j2ee + Spring

Лекция 2.
Сервер приложений

Application Server. Что это?

Как это делают обычно?

```
public class Main_1
{
    public static void main( String[ ] args )
    {
        Sensor sensor = new Sensor( "http://automation.local/sensor" );
        Actuator actuator = new Actuator( "http://automation.local/actuator" );

        IntellectualController controller = new IntellectualController( "Controller.1" );
        controller.setSensor( sensor );
        controller.setActuator( actuator );

        while ( true )
        {
            controller.doTheWork( );
        }
    }
}
```

Какие проблемы Вы здесь видите?

ApplicationServer. Что это?

Как это делают обычно?

```
public class Main_1
{
    public static void main( String[ ] args )
    {
        Sensor sensor = new Sensor( "http://automation.local/sensor" );
        Actuator actuator = new Actuator( "http://automation.local/actuator" );

        IntellectualController controller = new IntellectualController( "Controller.1" );
        controller.setSensor( sensor );
        controller.setActuator( actuator );

        while ( true )
        {
            controller.doTheWork( );
        }
    }
}
```

Какие проблемы Вы здесь видите?

1. А что если нам необходимо загрузить обновление нашей программы?
2. А что если нам необходимо соединение с базой данных, как мы это обеспечим?
3. Как будет проводится управление временем жизни ресурсов?
4. …

ApplicationServer. Что это?

Что бы Вы могли предложить для решения указанных проблем?

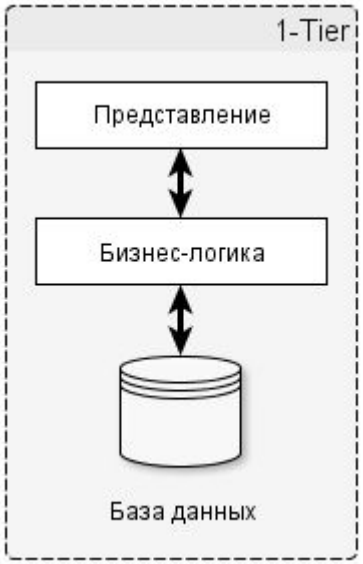
1. А что если нам необходимо загрузить обновление нашей программы?
2. А что если нам необходимо соединение с базой данных, как мы это обеспечим?
3. Как будет проводится управлением временем жизни ресурсов?
4. …

Решение подсказывает заголовок слайда, но не будем торопиться.

N-Tier

Многоуровневые архитектуры …

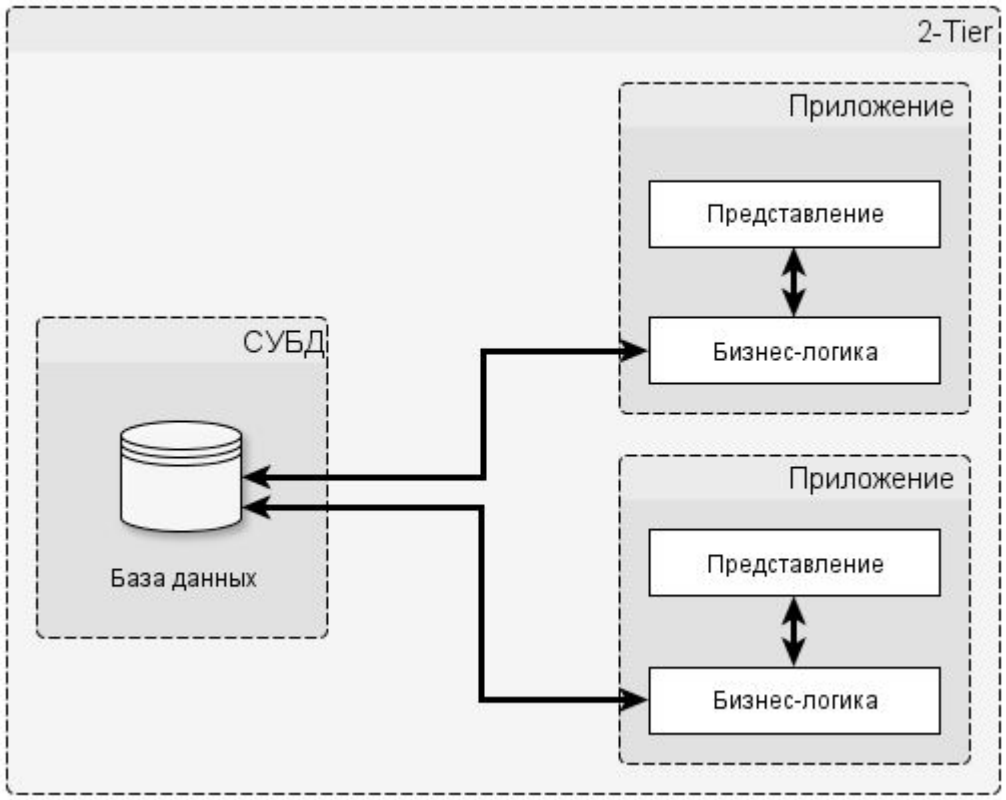
Когда-то приложения были простыми:



N-Tier

Многоуровневые архитектуры …

Но потом, понадобилось обособить базу данных. Появились СУБД:



N-Tier

Многоуровневые архитектуры …

Но были проблемы, разные … Как вы думаете, какие?

N-Tier

Многоуровневые архитектуры …

Но были проблемы, разные … Как вы думаете, какие?

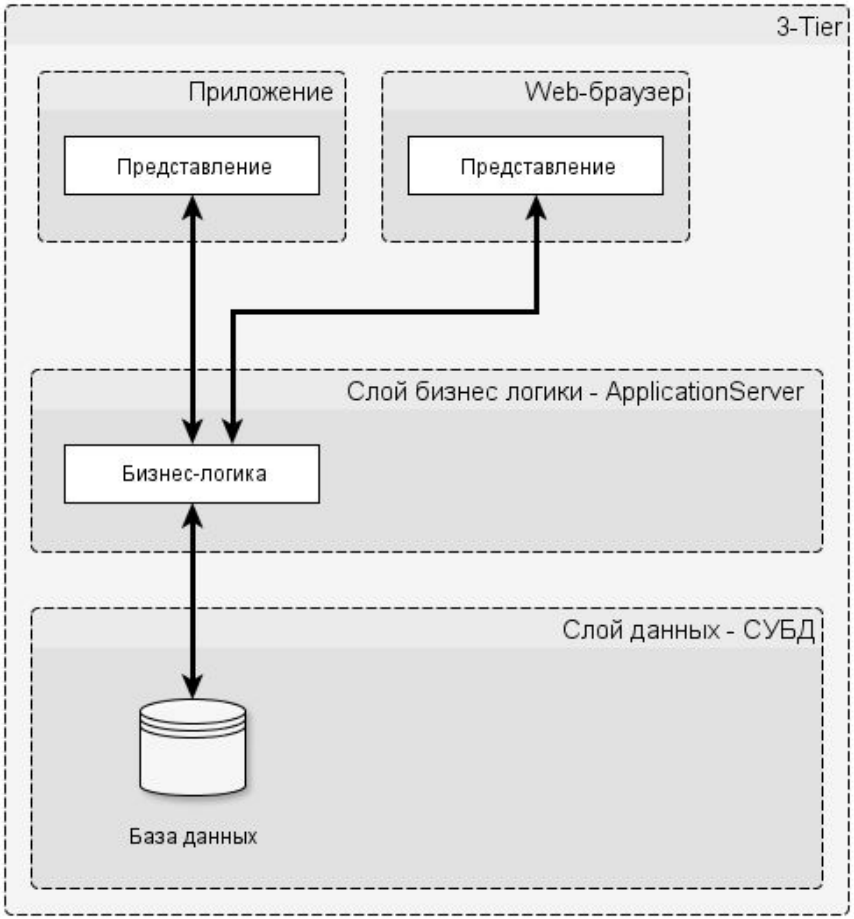
1. Обновление данных в СУБД (+Транзакции)
2. Версионность
3. Обновление
4. Управление слоем бизнес-логики
5. Сокрытие слоя бизнес логики, как средство борьбы с пиратством

Понадобилось выделение слоя бизнес-логики …

N-Tier

Многоуровневые архитектуры …

Понадобилось выделение слоя бизнес-логики …



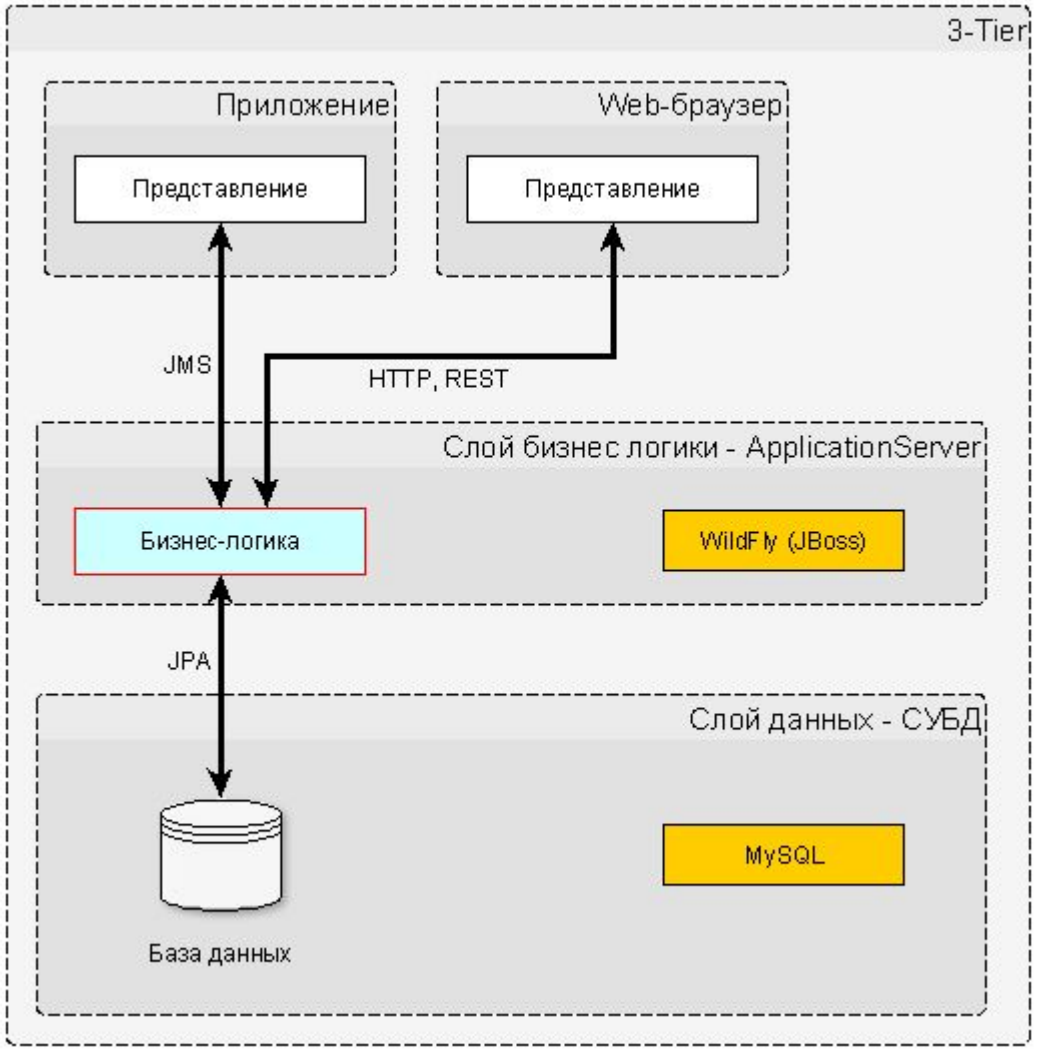
N-Tier

Многоуровневые архитектуры …

Можно добавлять еще слоев, делая 4-, 5-, … Tier приложения. Например, балансировщик нагрузки …

ApplicationServer

Но для наших целей достаточно рассмотреть 3-Tier (трехуровневые) архитектуры.



ApplicationServer

Таким образом сервер приложений позволяет:

1. Выделить бизнес логику в отдельный слой и управлять ею
2. Централизованно конфигурировать и переконфигурировать приложения, управлять их жизненным циклом
3. Обеспечивать безопасность как самого кода, так и его выполнения
4. Улучшить производительность за счет нахождения обработки «рядом» с источником данных
5. Снизить суммарную стоимость владения (TCO)
6. Обеспечить поддержку транзакций

Упаковка enterprise-приложений

Нам интересны два способа - .war и .ear

.war – Web Archive

- Включает зависимости (WEB-INF/lib)
- Включает все классы (как jar, но в WEB-INF/classes)
- Включает конфигурационный файл web.xml

.ear – Enterprise Archive

- Может включать в себя несколько .war архивов (соответственно несколько приложений)
- Содержит файлы описания приложений, например, jboss-all.xml или в общем случае application.xml

War с помощью maven

```
...
<packaging>war</packaging>
...
<build>
    <plugins>
        ...
        <plugin>
            <groupId>org.apache.maven.plugins</groupId>
            <artifactId>maven-war-plugin</artifactId>
            <configuration>
                <webXml>src\main\webapp\WEB-INF\web.xml</webXml>
            </configuration>
        </plugin>
        ...
    </plugins>
</build>
...
```

Содержимое проекта для war

В общем такое же как и для обычного maven проекта
Нужна дополнительная директория src/main/webapp

web.xml

Содержимое web.xml:

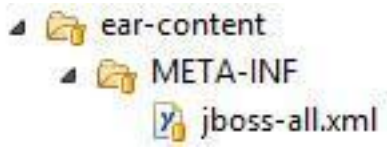
```
<?xml version="1.0" encoding="UTF-8"?>
<web-app version="2.4"
      xmlns="http://java.sun.com/xml/ns/j2ee"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee
                          http://java.sun.com/xml/ns/j2ee/web-app_2_4.xsd">

    <display-name>...</display-name>

    <context-param>
        <param-name>contextConfigLocation</param-name>
        <param-value>classpath:beans.xml</param-value>
    </context-param>

    <listener>
        <listener-class>
            org.springframework.web.context.ContextLoaderListener
        </listener-class>
    </listener>
</web-app>
```

ear с помощью maven

```
...
<packaging>ear</packaging>
...
<build>
    <plugins>
        ...
        <plugin>
            <artifactId>maven-ear-plugin</artifactId>
            <configuration>
                <modules>
                    <webModule>
                        <groupId>ru.mrdekk</groupId>
                        <artifactId>warp</artifactId>
                        <contextRoot>/warp</contextRoot>
                    </webModule>
                </modules>
                <displayName>WAR-Project</displayName>
                <generateApplicationXml>true</generateApplicationXml>
                <resourcesDir>target/classes</resourcesDir>
            </configuration>
        </plugin>
        ...
    </plugins>
</build>
...
```

Содержимое проекта для ear

Создаем earcontent и размещаем в нем необходимые конфигурации



jboss-all.xml

Содержимое weblogic-application.xml:

```
<jboss umlns="urn:jboss:1.0">
  <jboss-deployment-structure xmlns= "urn:jboss:deployment-structure:1.1">
    <deployment>
      <exclusions>
        <module name="org.joda.time" slot="main" />
      </exclusions>
      <dependencies>
        <module name="com.oracle.mysql" export="true" />
      </dependencies>
    </deployment>
    <sub-deployment name= "lec2-0.0.1.war" >
      <exclusions>
        <module name="org.joda.time" slot="main" />
      </exclusions>
    </sub-deployment>
  </jboss-deployment-structure>
</jboss>
```

war & ear

Примеры смотрите в git

Задача

1. Распаковать, запустить сервер приложений WildFly 8.2.0.Final
2. Добавить в ваш проект два новых модуля lec2war, lec2ear типов war и ear соответственно. Настроить их надлежащим образом.
3. Проект lec1 из лекции 1 подключить в качестве зависимости к lec2war, beans.xml перенести в src/main/resources war проекта.
4. Работу, которую вы делали в методе main основного класса перенести в init метод специального класса. Воспользовавшись init-method атрибутом при определении bean в контексте Spring
5. Выполнить deploy вашего проекта на сервер приложений WildFly развернутый на шаге 1. Проверить что действие выполняется проверив журнал server.log на наличие в нем записей