
2. Пары могут делиться на секции
3. Могут использоваться кавычки

1. Избыточность
2. Тип структуры (размерность?)
3. Читаемость
4. Легко сломать?
5. Типы данных

JavaScript Object Notation

```
[
  {
    "author": "Hans Christian Andersen",
    "country": "Denmark",
    "imageLink": "images/fairy-tales.jpg",
    "language": "Danish",
    "link": "https://en.wikipedia.org/wiki/Fairy_Tales_Told_for_Children._First_Collection.\n",
    "pages": 784,
    "title": "Fairy tales",
    "year": 1836
  },
  {
    "author": "Dante Alighieri",
    "country": "Italy",
    "imageLink": "images/the-divine-comedy.jpg",
    "language": "Italian",
    "link": "https://en.wikipedia.org/wiki/Divine_Comedy\n",
    "pages": 928,
    "title": "The Divine Comedy",
    "year": 1315
  }
]
```

1. Избыточность
2. Тип структуры (размерность?)
3. Читаемость
4. Легко сломать?
5. Типы данных

JavaScript Object Notation

1. Синтаксис JS
2. Типы значений согласно типам JS

1. Избыточность
2. Тип структуры (размерность?)
3. Читаемость
4. Легко сломать?
5. Типы данных

YAML ain't Markup Language

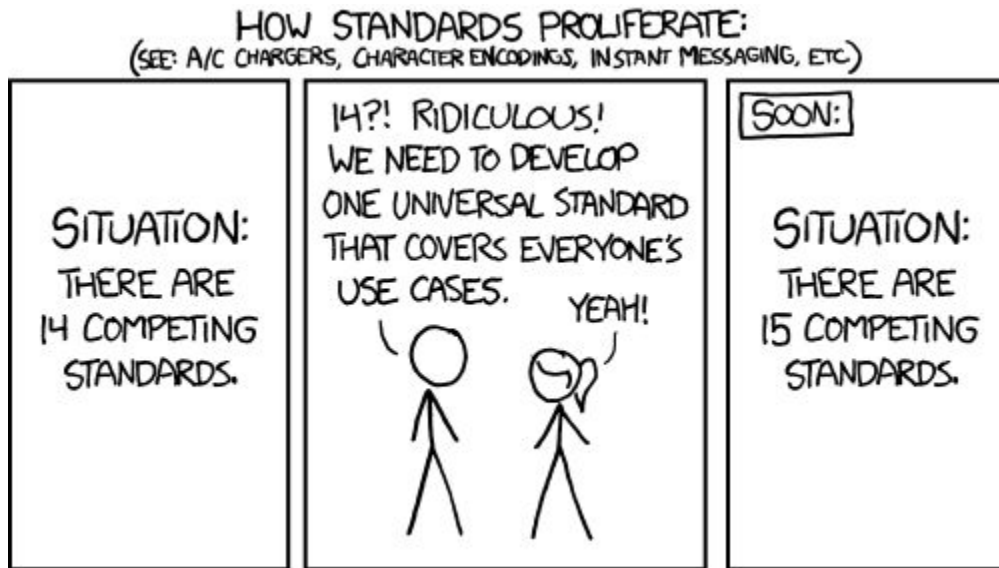
```
14 framework:
15   #esi: ~
16   translator: { fallbacks: ['%locale%'] }
17   secret: '%secret%'
18   router:
19     resource: ''
20     strict_requirements: ~
21   form: ~
22   csrf_protection: ~
23   validation: { enable_annotations: true }
24   #serializer: { enable_annotations: true }
25   default_locale: '%locale%'
26   trusted_hosts: ~
27   session:
28     handler_id: session.handler.native_file
29     save_path: '%kernel.project_dir%/var/sessions/%kernel.environment%'
30   fragments: ~
31   http_method_override: true
32   assets: ~
33   php_errors:
34     log: false
35   templating:
36     engines: ['twig']
```

1. Избыточность
2. Тип структуры (размерность?)
3. Читаемость
4. Легко сломать?
5. Типы данных

YAML ain't Markup Language

1. Синтаксис на основе отступов
2. Есть различные типы
3. Некоторые типы можно описывать разными способами (массивы, объекты)

1. Избыточность
2. Тип структуры (размерность?)
3. Читаемость
4. Легко сломать?
5. Типы данных



ПРЕДСТАВЛЕНИЕ ДАННЫХ. Ответ HTTP

Protocol CODE STATUS

Header-name: header-value

Header-name: header-value

Body

HTTP/1.1 200 OK

Date: Wed, 27 JUL 2016 11:20:59 GMT

Server: Apache

X-Powered-By: PHP/5.6.3-2ubuntu5wm1

Last-Modified: Wed, 27 JUL 2016 11:20:59 GMT

Content-Language: ru

Content-Type: text/html; charset=utf-8

Content-Length: 2437

Connection: keep-alive

<!DOCTYPE html>

<html>

...

Multipurpose Internet Mail Extension

Тип информации / Способ кодирования

Типы информации:

- Текстовая
- Изображения
- Аудио
- Видео
- ~Исполняемый код
- ...

text/xml

text/html

text/css

image/png

image/jpeg

image/webp

MP3-файл

???

Multipurpose Internet Mail Extension

Тип информации / Способ кодирования

Типы информации:

- Текстовая
- Изображения
- Аудио
- Видео
- ~Исполняемый код
- ...

text/xml

text/html

text/css

image/png

image/jpeg

image/webp

MP3-файл

~~audio/mp3~~

audio/mpeg

Multipurpose Internet Mail Extension

Тип информации / Способ кодирования

Типы информации:

- Текстовая
- Изображения
- Аудио
- Видео
- ~Исполняемый код
- ...

text/xml

text/html

text/css

image/png

image/jpeg

image/webp

MP3-файл

~~audio/mp3~~

audio/mpeg

Простой текст

???

Multipurpose Internet Mail Extension

Тип информации / Способ кодирования

Типы информации:

- Текстовая
- Изображения
- Аудио
- Видео
- ~Исполняемый код
- ...

text/xml

text/html

text/css

image/png

image/jpeg

image/webp

MP3-файл

~~audio/mp3~~

audio/mpeg

Простой текст

text/plain

Multipurpose Internet Mail Extension

Тип информации / Способ кодирования

Типы информации:

- Текстовая
- Изображения
- Аудио
- Видео
- ~Исполняемый код
- ...

text/xml

text/html

text/css

image/png

image/jpeg

image/webp

MP3-файл

~~audio/mp3~~

audio/mpeg

Простой текст

text/plain

Заливаемые файлы

???

Multipurpose Internet Mail Extension

Тип информации / Способ кодирования

Типы информации:

- Текстовая
- Изображения
- Аудио
- Видео
- ~Исполняемый код
- ...

text/xml

text/html

text/css

image/png

image/jpeg

image/webp

MP3-файл

~~audio/mp3~~

audio/mpeg

Простой текст

text/plain

Заливаемые файлы

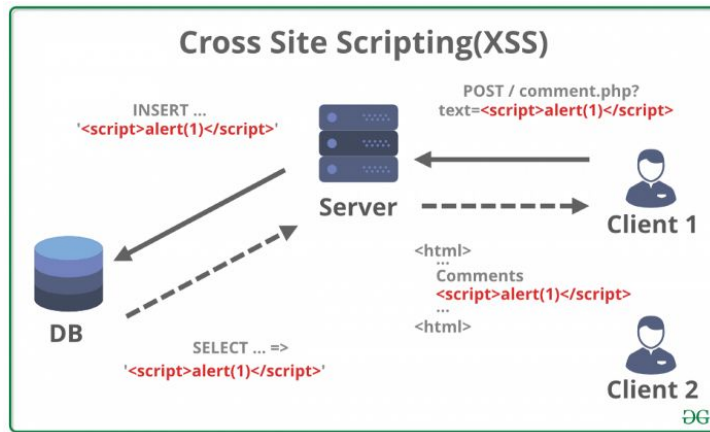
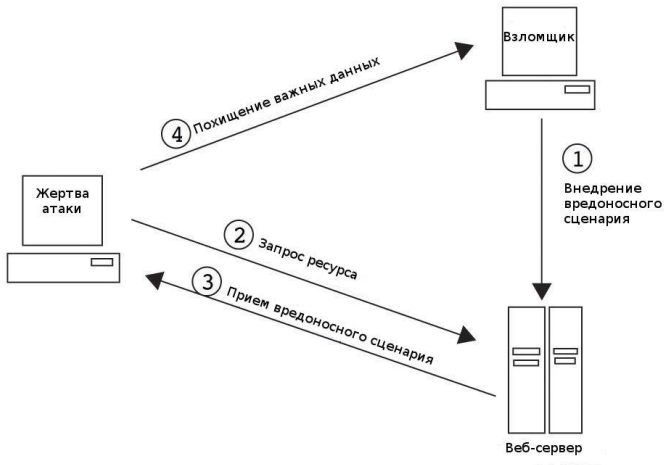
multipart/form-data

```
<form action="" method ="POST" enctype="multipart/form-data">
```

1. Абсолютные пути начинаются со слэша
 - a. Абсолютный путь: `/images/register`
 - b. Относительный путь: `images/register`
2. Абсолютные пути – хорошо!
 - a. `css/style.css` для `/news` = `/css/style.css`
 - b. `css/style.css` для `/news/42` = `/news/css/style.css`
3. Для сайта в контексте придётся вручную добавлять название контекста:
 - a. `${pageContext.request.contextPath}/css/style.css`
 - b. `<c:url value="/css/style.css"/>`

Не забудьте добавлять контекст и в Java-коде при генерации ссылок!

XSS. HTML entities



<
>
&
&#xxxx;