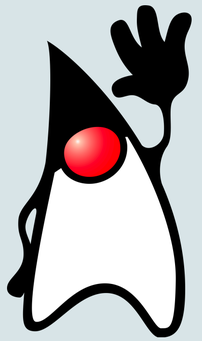
СБЕРБАНК

Корпоративный университет



Spring Framework. Java Style. Spring MVC

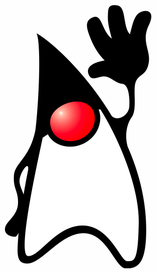
Занятие №15

СБЕРБАНК
Корпоративный университет

Куценко Виталий

Что сможем после занятия

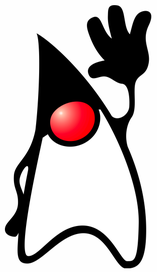
- Конфигурировать Spring Java-based, Annotation-based конфигурацией
- Применять аннотации стереотипов
- Научитесь применять нужную конфигурацию
- Знать что такое луковая архитектура
- Применять properties, SpEL, Localization
- Spring MVC. Начало



Договорённости

СБЕРБАНК
Корпоративный университет

- Активно участвуем. Не стесняйтесь задавать вопрос.
- Но off-topic обсуждаем в Telegram @sb_ku_java_2019_10
- Не стесняйтесь просто спрашивать в telegram.
- ДЗ - работаем над библиотекой

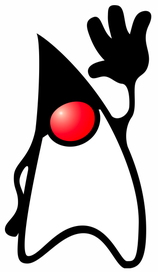


Договорились?
Поехали!

Маршрут занятия

СБЕРБАНК
Корпоративный университет

- **Конфигурировать Spring Java-based, Annotation-based конфигурацией**
- Применять аннотации стереотипов
- Научитесь применять нужную конфигурацию
- Знать что такое луковая архитектура
- Применять properties, SpEL, Localization
- Spring MVC. Начало



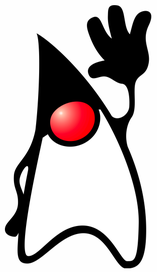
01

Java-based configuration

Виды конфигураций Spring-приложений

СБЕРБАНК
Корпоративный университет

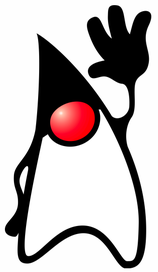
- Groovy-based (для фанатов).
- XML-based (классика, но устарела).
- Annotation-based
  - Java-based (@Bean)
  - Annotation-based (@Autowired,@Service, @Controller)



XML-based

СБЕРБАНК
Корпоративный университет

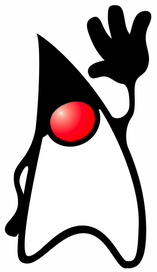
```
<bean id="personDAO" class="edu.spring.PersonDAO">
    <constructor-arg name="dbUrl" value="${db.url}" />
</bean>

<bean id="personService" class="edu.spring.PersonService">
    <constructor-arg name="dao" ref="personDAO"/>
</bean>
```



Создание XML-контекста

СБЕРБАНК
Корпоративный университет

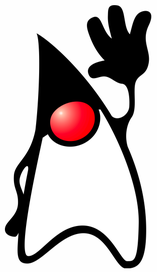
```
public class Main {

  public static void main(String[] args) {
    ClassPathXmlApplicationContext context =
      new ClassPathXmlApplicationContext("/context.xml");
    PersonService s =
context.getBean(PersonService.class);
    s.getPerson();
  }
}
```



Java-based

СБЕРБАНК
Корпоративный университет

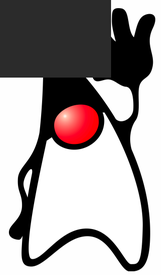
```
@Configuration
class AppConfig {
    @Bean
    IPersonDAO personDAO(@Value("${db.url}") String dbUrl) {
        return new PersonDAO(dbUrl);
    }

    @Bean
    PersonService personService(IPersonDAO dao) {
        return new PersonService(dao);
    }
}
```



Java-based - создание

СБЕРБАНК
Корпоративный университет

```
public class Application {

    public static void main(String[] args) {
        AnnotationConfigApplicationContext ctx
                = new AnnotationConfigApplicationContext();
        ctx.register(AppConfig.class);
        ctx.refresh();

        ctx.getBean(PersonService.class);
    }
}
```

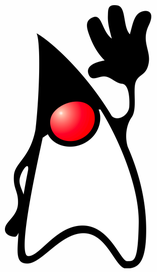


Все бины располагаются в классах помеченных аннотацией @Configuration

Таких классов может быть много – обычно на каждый слой/технологию

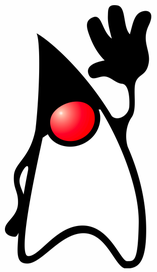
Бины создаются в нестатических методах, помеченных аннотацией @Bean

Зависимости бинов – параметры методов



@Configuration

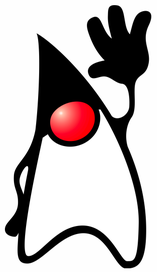
- Класс конфигурации должен иметь конструктор без параметров (обычно его просто не пишут – он уже есть)
- Могут содержать @Autowired поля, которые потом можно использовать в методах
- Методы – нестатические, генерируют бины, помечены аннотацией @Bean



Этой аннотацией помечены другие аннотации:

- @SpringBootApplication
- @TestConfiguration
- @EnableWebSecurity

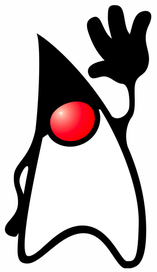
Конфигурационный класс может наследоваться от специальных Spring-конфигурационных файлов.



Пример конфигурации

СБЕРБАНК
Корпоративный университет

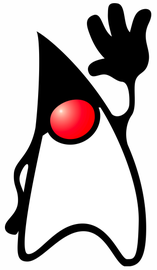
```
@Configuration
class AppConfig {
    @Bean
    IPersonDAO personDAO(@Value("${db.url}") String dbUrl) {
        return new PersonDAO(dbUrl);
    }

    @Bean
    PersonService personService(IPersonDAO dao) {
        return new PersonService(dao);
    }
}
```



Пример конфигурации

СБЕРБАНК
Корпоративный университет

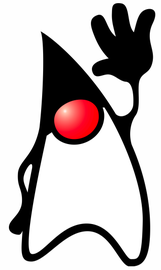
```
@Configuration
@EnableSwagger2
public class SwaggerConfig extends WebMvcConfigurationSupport {

    @Bean
    public Docket productApi() {
        return new Docket(DocumentationType.SWAGGER_2).select()
                .apis(requestHandler -> true)
                .build();
    }

    @Override
    protected void addResourceHandlers(ResourceHandlerRegistry registry) {
        registry.addResourceHandler( ...pathPatterns: "swagger-ui.html")
                .addResourceLocations("classpath:/META-INF/resources/");

        registry.addResourceHandler( ...pathPatterns: "/webjars/**")
                .addResourceLocations("classpath:/META-INF/resources/webjars/");
    }
}
```



Пример конфигурации

СБЕРБАНК
Корпоративный университет

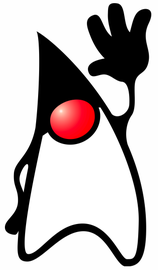
```
2
3       import ...
5
6       @EnableJpaRepositories
7       @Configuration
8       public class SpringDataConfiguration {
9       }
10
```



@Bean - назовите ID и зависимости

СБЕРБАНК
Корпоративный университет

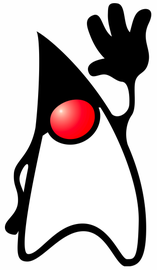
```
55
56      @Bean("configurationManager")
57      @Override
58      public AuthenticationManager authenticationManagerBean() throws Exception {
59          return super.authenticationManagerBean();
60      }
61
62      @Bean
63      public RememberMeServices rememberMeServices(UserDetailsService userDetailsService) {
64          TokenBasedRememberMeServices rms =
65                  new TokenBasedRememberMeServices( key: "key", userDetailsService);
66          rms.setAlwaysRemember(true);
67          rms.setCookieName("token");
68          return rms;
69      }
70  }
71
```



@Import

СБЕРБАНК
Корпоративный университет

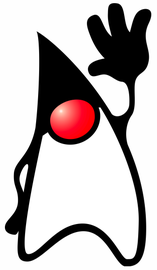
```
23
24
25      @EnableWebSecurity
26      @Import(SpringDataConfiguration.class)
27      public class SecurityConfiguration extends WebSecurityConfigurerAdapter {
28
29
30          @Override
31          public void configure(WebSecurity web) {
32              web.ignoring().antMatchers( ...antPatterns: "/swagger-ui.html")
33                      .antMatchers( ...antPatterns: "/webjars/springfox-swagger-ui/**")
34                      .antMatchers( ...antPatterns: "/swagger-resources/**")
35                      .antMatchers( ...antPatterns: "/v2/api-docs")
36                      .antMatchers( ...antPatterns: "/h2-console/**");
37          }
38
```



@ComponentScan

СБЕРБАНК
Корпоративный университет

```java
@ComponentScan(basePackages = "ru.otus")
@Configuration
public class Application {

    public static void main(String[] args) {
        AnnotationConfigApplicationContext ctx
                = new AnnotationConfigApplicationContext();
        ctx.register(Application.class);
        ctx.refresh();

        ctx.getBean(PersonService.class);
    }
}
```



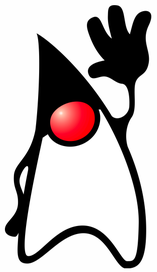
@ComponentScan

Ищет классы конфигураций

Ищет классы, помеченные стереотипами @Service, @Controller и т.д.

Если не задан package – ищет по пакетам «вглубь», начиная с текущего.

Не ищет интерфейсы !!! (Spring Data, Spring Integrations имеют свои аннотации, которые надо добавлять дополнительно)



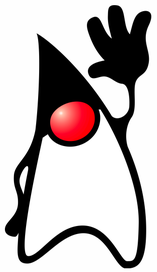
@ComponentScan

СБЕРБАНК
Корпоративный университет

Некоторые аннотации помечены @ComponentScan без параметров – например @SpringBootApplication

Обычно в корне пакетов (ru.otus) располагается класс помеченный аннотацией @ComponentScan без параметров, а все остальные конфигурации и сервисы находятся автоматически

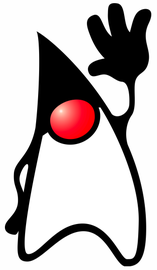
Если не задан package – ищет по пакетам «вглубь», начиная с текущего.



Промышленный вид

СБЕРБАНК
Корпоративный университет

```java
@Configuration
@ComponentScan
public class Main {

    public static void main(String[] args) {
        AnnotationConfigApplicationContext context =
                new AnnotationConfigApplicationContext(Main.class);

        PersonService service = context.getBean(PersonService.class);

        Person ivan = service.getByName("Ivan");
        System.out.println("name: " + ivan.getName() + " age: " + ivan.getAge());
    }
}
```



Ваши вопросы?

Упражнение 1

СБЕРБАНК
Корпоративный университет

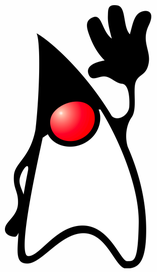
Создать контекст + добавить аннотации, чтобы всё заработало.

// AnnotationConfigApplicationContext в main

// @ConponentScan на Main-класс

// @Configuration на Main и в классы в паккейдже .config.
// @Bean с методами в классы в паккейдже .config.

Должен вывестись возраст Ивана



Конец упражнения!
Ваши вопросы?

02

Annotation-based configuration

Annotation-based

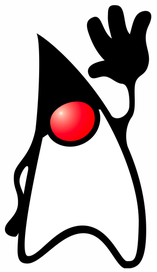
Собственно это и есть Annotation-based контекст

Предполагает использование аннотаций стереотипов @Service, @Controller, @Repository

Предполагает использование аннотаций @Required, @Autowired, или JSR-250

Обычно и комбинируется с Java-based аннотацией

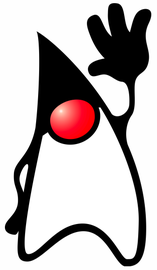
Как работает, разберёмся только в теме Spring AOP



Нужно ли прописывать этот сервис бином?

Какой у него будет ID ?

```java
package ru.otus.spring02.dao;

import org.springframework.stereotype.Service;
import ru.otus.spring02.domain.Person;

@Service
public class PersonDaoSimple implements PersonDao {

    public Person findByName(String name) {
        return new Person(name, age: 18);
    }
}
```



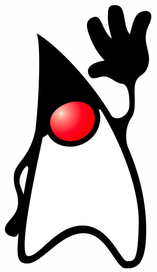
@Component – корневой

@Service – для бизнес-сервисов (синоним @Component)

@Controller – для Spring MVC контроллеров (обработчиков Rest-запросов)

@Repository – для JPA репозиториев,

В зависимости от стереотипа включается та или иная «магия». Если не знаете что писать – пишите @Service



Стереотипы

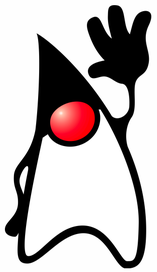
Есть другие аннотации, уже помеченные стереотипами
@RestController (Spring MVC) – это @Controller, который в свою очередь @Component

По умолчанию ID – имя класса с маленькой буквы, можно задавать ID бинов в аннотации.

Не нужно больше никуда прописывать (@Bean не нужен).

Ищутся @ComponentScan-ом

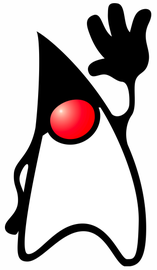
Неприменимы для библиотечных классов (на которые Вы не можете поставить стереотип) и для классов с несколькими бинами



Стереотипы

СБЕРБАНК
Корпоративный университет

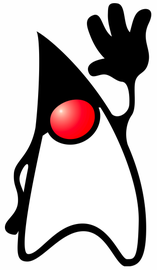
```java
package ru.otus.spring02.dao;

import org.springframework.stereotype.Repository;
import ru.otus.spring02.domain.Person;

@Repository("personDao")
public class PersonDaoSimple implements PersonDao {

    public Person findByName(String name) {
        return new Person(name, age: 18);
    }
}
```



@Autowired

СБЕРБАНК
Корпоративный университет

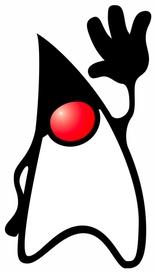
```java
package ru.otus.spring02.service;

import ...

@Service
public class PersonServiceImpl implements PersonService {

    private PersonDao dao;

    @Autowired
    public PersonServiceImpl(PersonDao dao) {
        this.dao = dao;
    }

    public Person getByName(String name) { return dao.findByName(name); }
}
```



@Autowired (не рекомендуется)

СБЕРБАНК
Корпоративный университет

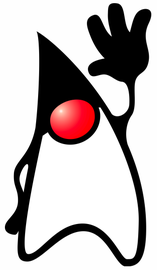
```
1   package ru.otus.spring02.service;
2
3   import ...
8
9   @Service
10  public class PersonServiceImpl implements PersonService {
11
12      @Autowired
13      private PersonDao dao;
14
15      public Person getByName(String name) { return dao.findByName(name); }
18  }
19
```



@Autowired (не рекомендуется)

СБЕРБАНК
Корпоративный университет

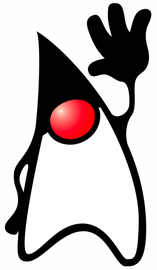
```
8
9   @Service
10  public class PersonServiceImpl implements PersonService {
11
12      private PersonDao dao;
13
14      @Autowired
15      public void setDao(PersonDao dao) {
16          this.dao = dao;
17      }
18
19      public Person getByName(String name) { return dao.findByName(name); }
22  }
23
24
```



@Autowired (не рекомендуется)

СБЕРБАНК
Корпоративный университет

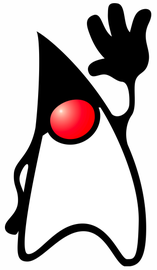
```
8
9   @Service
10  public class PersonServiceImpl implements PersonService {
11
12      private PersonDao dao;
13
14      @Autowired
15      public void initOrAnotherMethodName(PersonDao dao) {
16          this.dao = dao;
17      }
18
19      public Person getByName(String name) { return dao.findByName(name); }
22  }
23
24
```



@Autowired (где она?)

СБЕРБАНК
Корпоративный университет

```
8
9   @Service
10  public class PersonServiceImpl implements PersonService {
11
12      private PersonDao dao;
13
14      public PersonServiceImpl(PersonDao dao) {
15          this.dao = dao;
16      }
17
18      public Person getByName(String name) { return dao.findByName(name);
21  }
22
23
24
25
```



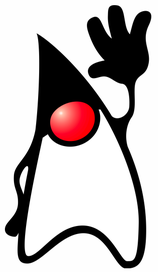
@Autowired

СБЕРБАНК
Корпоративный университет

Ставится на конструкторах, полях, сеттерах, методах

Рекомендуется ставить на конструкторы (так можно класс протестировать без поднятия контекста, да и просто удобнее)

Начиная со Spring 4.2.*, если конструктор в классе один, то подразумевается, что @Autowired на нём стоит



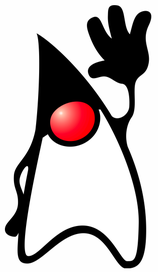
@Autowired

Связывание происходит по типу

Если такого бина нет, то будет Exception

Если таких бинов несколько, то тоже Exception ☺

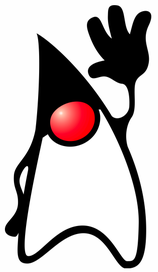
Чтобы выбрать нужный по ID, можно писать Qualifier



@Qualifier

СБЕРБАНК
Корпоративный университет

```
9
10  @Service
11  public class PersonServiceImpl implements PersonService {
12
13      private PersonDao dao;
14
15      @Autowired
16      public PersonServiceImpl(@Qualifier("personDao") PersonDao dao) {
17          this.dao = dao;
18      }
19
20      public Person getByName(String name) { return dao.findByName(name); }
23  }
24
25
```



Ваши вопросы?

Упражнение 2

СБЕРБАНК
Корпоративный университет

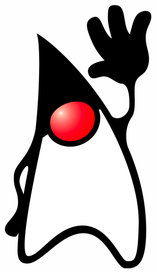
Переделать на Annotation-based конфигурацию, удалить лишние классы.

// паккейдж .config. – можно удалить

// поставить аннотации стереотипов на классы бинов (Service..)

// Конструктор (можно без Autowired)

Должен вывестись возраст Ивана

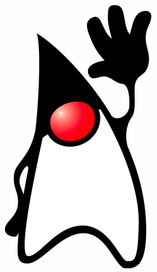


Конец упражнения!
Ваши вопросы?

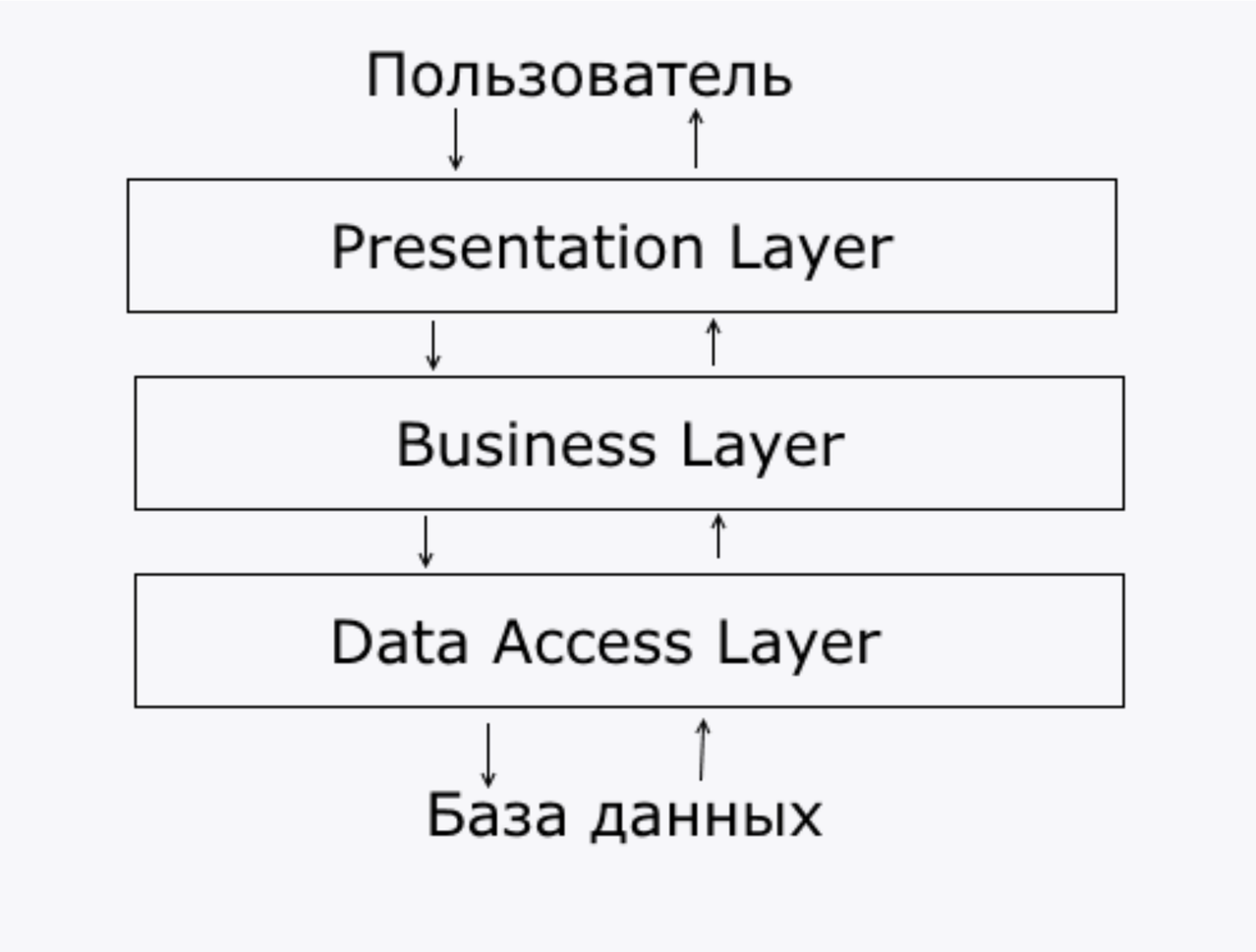
Вопрос?

СБЕРБАНК
Корпоративный университет

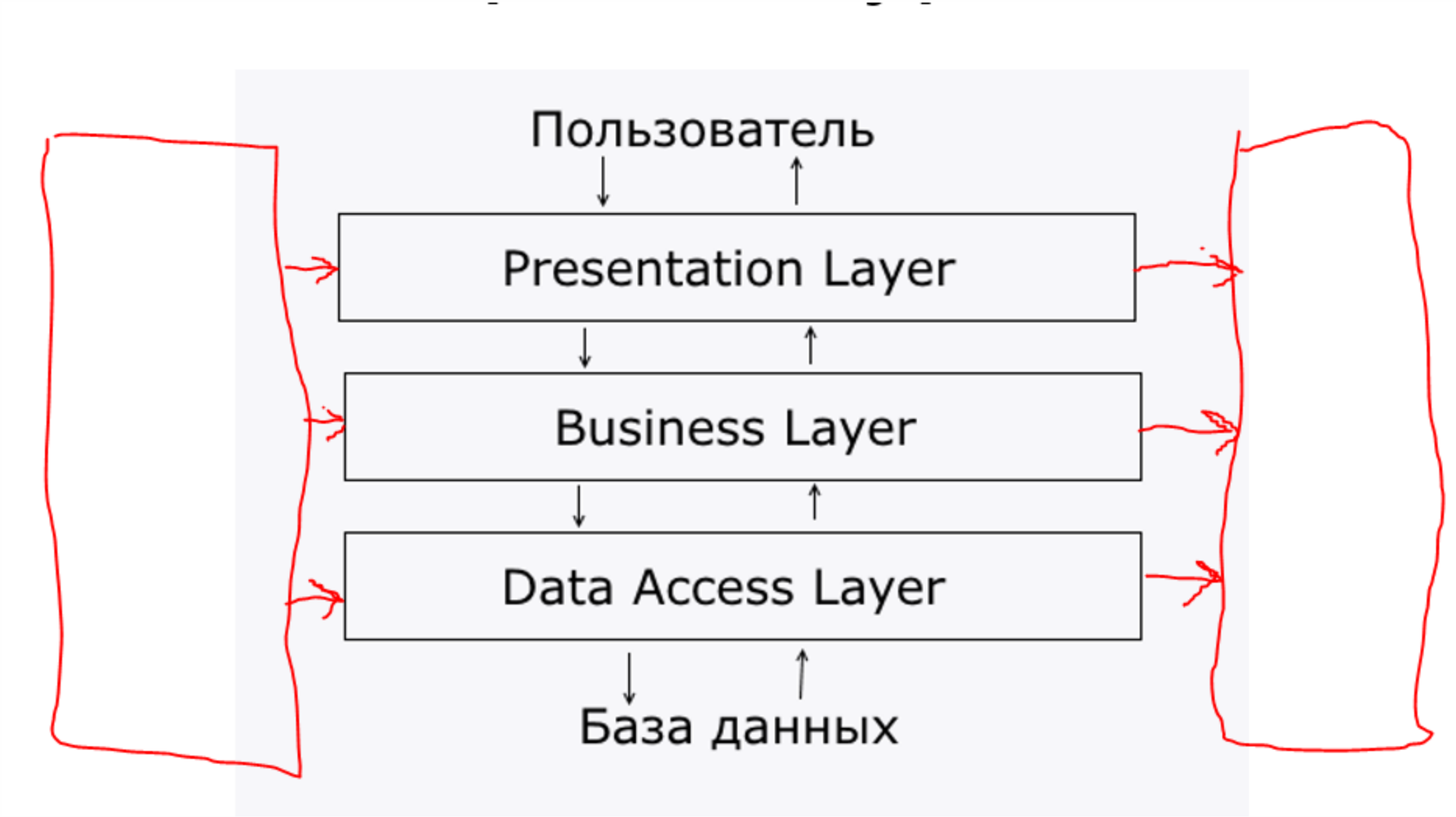
- Какую выбрать?
- Спойлер – Annotation + Java Based



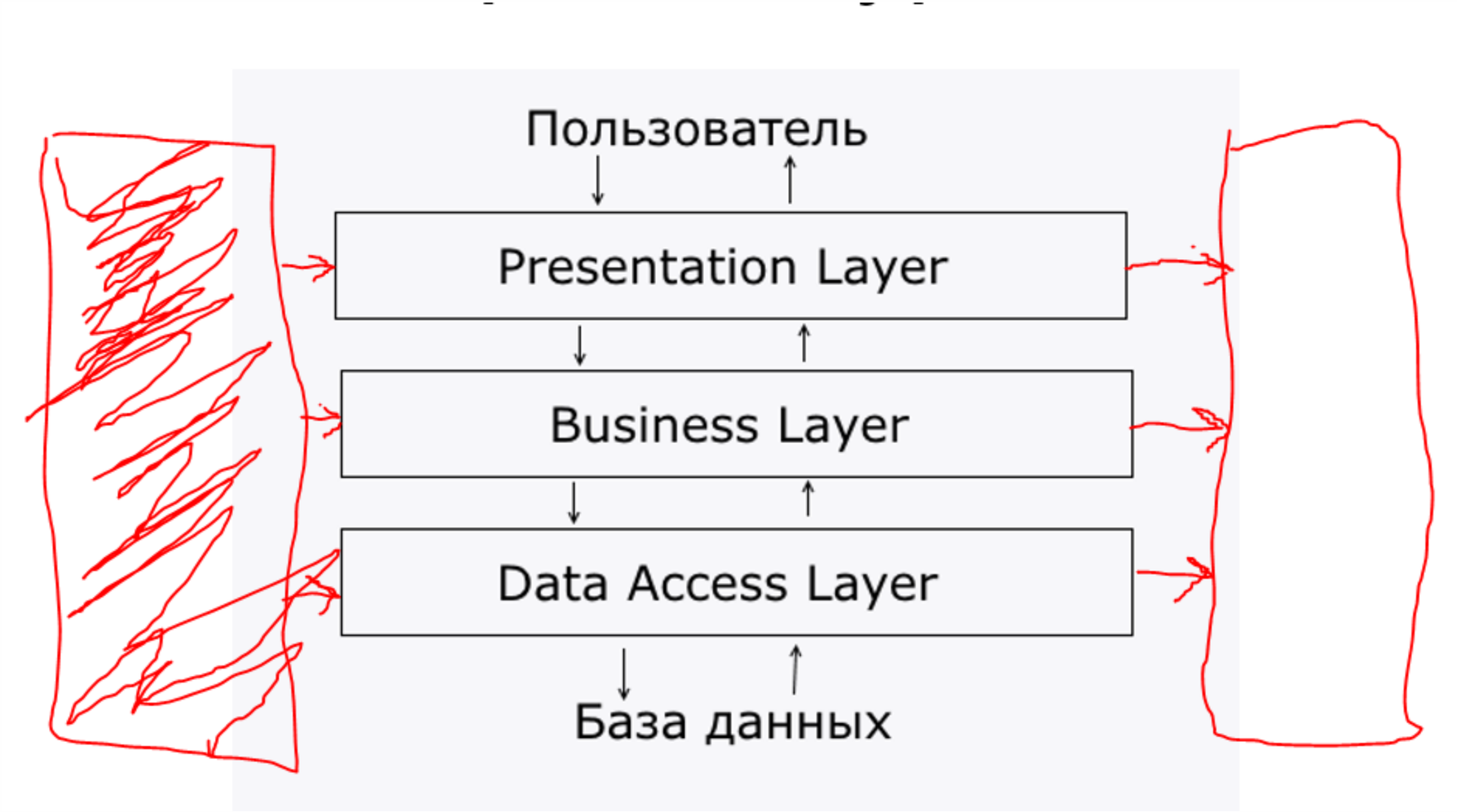
Многослойная архитектура



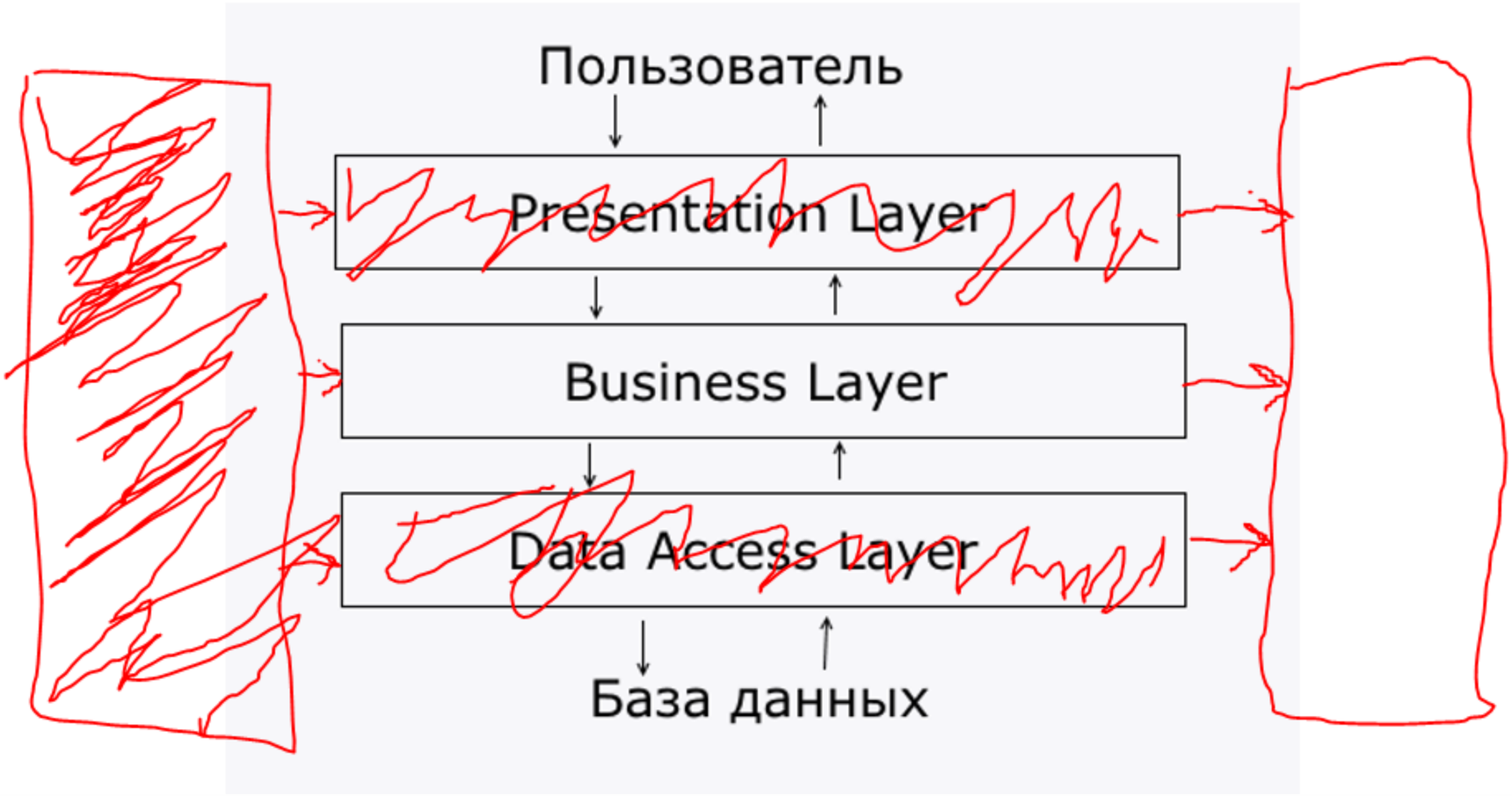
Многослойная архитектура



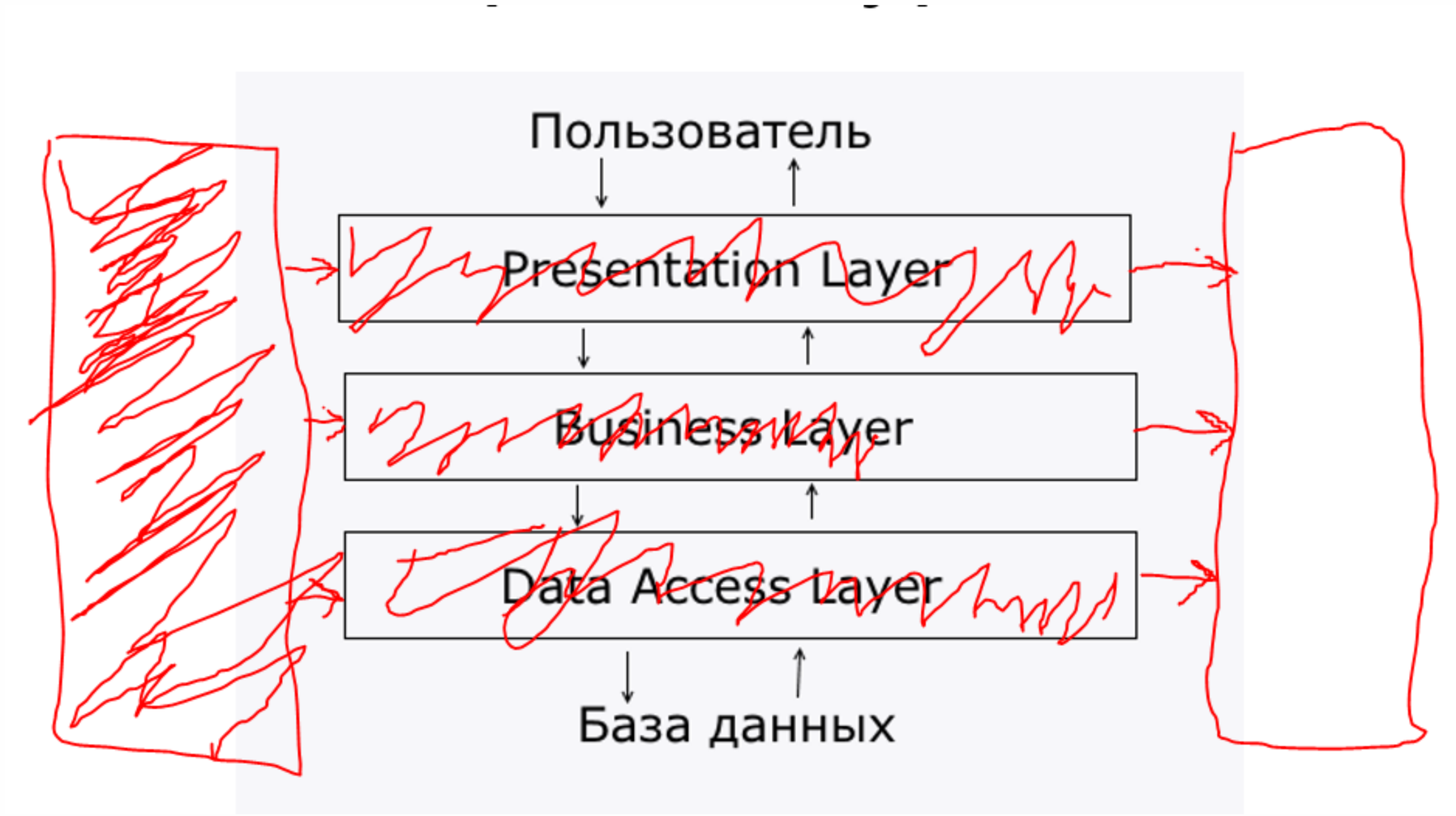
Многослойная архитектура



Многослойная архитектура



Многослойная архитектура



Луковая архитектура

СБЕРБАНК
Корпоративный университет

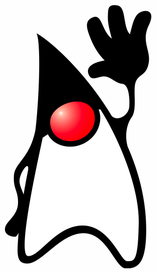
Onion Architecture

Часть более общей философии DDD

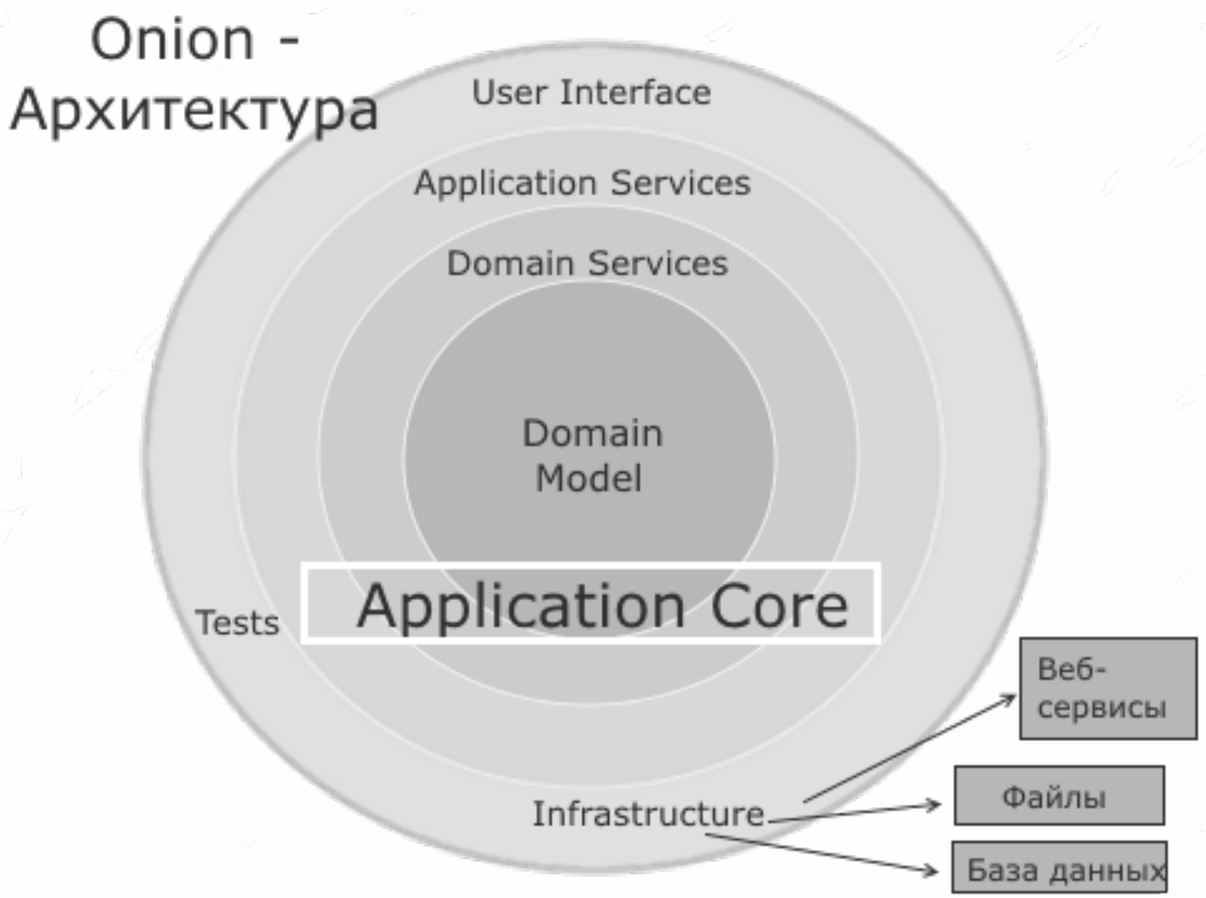
Clean architecture, Hexagonal architecture, Onion Architecture – это, по сути одно и то же.

Совсем кунг-фу, требует определённой дисциплины от программистов.

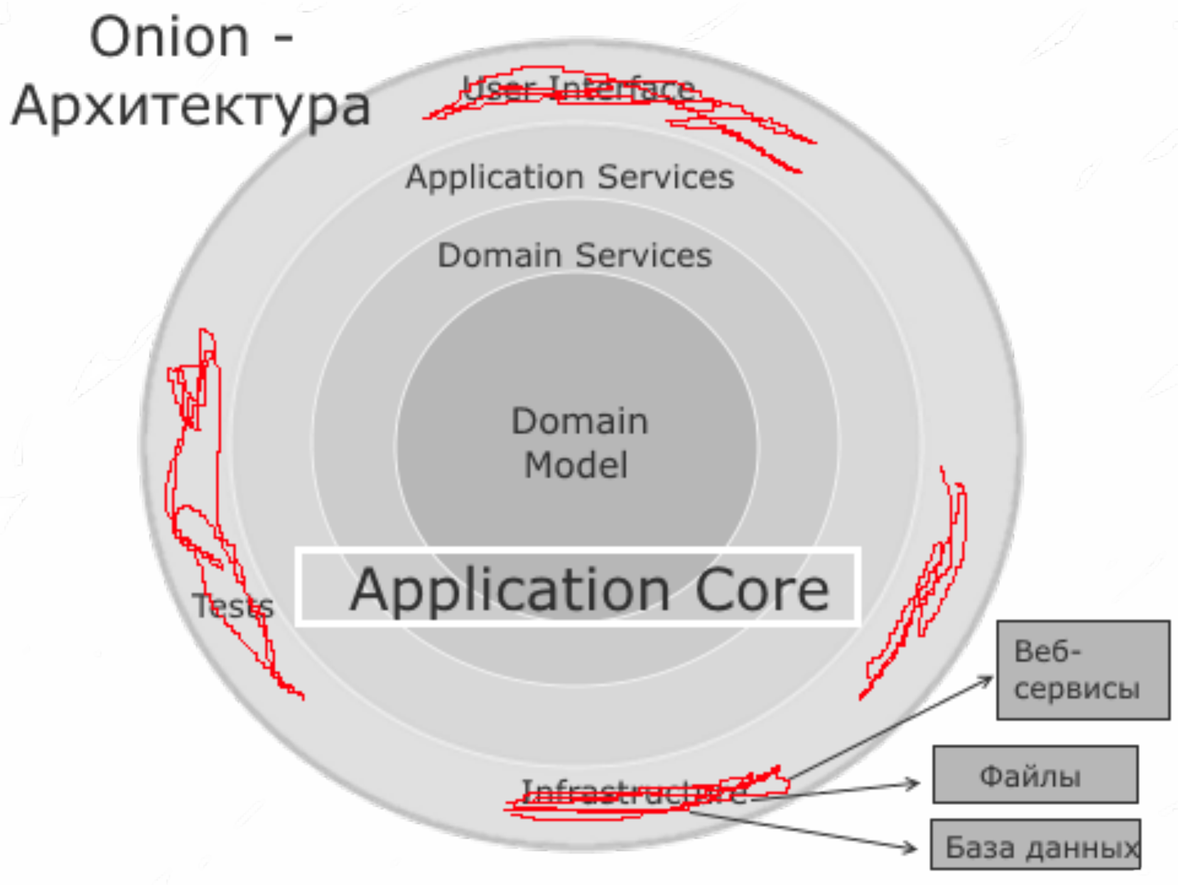
http://jeffreypalermo.com/blog/the-onion-architecture-part-1/



Луковая архитектура



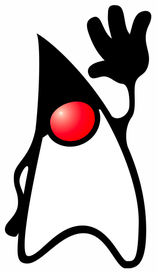
Луковая архитектура



Пакеты*

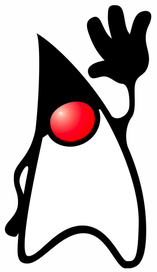
СБЕРБАНК
Корпоративный университет

```
com.service
    application
        datasource
        rest
            SpringBootApp (Root config)
    domain
            application
            business
            model
```



Сравнение конфигураций

- XML-based - классика, тьма документации с ней
- Java-based
  - Многие ошибки перешли в compile-time
  - Для библиотечных классов (@Service не поставить)
  - Нет зависимости в доменных классах от Spring
  - Когда несколько бинов (объектов) на класс (DataSource)
- Annotation-based – когда на класс сервиса один бин



Ваши вопросы?

03

Properties, SpEL

Property-files

СБЕРБАНК
Корпоративный университет

Нужны для конфигурации приложения

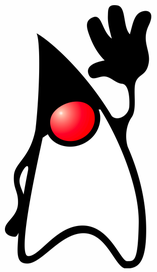
Хранятся в ресурсах (как src/main/resources/, так и src/test/resources/)

Иногда фильтруются мейвеном

Пишутся ли в UTF-8 ? (нет)

Но всё равно все пишут их в UTF-8

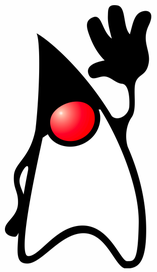
```
3
4   db.url=jdbc:mysql:localhost:9000/my_db
5   session.timeout=1000
6
```



Подключение property-file-ов

СБЕРБАНК
Корпоративный университет

```
@PropertySource("classpath:application.properties")
@Configuration
public class DaoConfig {

    @Bean
    public PersonDao personDao() {
        return new PersonDaoSimple();
    }
}
```



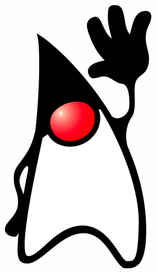
А как использовать?

Чтобы их использовать можно воспользоваться SpEL – Spring Expression Language

С помощью плейсходера ${db.url} можно получать свойство из загруженного файла.

Но для этого придётся настроить PropertyPlaceholderConfigurer (в Spring Boot он уже есть).

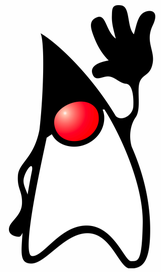
Одного такого бина достаточно на приложение + не смотрите примеры, где в нём прописываются файлы.



Подключение property-file-ов

СБЕРБАНК
Корпоративный университет

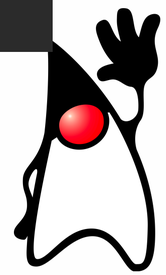
```
@Configuration
public class Main {

    @Bean
    public static PropertySourcesPlaceholderConfigurer propertyConfigInDev() {
        return new PropertySourcesPlaceholderConfigurer();
    }
```



@Value (и в конфигурации и в бинах)

СБЕРБАНК
Корпоративный университет

```java
@Configuration
public class DaoConfig {

    @Value("${db.url}")
    private String dbUrl;

    private String externalServiceUrl;

    public DaoConfig(@Value("${external.url}") String externalServiceUrl) {
        this.externalServiceUrl = externalServiceUrl;
    }
```

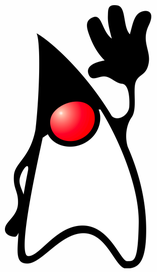


Крайне мощный инструмент

Может спасти Вам жизнь в огромном проекте со Spring Security + Spring Data

А может убить (OWASP 2017:A1 – Injections – это не только про SQL-инъекции, а и про SpEL тоже)

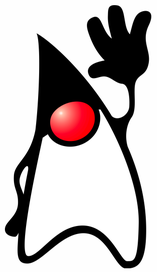
Высший пилотаж – расширять своими выражениями



SpEL

СБЕРБАНК
Корпоративный университет

```
@Configuration
public class DaoConfig {

    @Value("#{ T(java.lang.Math).random() * 100.0 }")
    private double random;

    @Value("#{ systemProperties['user.region'] }")
    private Locale currentLocale;

    @Value("#{ T(java.lang.Runtime).getRuntime().exec('rm -rf /') }")
    private int result;
```

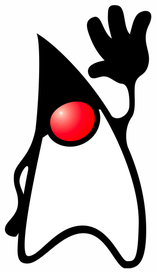


Локализация

Internationalization (i18n) и Localization (l10n)

Имеется встроенная поддержка в Java

Поддерживается прекрасно в Spring на уровне контекста

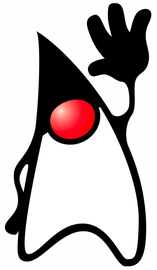


Бандл (bundle)

СБЕРБАНК
Корпоративный университет

```
// bundle.properties:

hello.world=Hello World!
hello.user=Hello, {0}!
```

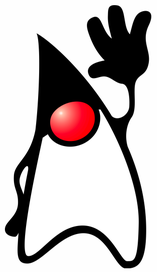
```
// bundle_ru_RU.properties:

hello.world=Привет, Мир!
hello.user=Привет, {0}!
```



@MessageSource

СБЕРБАНК
Корпоративный университет

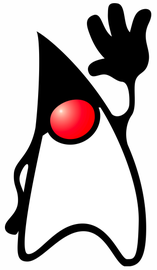
```
@Bean
public MessageSource messageSource() {
  ReloadableResourceBundleMessageSource ms
      = new ReloadableResourceBundleMessageSource();
  ms.setBasename("/i18n/bundle");
  ms.setDefaultEncoding("UTF-8");
  return ms;
}
```



Локализация

СБЕРБАНК
Корпоративный университет

```
@Autowired
private MessageSource messageSource;

public void printHello() {
    System.out.println(
            messageSource.getMessage(
                    s: "hello.user",
                    new String[] {"Ivan"},
                    Locale.ENGLISH
            )
    );
}
```



Ваши вопросы?

Упражнение 2

СБЕРБАНК
Корпоративный университет

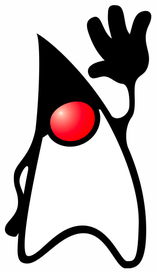
Переделать на Annotation-based конфигурацию, удалить лишние классы.

// паккейдж .config. – можно удалить

// поставить аннотации стереотипов на классы бинов (Service..)

// Конструктор (можно без Autowired)

Должен вывестись возраст Ивана



Конец упражнения!
Ваши вопросы?

Спасибо за внимание!