

2017

PISL 14

PISL14. Spring (введение)

Spring  
Framework

Кафедра экономической информатики. Бгуир., 2017

PISL14. Spring (введение)

# SPRING FRAMEWORK

ARCHITECTURE

Кафедра экономической информатики. Бгуир. 2017

2

PISL14. Spring (введение)

- Lightweight jar libraries
- Container
  - Manages lifecycle of objects
- Framework
  - Classes and utilities for application creation
- Dependency Injection (Inversion of Control)
  - Objects get their dependencies and do not create them
- AOP (aspect oriented programming)
  - Separation of cross-cutting concerns into separate modules

Кафедра экономической информатики. Бгуир. 2017

3

PISL14. Spring (введение)

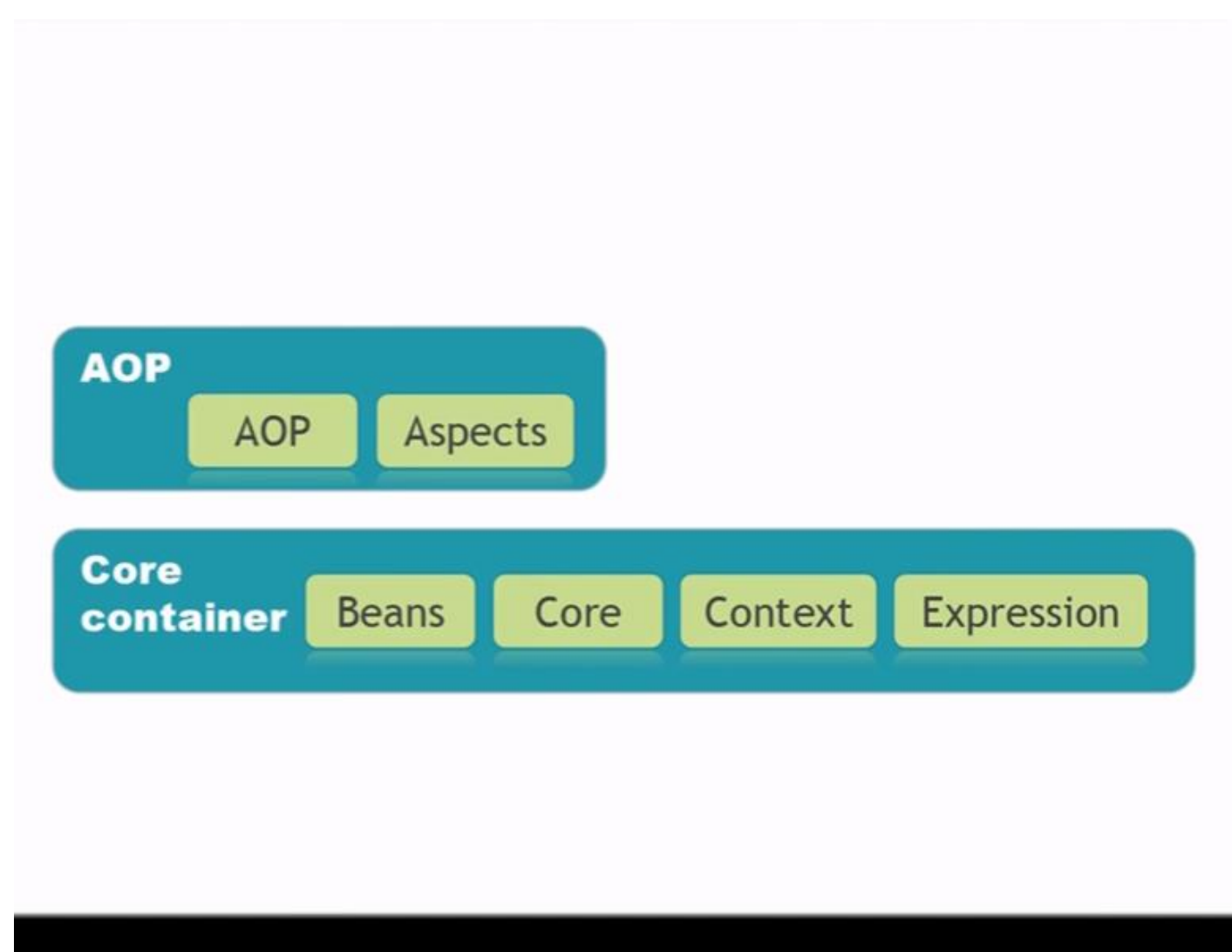
Core  
container

Beans Core Context Expression

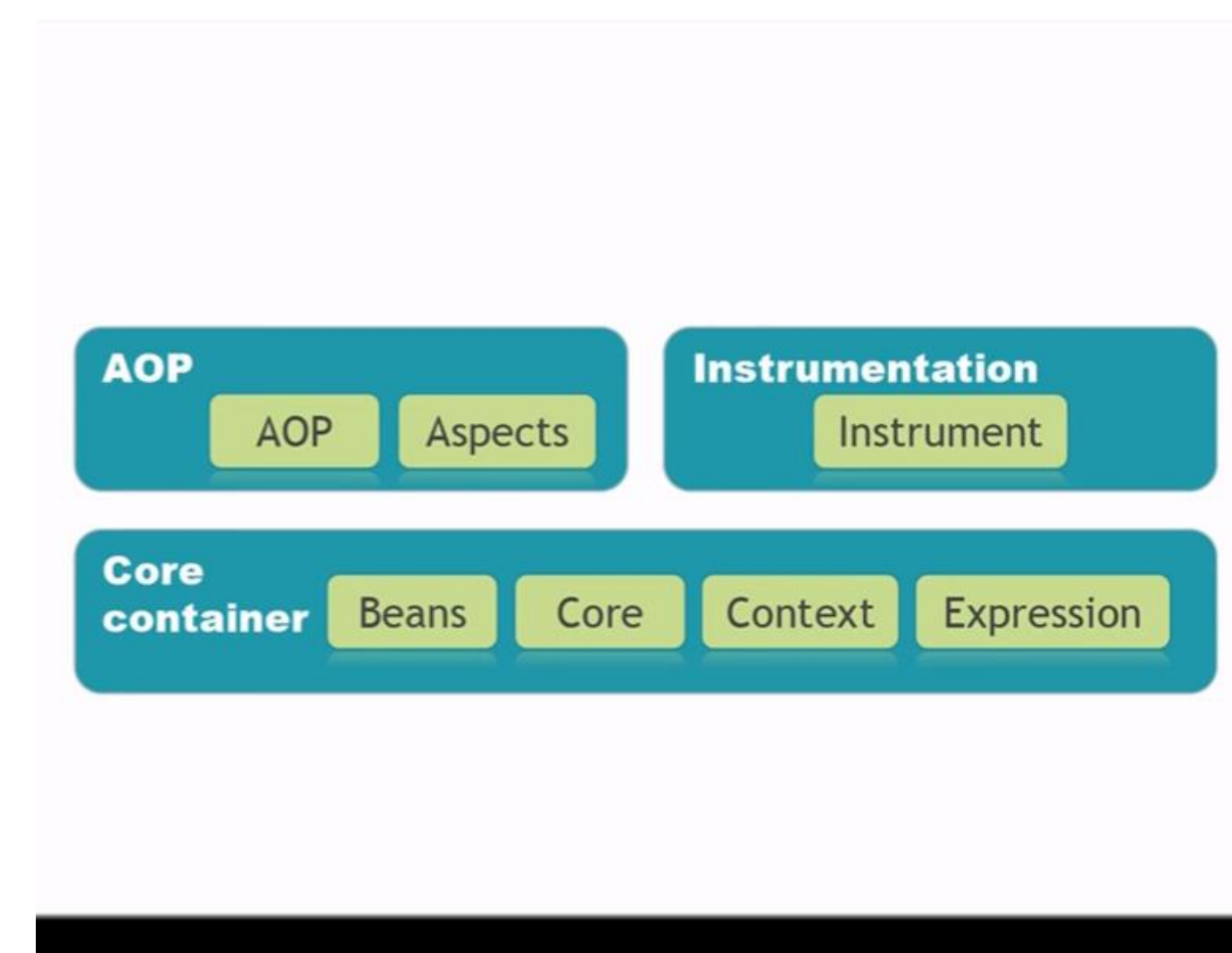
Кафедра экономической информатики. Бгуир. 2017

4

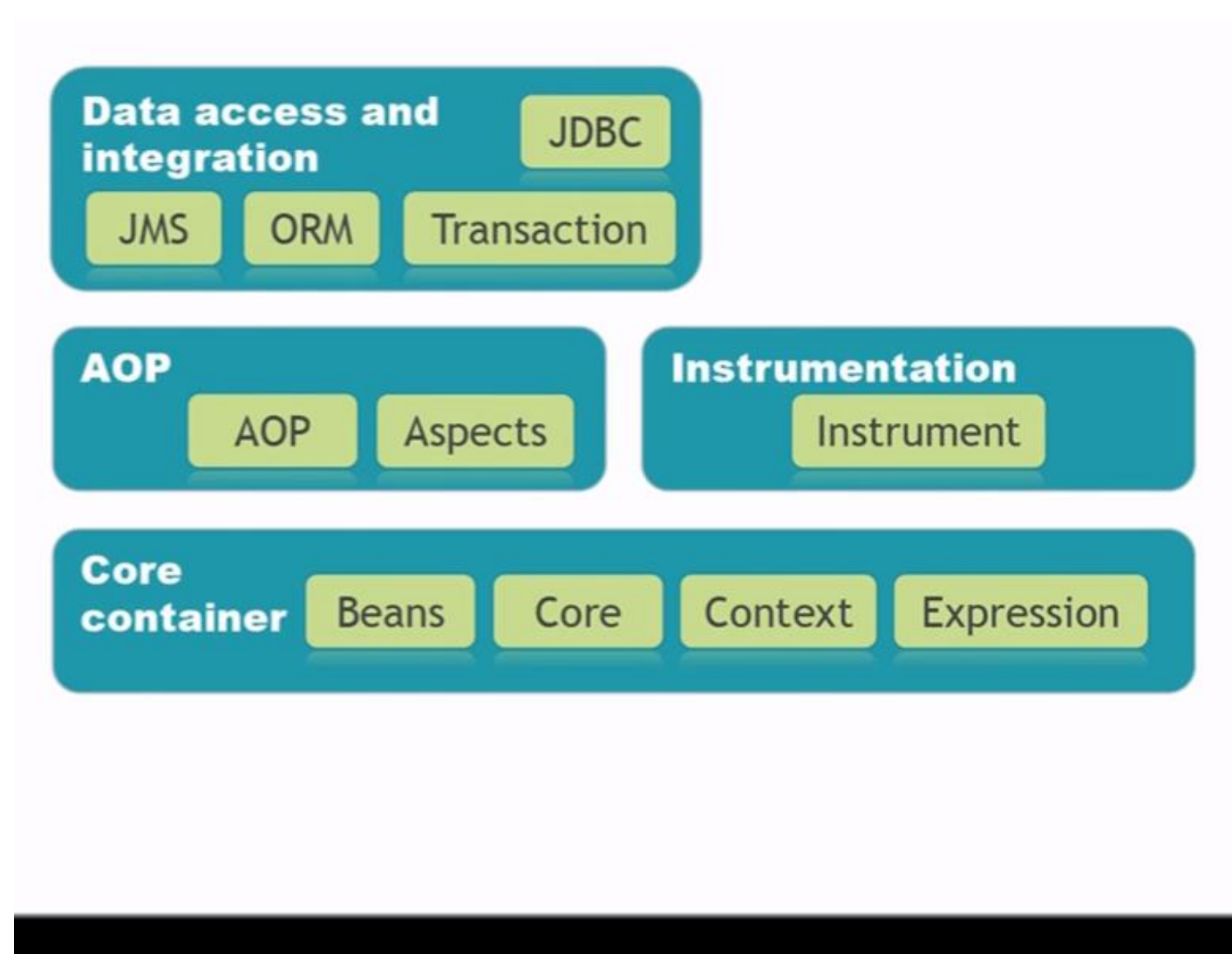
# PISL14. Spring (введение)



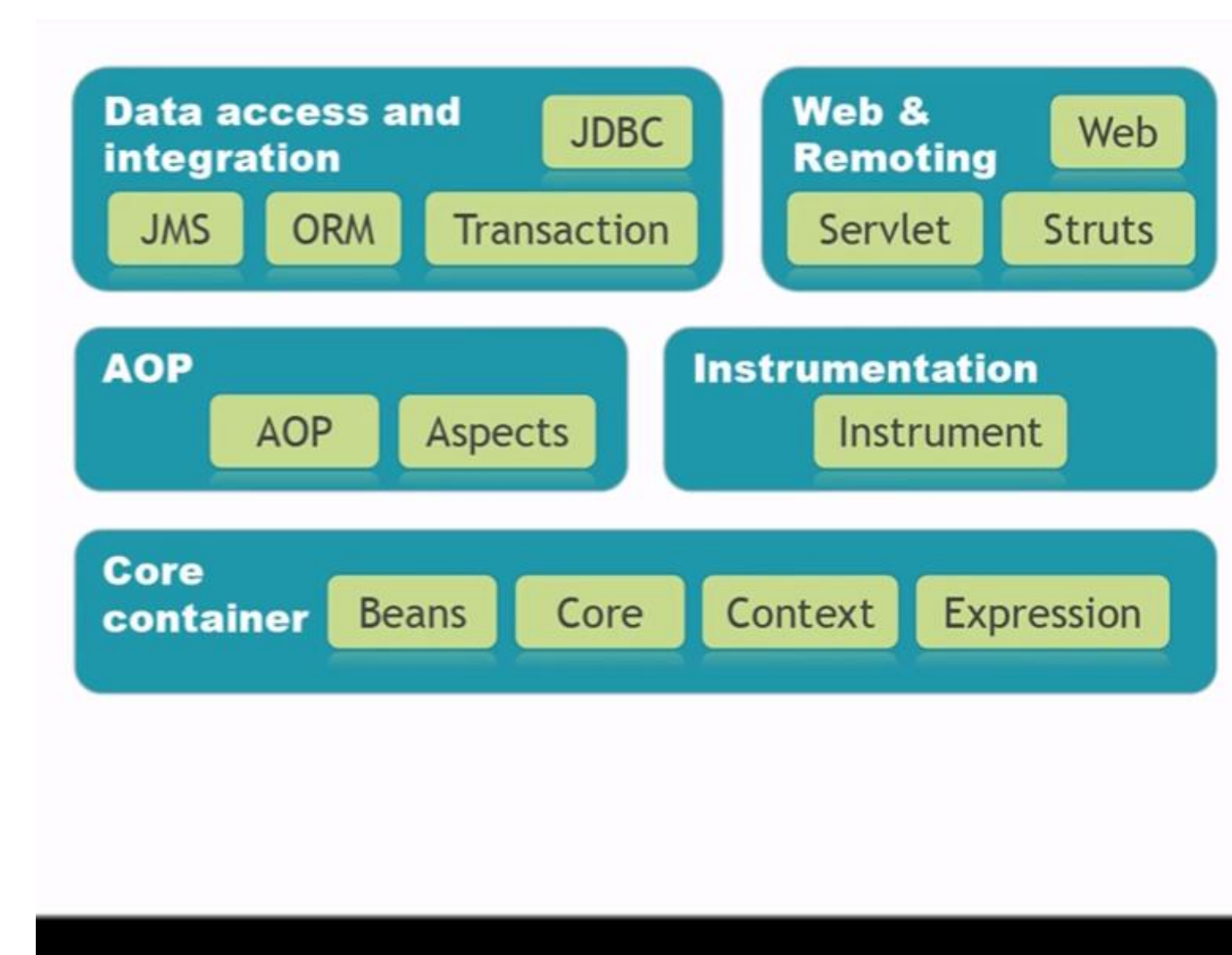
# PISL14. Spring (введение)



# PISL14. Spring (введение)

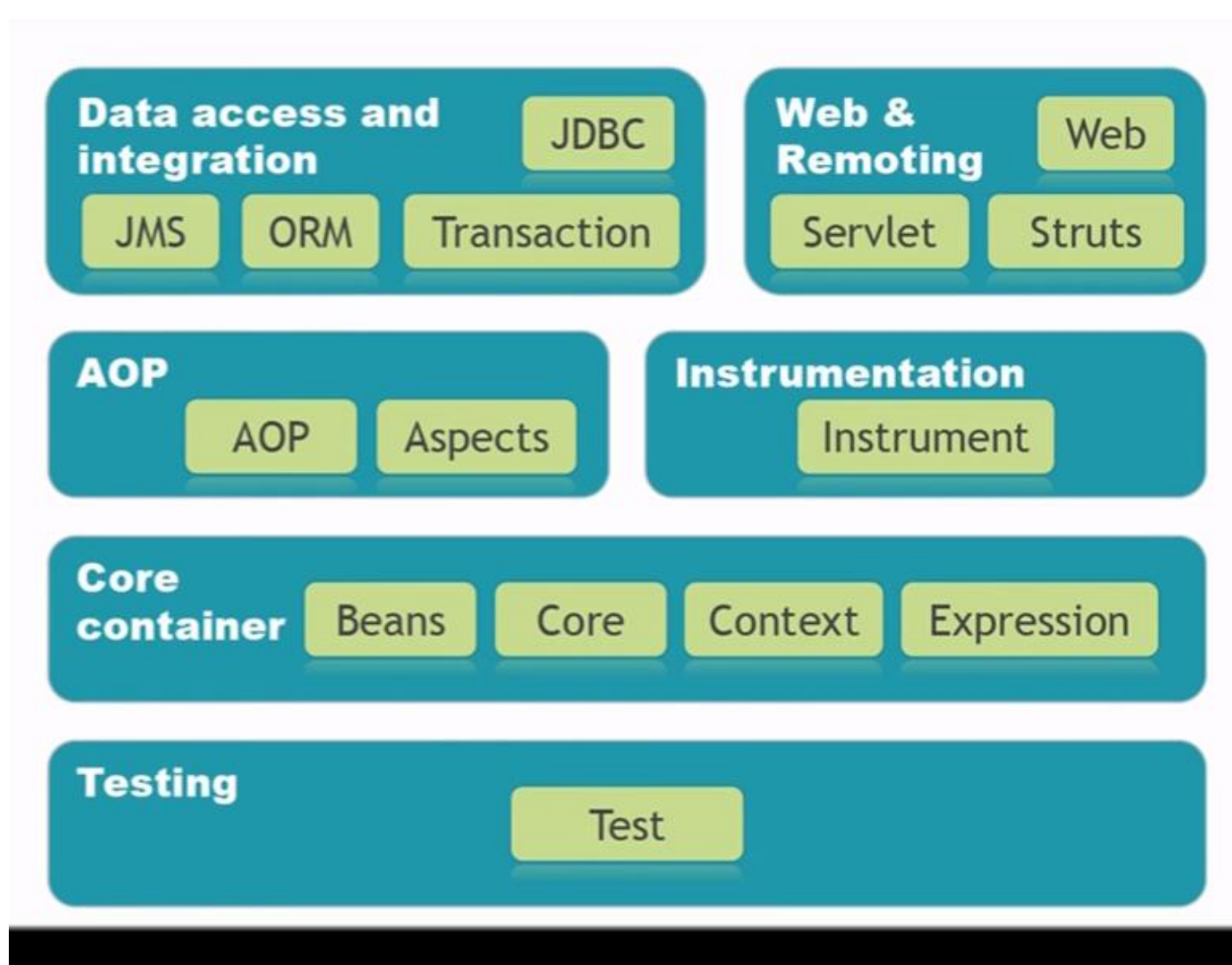


# PISL14. Spring (введение)





## PISL14. Spring (введение)



## PISL14. Spring (введение)

- Create Maven Project
  - `groupId: com.yet.spring`
  - `artifactId: com.yet.spring.core`
- In `pom.xml` add dependencies for Spring modules:
  - `spring-context`
  - `spring-context-support`
  - `spring-tx`
  - `spring-jdbc`

## PISL14. Spring (введение)

```

<properties>
  <spring.ver>3.2.13.RELEASE</spring.ver>
</properties>

<dependencies>
  <dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-context</artifactId>
    <version>${spring.ver}</version>
  </dependency>

  ...
</dependencies>
  
```

spring-context-support  
spring-tx  
spring-jdbc

## PISL14. Spring (введение)

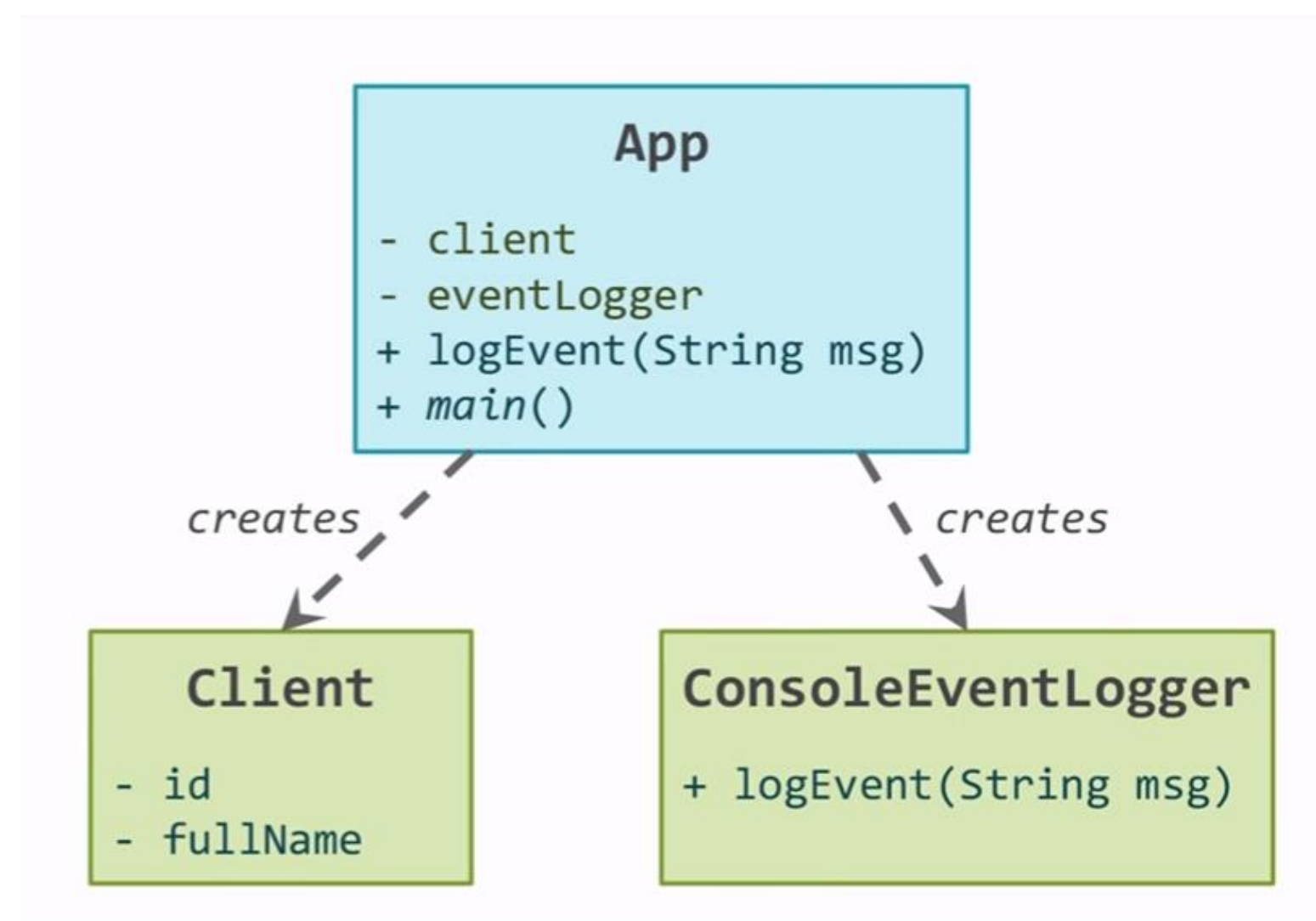
```

<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>com.yet.spring</groupId>
  <artifactId>com.yet.spring.core</artifactId>
  <version>0.1-SNAPSHOT</version>
  <name>Spring Basics Course Project</name>
  <description>The project created throughout the course 'Spring Framework - The Basics'</description>

  <properties>
    <spring.version>3.2.13.RELEASE</spring.version>
    <junit.version>4.12</junit.version>
  </properties>

  <dependencies>
    <dependency>
      <groupId>org.springframework</groupId>
      <artifactId>spring-context</artifactId>
      <version>${spring.version}</version>
    </dependency>
    <dependency>
      <groupId>org.springframework</groupId>
      <artifactId>spring-context-support</artifactId>
      <version>${spring.version}</version>
    </dependency>
    <dependency>
      <groupId>org.springframework</groupId>
      <artifactId>spring-tx</artifactId>
      <version>${spring.version}</version>
    </dependency>
  </dependencies>
</project>
  
```

## PISL14. Spring (введение)



## PISL14. Spring (введение)

```
public static void main(String[] args) {
    App app = new App();

    app.client = new Client("1", "John Smith");
    app.eventLogger = new ConsoleEventListener();

    app.logEvent("Some event for user 1");
}

private void logEvent(String msg) {
    String message = msg.replaceAll(
        client.getId(), client.getFullName());
    eventLogger.logEvent(message);
}
```

## PISL14. Spring (введение)

```
package com.yet.spring.core.beans;

public class ConsoleEventListener {

    public void logEvent(String msg) {
        System.out.println(msg);
    }
}

package com.yet.spring.core.beans;

public class Client {

    private String id;
    private String fullName;

    public Client(String id, String fullName) {
        super();
        this.id = id;
        this.fullName = fullName;
    }

    public String getId() { return id; }

    public void setId(String id) { this.id = id; }

    public String getFullName() { return fullName; }

    public void setFullName(String fullName) { this.fullName = fullName; }
}

package com.yet.spring.core.beans;

public class App {

    private Client client;
    private ConsoleEventListener eventLogger;

    public static void main(String[] args) {
        App app = new App();

        app.client = new Client("1", "John Smith");
        app.eventLogger = new ConsoleEventListener();

        app.logEvent("Some event for user 1");
    }

    private void logEvent(String msg) {
        String message = msg.replaceAll(client.getId(), client.getFullName());
        eventLogger.logEvent(message);
    }
}
```

## PISL14. Spring (введение)

- Modification is problematic
  - Data inside code



## PISL14. Spring (введение)

- Modification is problematic
  - Data inside code
- Scaling is impossible
  - Logger is created in single instance

## PISL14. Spring (введение)

- Modification is problematic
  - Data inside code
- Scaling is impossible
  - Logger is created in single instance
- Testing is hard
  - Unit test for App.logEvent() method indirectly tests ConsoleEventLogger.logEvent()

## PISL14. Spring (введение)

- Modification is problematic
  - Data inside code
- Scaling is impossible
  - Logger is created in single instance
- Testing is hard
  - Unit test for App.logEvent() method indirectly tests ConsoleEventLogger.logEvent()



**HIGH / TIGHT COUPLING**

## PISL14. Spring (введение)

# SPRING FRAMEWORK

DEPENDENCY INJECTION



## PISL14. Spring (введение)

- Externalize static data
  - Use non-binary and non-code files

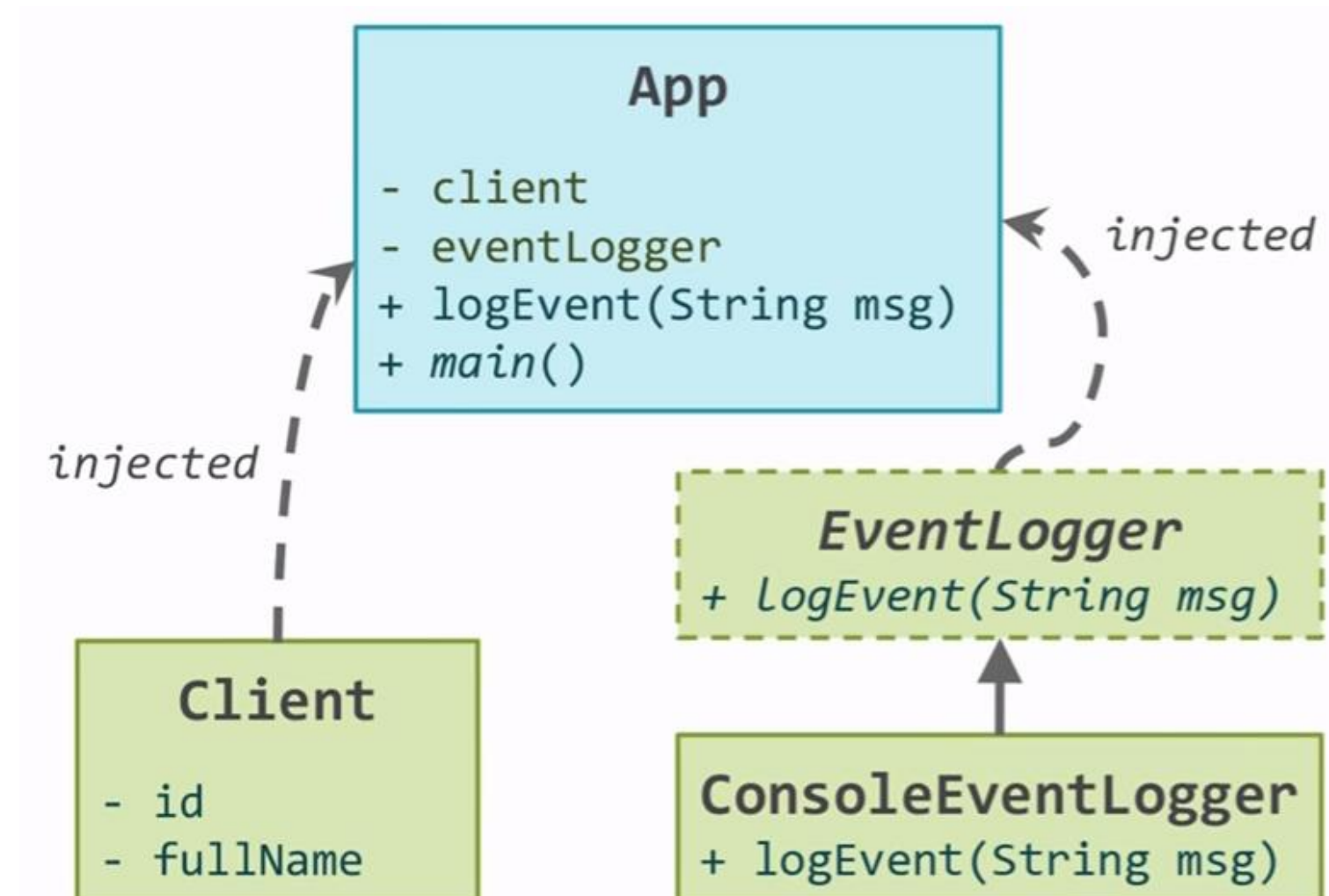
## PISL14. Spring (введение)

- Externalize static data
  - Use non-binary and non-code files
- Decouple with interfaces
  - Use with multiple implementations

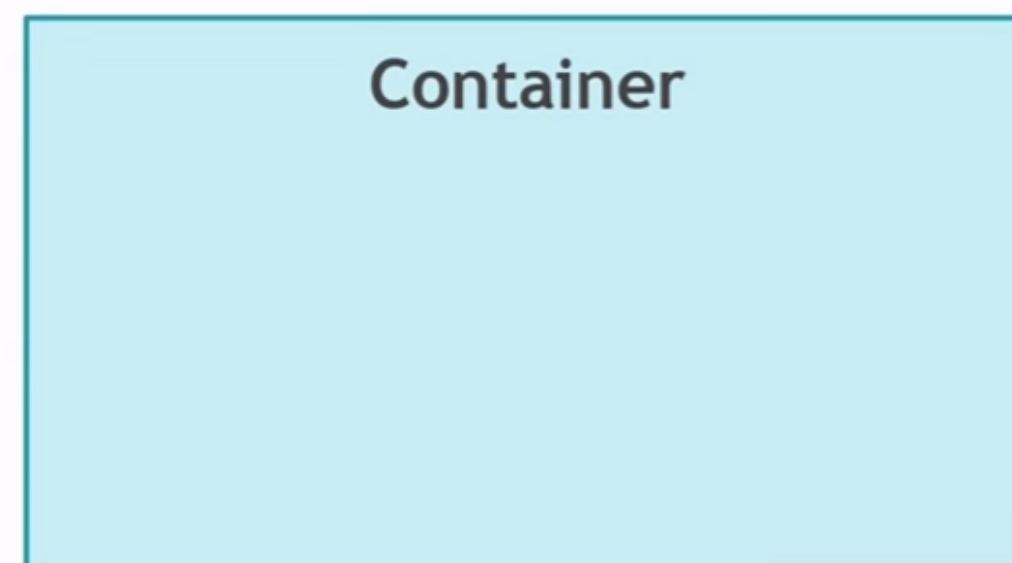
## PISL14. Spring (введение)

- Externalize static data
  - Use non-binary and non-code files
- Decouple with interfaces
  - Use with multiple implementations
- Inject dependencies
  - Let objects receive and not create others

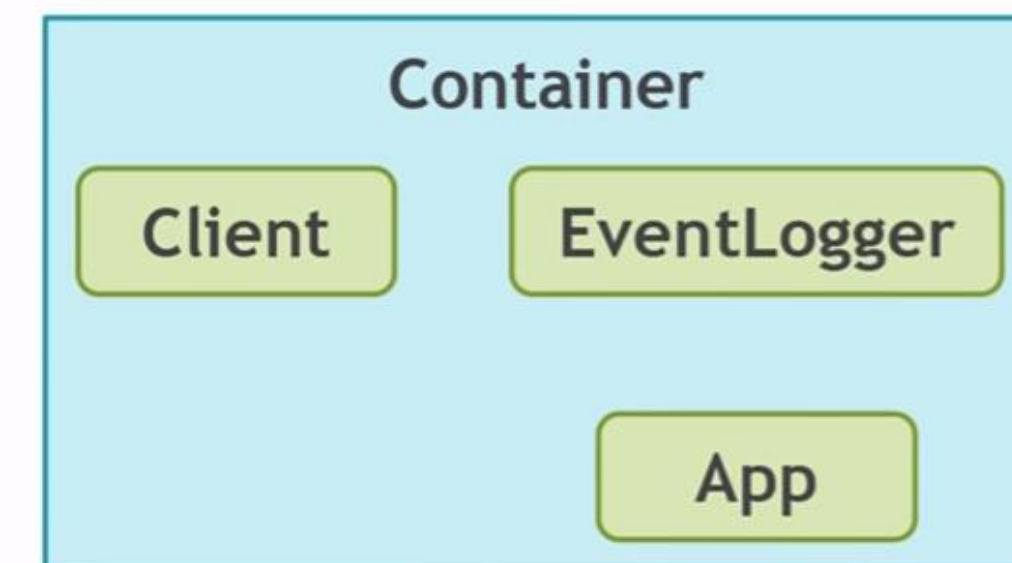
## PISL14. Spring (введение)



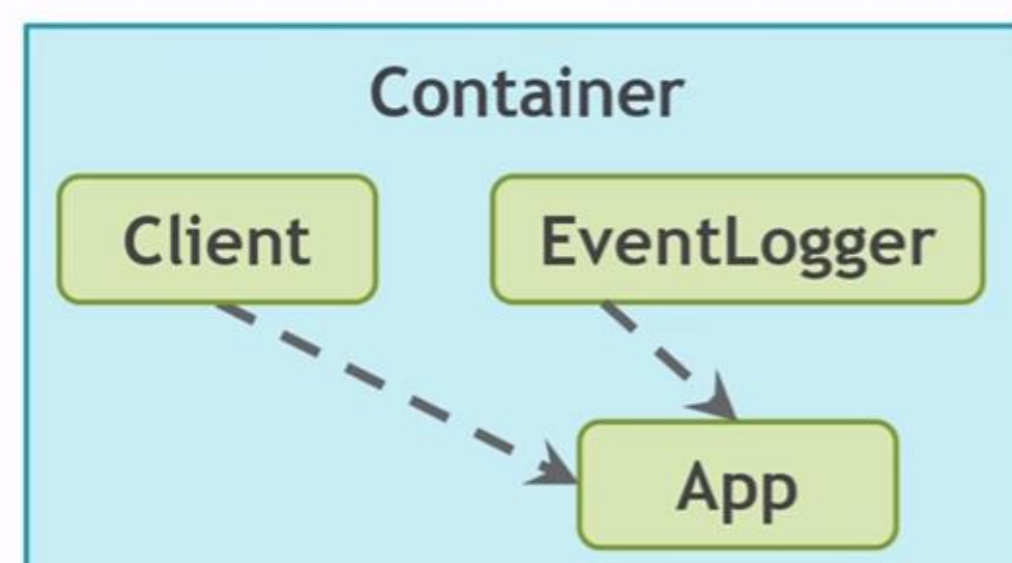
## PISL14. Spring (введение)



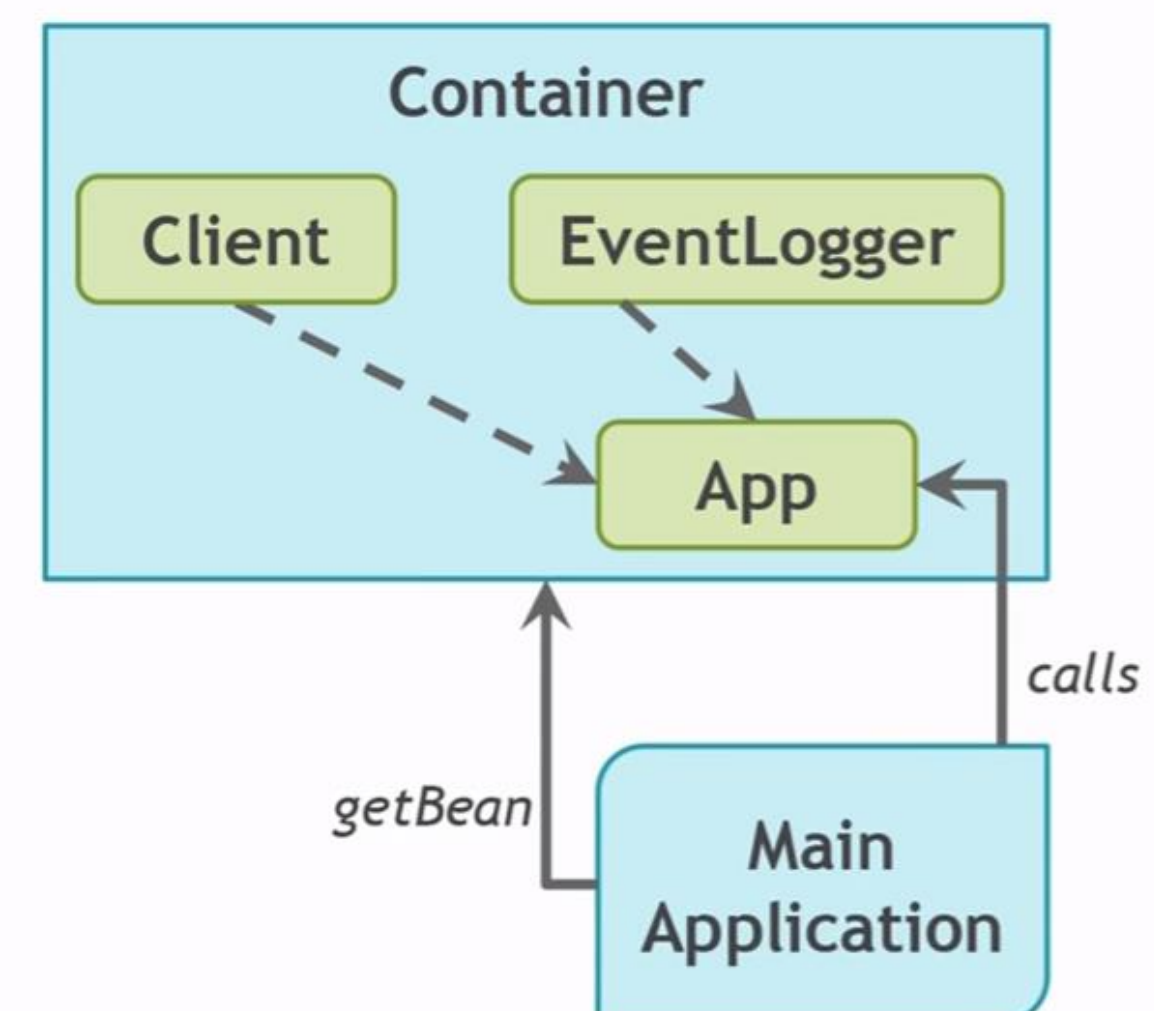
## PISL14. Spring (введение)



## PISL14. Spring (введение)

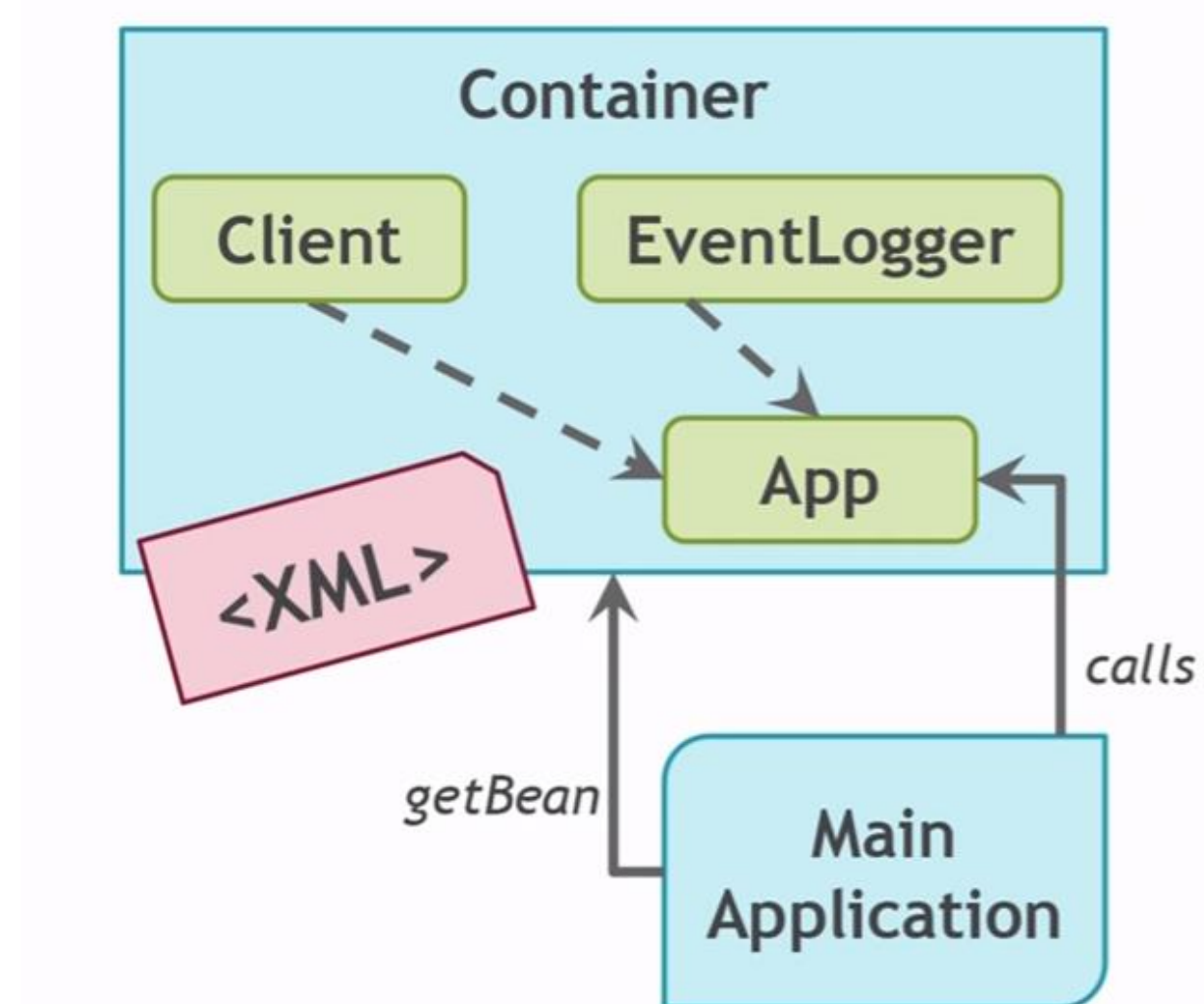


## PISL14. Spring (введение)





## PISL14. Spring (введение)



## PISL14. Spring (введение)

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns=http://www.springframework.org/schema/beans
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="
    http://www.springframework.org/schema/beans
    http://www.springframework.org/schema/beans/spring-beans-3.2.xsd">

  <bean id="client" class="com.yet.spring.core.beans.Client"/>

  <bean id="eventLogger"
    class="com.yet.spring.core.loggers.ConsoleEventLogger"/>

  <bean id="app" class="com.yet.spring.core.App"/>

</beans>
```

## PISL14. Spring (введение)

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns=http://www.springframework.org/schema/beans
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="
    http://www.springframework.org/schema/beans
    http://www.springframework.org/schema/beans/spring-beans-3.2.xsd">

  <bean id="client" class="com.yet.spring.core.beans.Client"/>

  <bean id="eventLogger"
    class="com.yet.spring.core.loggers.ConsoleEventLogger"/>

  <bean id="app" class="com.yet.spring.core.App"/>

</beans>
```

## PISL14. Spring (введение)

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns=http://www.springframework.org/schema/beans
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="
    http://www.springframework.org/schema/beans
    http://www.springframework.org/schema/beans/spring-beans-3.2.xsd">

  <bean id="client" class="com.yet.spring.core.beans.Client"/>

  <bean id="eventLogger"
    class="com.yet.spring.core.loggers.ConsoleEventLogger"/>

  <bean id="app" class="com.yet.spring.core.App"/>

</beans>
```



## PISL14. Spring (введение)

# SPRING FRAMEWORK

### BEAN NAMING & CONTEXT START-UP



## PISL14. Spring (введение)

`<bean id="..."` или `<bean name="..."`

- **id** - формальный XML атрибут, до версии Spring 3 допускал только валидные символы. Может быть только одно значение.
- **name** - допускает любые символы, может быть несколько значений через запятую/пробел/; (алиасы бина)

Может присутствовать id вместе с name

## PISL14. Spring (введение)

### ALIAS

```
<bean id="aaa" name="xxx,zzz" .. />
```

```
<alias name="aaa" alias="bbb"/>
```

## PISL14. Spring (введение)

### SIMPLE VALUES

```
<bean id="client"
    class="com.yet.spring.core.beans.Client">
    <constructor-arg value="1"/>
    <constructor-arg value="John Smith"/>
</bean>
```

### WHEN ORDER MATTERS

```
<constructor-arg index="0" value="1"/>
<constructor-arg index="1" value="J. Smith"/>
```

## PISL14. Spring (введение)

### WHEN TYPE MATTERS

```
<constructor-arg type="java.lang.Integer"
    value="1"/>

<constructor-arg type="java.lang.String"
    value="1"/>
```

### NAMES CAN BE USED *(Debug symbols are needed!)*

```
<constructor-arg name="arg1" value="1"/>
<constructor-arg name="arg2" value="Smith"/>
```

## PISL14. Spring (введение)

### OTHER BEANS

```
<bean id="app" class="com.yet.spring.core.App"/>

    <constructor-arg ref="client"/>
    <constructor-arg ref="eventLogger"/>

</bean>
```

## PISL14. Spring (введение)

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <beans xmlns="http://www.springframework.org/schema/beans"
3     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4     xsi:schemaLocation="
5         http://www.springframework.org/schema/beans
6         http://www.springframework.org/schema/beans/spring-beans-3.2.xsd">
7
8     <bean id="client" class="com.yet.spring.core.beans.Client">
9         <constructor-arg index="0" value="1" />
10        <constructor-arg value="John Smith" />
11    </bean>
12
13    <bean id="eventLogger" class="com.yet.spring.core.loggers.ConsoleEventLogger" />
14
15    <bean id="app" class="com.yet.spring.core.App">
16        <constructor-arg ref="client" />
17        <constructor-arg ref="eventLogger" />
18    </bean>
19
20 </beans>
```

## PISL14. Spring (введение)

- **BeanFactory**  
Simple container that supports only DI
- **ApplicationContext**  
Supports DI + framework services
  - ClassPathXmlApplicationContext
  - FileSystemXmlApplicationContext
  - AnnotationConfigApplicationContext
  - XmlWebApplicationContext
  - StaticApplicationContext



# PISL14. Spring (введение)

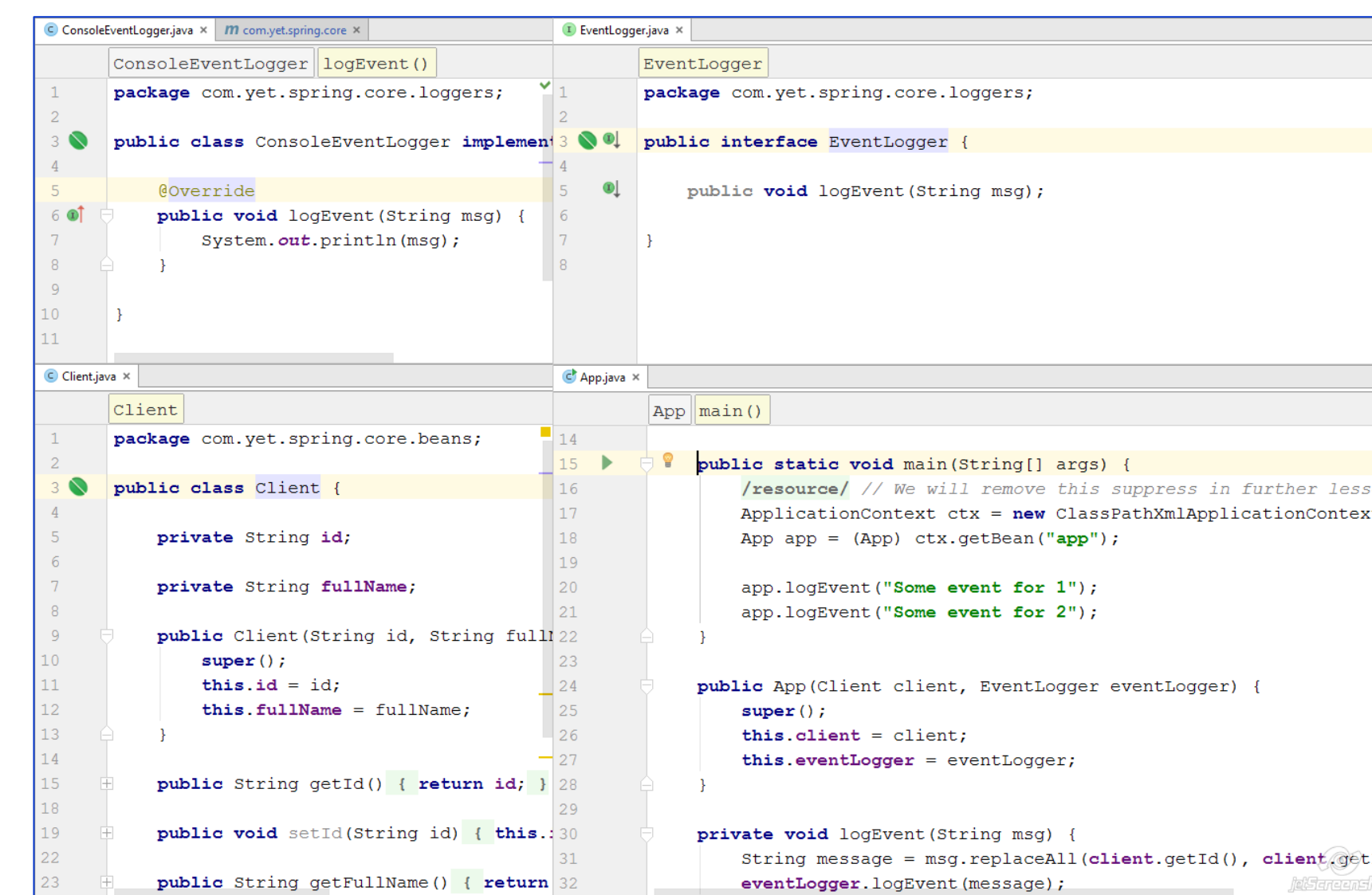
```
public static void main(String[] args) {

    ApplicationContext ctx =
        new ClassPathXmlApplicationContext(
            "spring.xml");

    App app = (App) ctx.getBean("app");

    app.logEvent("Some event for 1");
    app.logEvent("Some event for 2");
}
```

# PISL14. Spring (введение)



# PISL14. Spring (введение)

## SPRING FRAMEWORK

### SCOPES AND INNER BEANS



# PISL14. Spring (введение)

**Event**

- id: int; // Auto-generated
- msg: String; // Set in setter
- date: Date; // Set in constructor
- + toString()

## PISL14. Spring (введение)

```

Event
- id: int;      // Auto-generated
- msg: String;  // Set in setter
- date: Date;   // Set in constructor
+ toString()

```

```

EventLogger
+ LogEvent(Event event)

```

## PISL14. Spring (введение)

```

Event
- id: int;      // Auto-generated
- msg: String;  // Set in setter
- date: Date;   // Set in constructor
+ toString()

```

```

EventLogger
+ LogEvent(Event event)

```

Class **App** now creates **Event** objects.  
Or, maybe, it should receive them? :)

## PISL14. Spring (введение)

```

7 public class Event {
8
9     private static final AtomicInteger AUTO_ID = new AtomicInteger(0);
10
11     private int id;
12     private String msg;
13     private Date date;
14
15     private DateFormat dateFormat;
16
17     public Event(Date date, DateFormat df) {
18         this.id = AUTO_ID.getAndIncrement();
19
20         this.date = date;
21         this.dateFormat = df;
22     }
23
24     public String getMsg() { return msg; }
27
28     public void setMsg(String msg) { this.msg = msg; }
31
32     public int getId() { return id; }
35

```

## PISL14. Spring (введение)

### BEAN SCOPE

By default all beans are singletons (within container)  
`<bean id="..." class="...">`  
`<bean id="..." class="..." scope="singleton">`

Prototype - new object on every getBean() call  
`<bean id="..." class="..." scope="prototype">`

For web applications:

- request
- session
- global-session



## PISL14. Spring (введение)

**By default** all beans are singletons (within container)

```
<bean id="..." class="...">
<bean id="..." class="..." scope="singleton">
```

Prototype - new object on every getBean() call

```
<bean id="..." class="..." scope="prototype">
```

For web applications:

- request
- session
- global-session

## PISL14. Spring (введение)

**By default** all beans are singletons (within container)

```
<bean id="..." class="...">
<bean id="..." class="..." scope="singleton">
```

Prototype - new object on every getBean() call

```
<bean id="..." class="..." scope="prototype">
```

For web applications:

- request
- session
- global-session

Make Event  
a prototype

## PISL14. Spring (введение)

```
<bean id="event" class="..." scope="prototype">
```

```
<constructor-arg>
```

```
<bean class="java.util.Date"/>
```

```
</constructor-arg>
```

```
</bean>
```

- Visible where they are defined - can't be reused

## PISL14. Spring (введение)

```
public Event(Date date, DateFormat df) {
    this.date = date; this.df = df;
}

public String toString() {
    ... df.format(date) ...
}
```

## PISL14. Spring (введение)

```
public Event(Date date, DateFormat df) {  
    this.date = date; this.df = df;  
}  
  
public String toString() {  
    ... df.format(date) ...  
}
```

DateFormat -  
abstract class

```
DateFormat.getDateInstance()  
DateFormat.getTimeInstance()  
DateFormat.getDateTimeInstance()
```

## PISL14. Spring (введение)

```
public Event(Date date, DateFormat df) {  
    this.date = date; this.df = df;  
}  
  
public String toString() {  
    ... df.format(date) ...  
}
```

DateFormat -  
abstract class

```
DateFormat.getDateInstance()  
DateFormat.getTimeInstance()  
DateFormat.getDateTimeInstance()
```

```
<bean id="dateFormat"  
      class="java.text.DateFormat"  
      factory-method="getDateTimeInstance"/>
```

## PISL14. Spring (введение)

```
<bean id="event" class="com.yet.spring.core.beans.Event" scope="prototype">  
    <constructor-arg>  
        <bean class="java.util.Date"/>  
    </constructor-arg>  
    <constructor-arg ref="dateFormat"/>  
</bean>  
  
<bean id="dateFormat" class="java.text.DateFormat"  
      factory-method="getDateTimeInstance"/>  
  
</beans>
```

## PISL14. Spring (введение)

# SPRING FRAMEWORK

INITIALIZE & DESTROY





## PISL14. Spring (введение)

### FileEventLogger

```
- filename: String;
+ logEvent(...); // appends to file
```

## PISL14. Spring (введение)

### FileEventLogger

```
- filename: String;
+ logEvent(...); // appends to file
```

Use FileUtils from commons-io:

- `<groupId>commons-io</groupId>`  
`<artifactId>commons-io</artifactId>`  
`<version>2.4</version>`
- `FileUtils.writeStringToFile(File file, String str, boolean append)`

## PISL14. Spring (введение)

- Actions for initializing bean's state after creation
  - For example, add check for file write access and create file
- Initialize beans in constructor is not always possible
  - Bean can get its dependencies after creation

## PISL14. Spring (введение)

- FileEventLogger:
 

```
public void init() throws IOException {
    this.file = new File(fileName);
    // check file write access
}
```

## PISL14. Spring (введение)

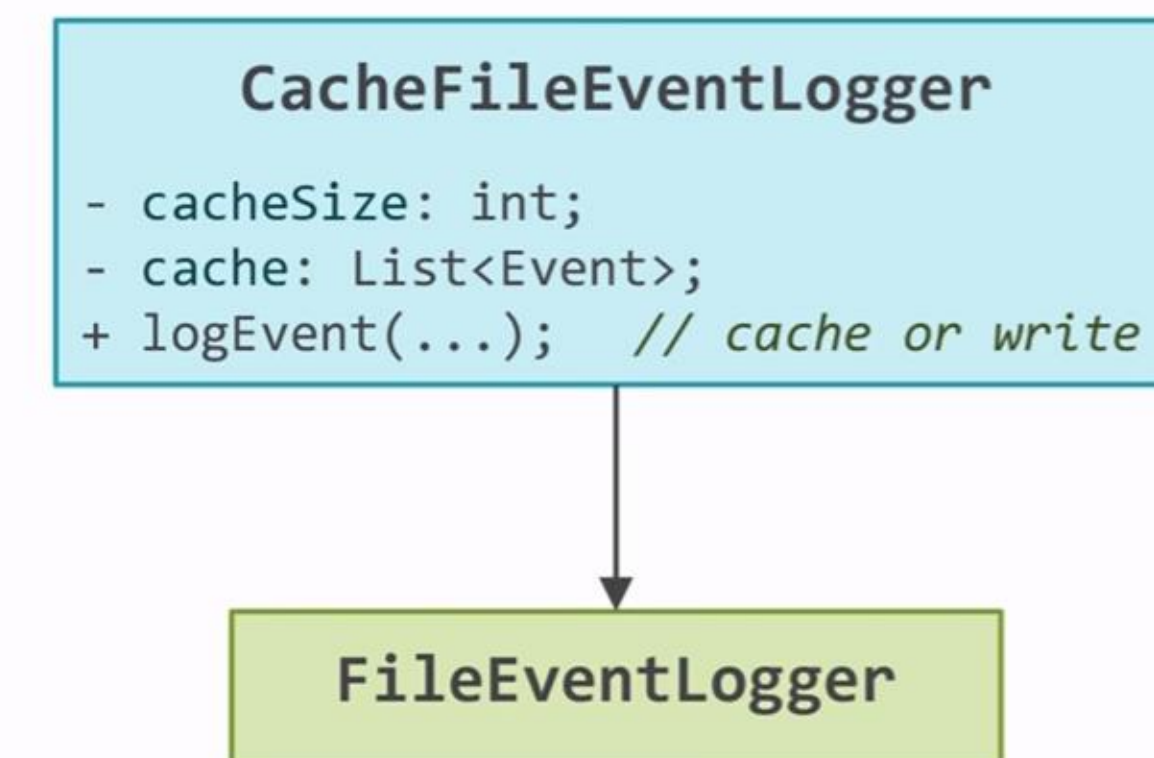
- FileEventListener:
 

```
public void init() throws IOException {
    this.file = new File(fileName);
    // check file write access
}
```
- spring.xml
 

```
<bean id="..." class="..."
      init-method="init">
```

  - no arguments
  - any access modifier
  - can return something or throw Exception

## PISL14. Spring (введение)



## PISL14. Spring (введение)

```

CacheFileEventListener
- cacheSize: int;
- cache: List<Event>;
+ logEvent(...); // cache or write

public void logEvent(Event event) {
    cache.add(event);

    if (cache.size() == cacheSize) {
        writeEventsFromCache();
        cache.clear();
    }
}
  
```

## PISL14. Spring (введение)

- Being called when context is closing
- CacheFileEventListener:
 

```
public void destroy() {
    if ( !cache.isEmpty() )
        writeEventsFromCache();
}
```
- spring.xml
 

```
<bean id="..." class="..."
      destroy-method="destroy">
```
- App class
 

```
ConfigurableApplicationContext ctx = ...
ctx.close();
```



## PISL14. Spring (введение)

```

1 package com.yet.spring.core.loggers;
2
3 import com.yet.spring.core.beans.Event;
4
5 public class ConsoleEventListener implements EventLogger {
6
7     @Override
8     public void logEvent(Event event) {
9         System.out.println(event.toString());
10    }
11}
12
13 FileEventListener.java
14 package com.yet.spring.core.loggers;
15
16 import ...
17
18 public class FileEventListener implements EventLogger {
19
20     private File file;
21     private String filename;
22
23     public FileEventListener(String filename) { this.filename = filename; }
24
25     public void init() {
26         file = new File(filename);
27         if (file.exists() && !file.canWrite()) {
28             throw new IllegalArgumentException("Can't write to file: " + filename);
29         }
30     }
31
32     @Override
33     public void logEvent(Event event) {
34         System.out.println(event.toString());
35     }
36 }
37
38 CacheFileEventListener.java
39 package com.yet.spring.core.loggers;
40
41 import com.yet.spring.core.beans.Event;
42
43 public class CacheFileEventListener extends FileEventListener {
44
45     private int cacheSize;
46     private List<Event> cache;
47
48     public CacheFileEventListener(String filename, int cacheSize) {
49         super(filename);
50         this.cacheSize = cacheSize;
51         this.cache = new ArrayList<Event>(cacheSize);
52     }
53
54     public void destroy() {
55         if (!cache.isEmpty()) {
56             writeEventsFromCache();
57         }
58     }
59
60     @Override
61     public void logEvent(Event event) {
62         cache.add(event);
63
64         if (cache.size() == cacheSize) {
65             writeEventsFromCache();
66             cache.clear();
67         }
68     }
69
70     private void writeEventsFromCache() {
71         // ...
72     }
73 }

```

Кафедра экономической информатики. Бгуир. 2017

65

## PISL14. Spring (введение)

# SPRING FRAMEWORK

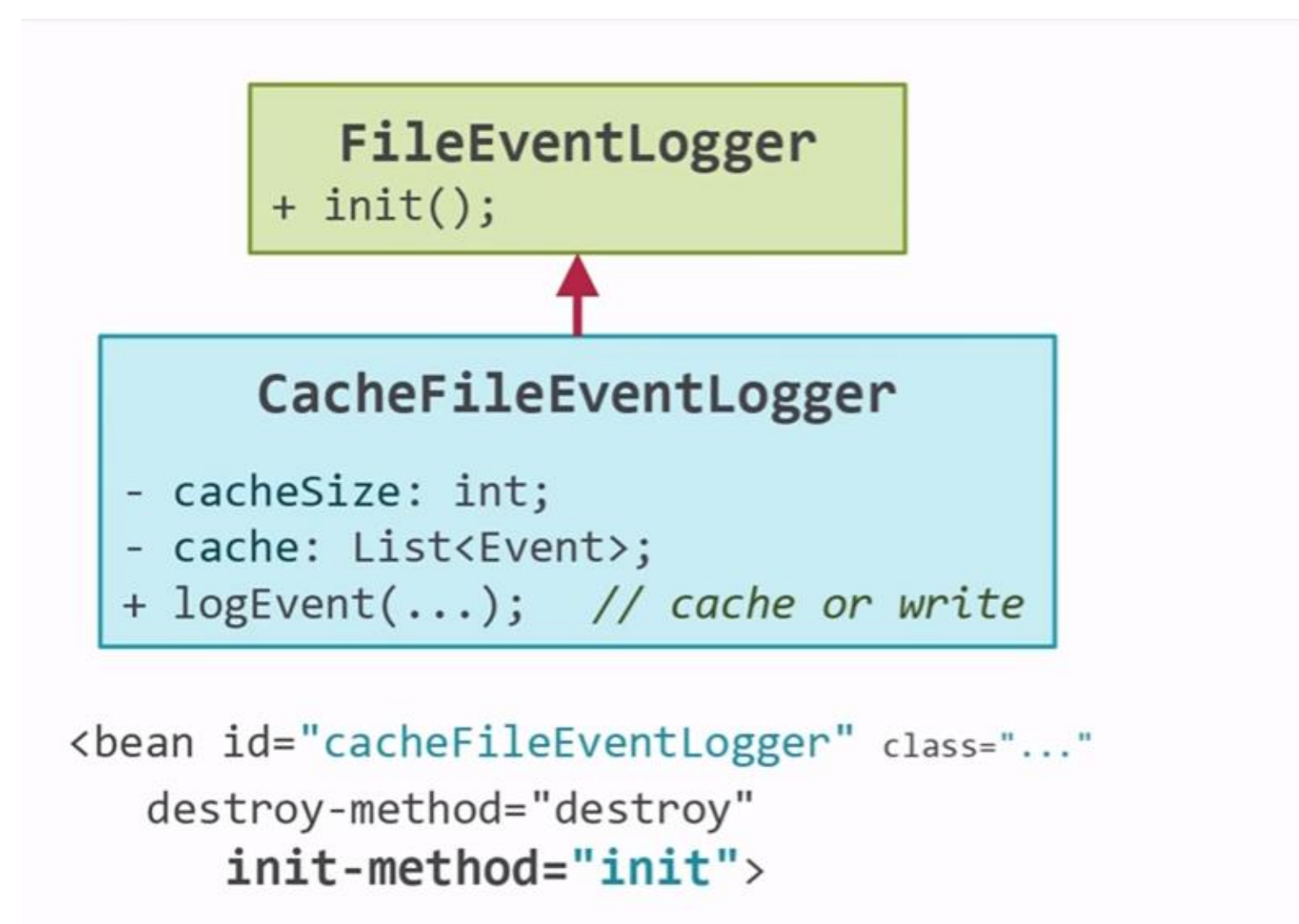
CONTEXT FEATURES

JSERVERS

Кафедра экономической информатики. Бгуир. 2017

66

## PISL14. Spring (введение)



Кафедра экономической информатики. Бгуир. 2017

67

## PISL14. Spring (введение)

- **Parent:**

```

<bean id="fileLogger" class="..."
    <constructor-arg value="log.txt"/>
</bean>

```
- **Child:**

```

<bean id="cacheFileLogger" class="..."
    parent="fileLogger"
    <constructor-arg value="10"/>
</bean>

```
- **Abstract:**

```

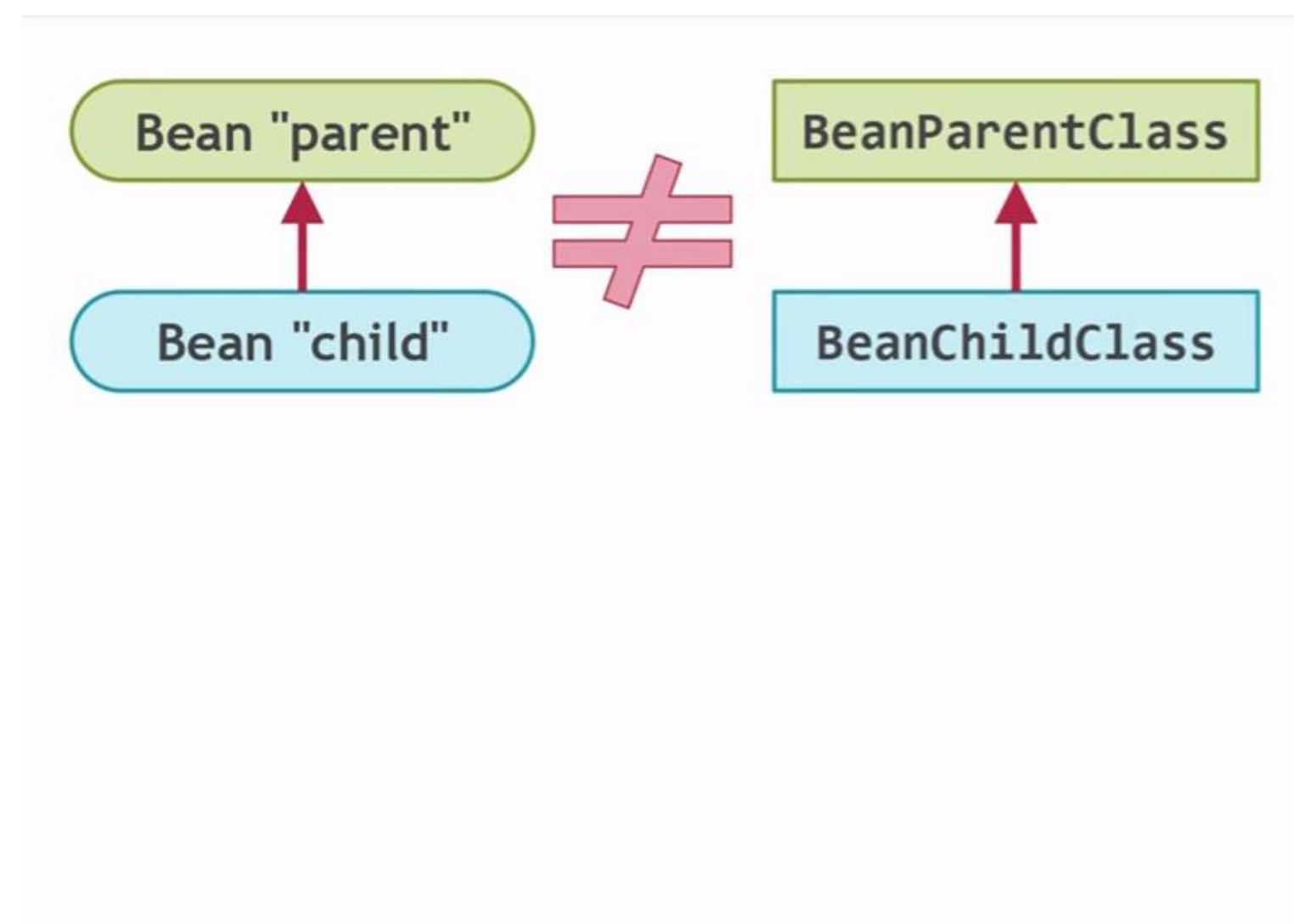
<bean id="..." abstract="true">

```

Кафедра экономической информатики. Бгуир. 2017

68

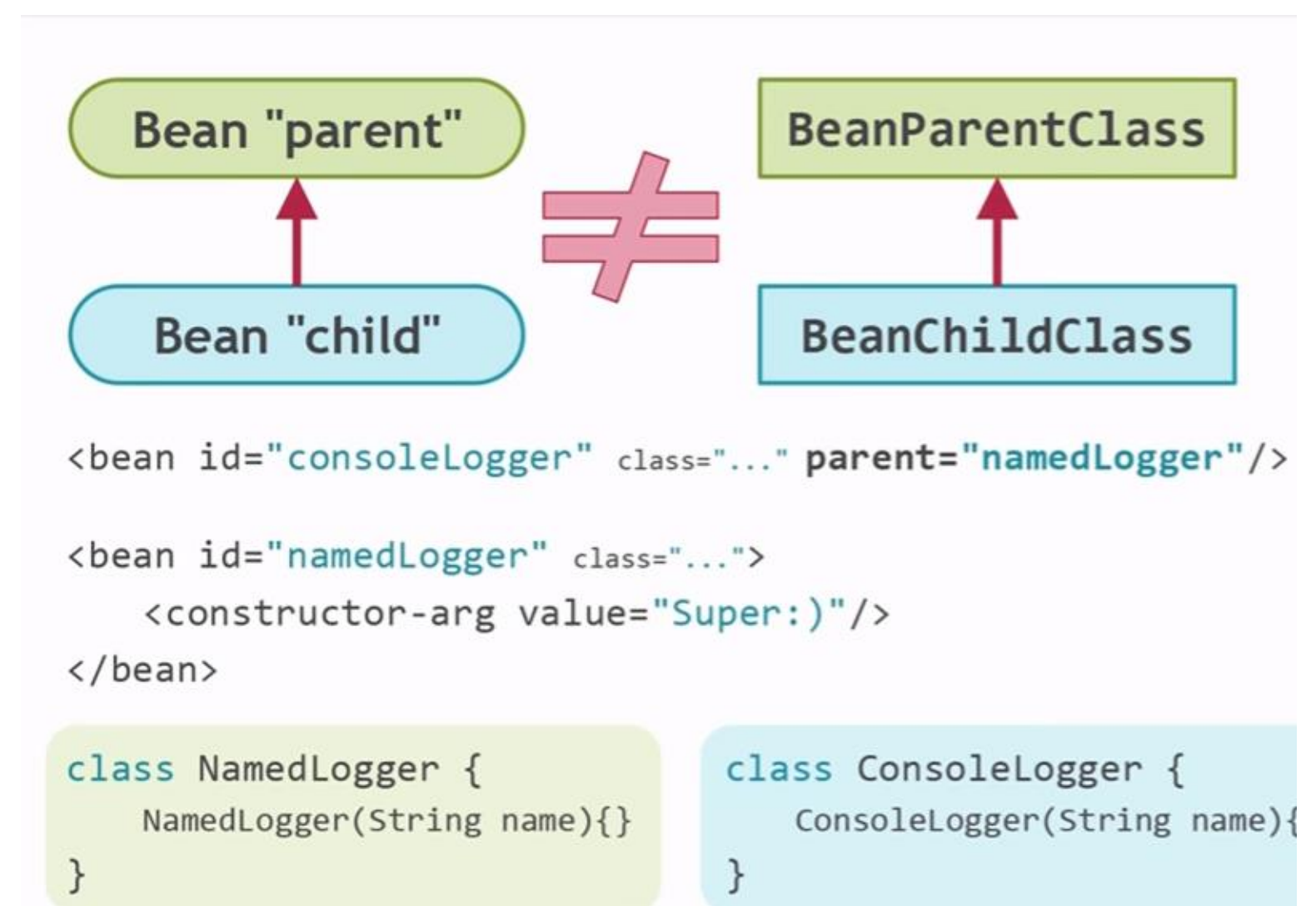
## PISL14. Spring (введение)



Кафедра экономической информатики. Бгуир. 2017

69

## PISL14. Spring (введение)



Кафедра экономической информатики. Бгуир. 2017

70

## PISL14. Spring (введение)

### BEAN DEPENDENCY

```
<bean id="monitoring" class="..."
      depends-on="app"/>
<bean id="app" class="..." />
```

Кафедра экономической информатики. Бгуир. 2017

71

## PISL14. Spring (введение)

### BEAN DEPENDENCY

```
<bean id="monitoring" class="..."
      depends-on="app"/>
<bean id="app" class="..." />
```

### LAZY INITIALIZATION

```
<bean id="lazy" class="..."
      lazy-init="true"/>
```

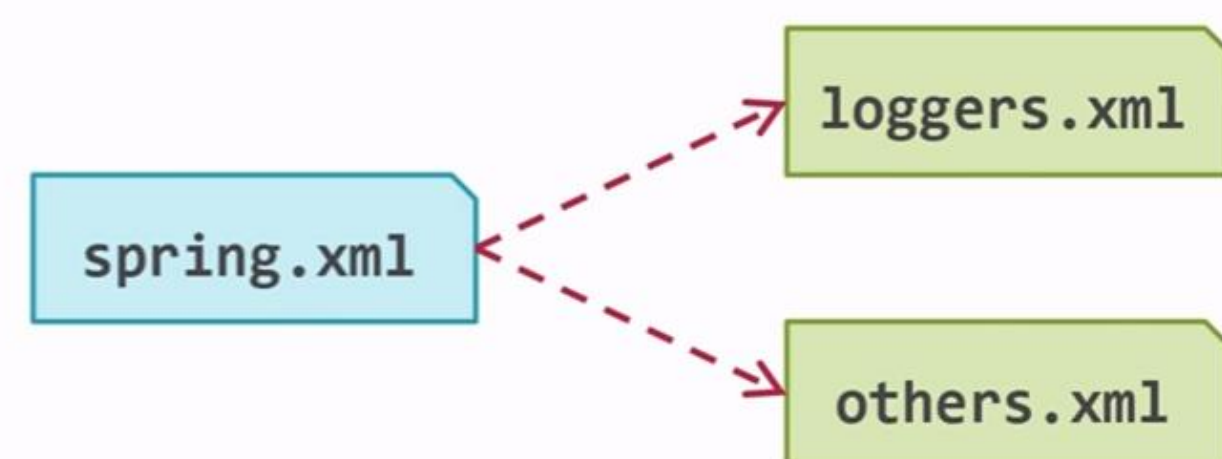
- ```
<beans default-lazy-init="true">
  <!-- no beans will be pre-instantiated... -->
</beans>
```

Кафедра экономической информатики. Бгуир. 2017

72



## PISL14. Spring (введение)



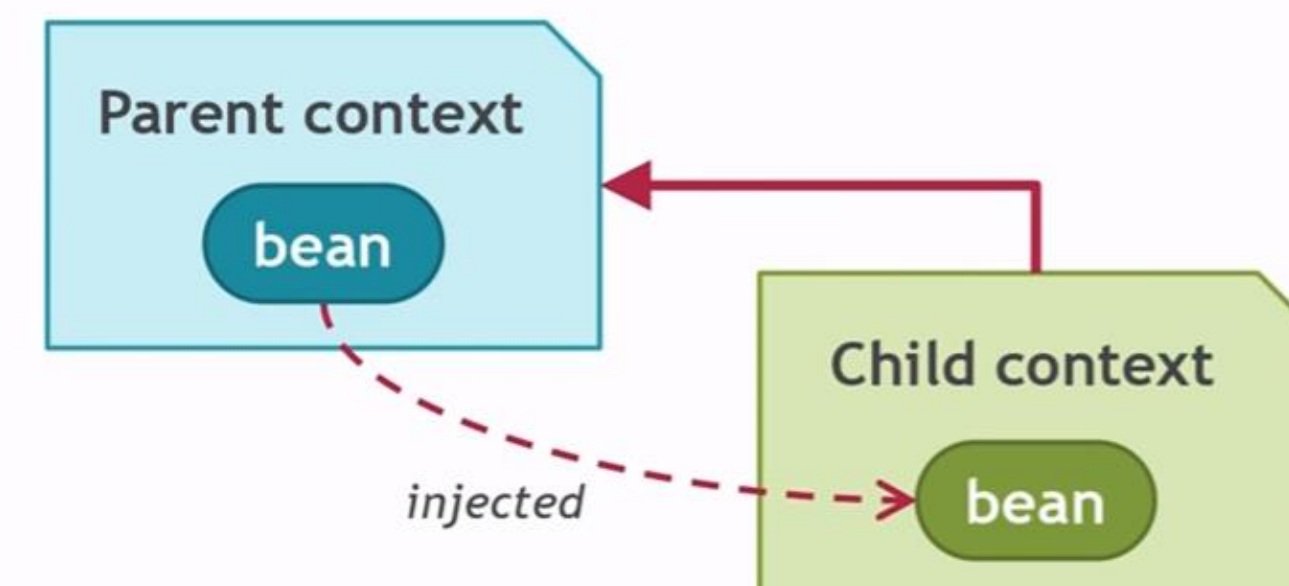
```

<beans ...>
  <import resource="loggers.xml"/>
</beans>
  
```

Кафедра экономической информатики. Бгуир. 2017

73

## PISL14. Spring (введение)



```

ApplicationContext parent =
  new ClassPathXmlApplicationContext(xmls);

ApplicationContext child =
  new ClassPathXmlApplicationContext(xmls, parent);
  
```

Кафедра экономической информатики. Бгуир. 2017

74

## PISL14. Spring (введение)

# SPRING FRAMEWORK

PROPERTY INJECTION

JavaScreencast

Кафедра экономической информатики. Бгуир. 2017

75

## PISL14. Spring (введение)

```

<bean id="client" class="...">
  <property name="greeting"
    value="Hello there!" />
</bean>

class Client {
  public void setGreeting(String gr) {...}
}
  
```

Кафедра экономической информатики. Бгуир. 2017

76

## PISL14. Spring (введение)

```
<property name="..." ref="bean" />

<property name="...">
  <bean class="..." /> -- inner-bean
</property>
```

Кафедра экономической информатики. Бгуир. 2017

77

## PISL14. Spring (введение)

```
<property name="..." ref="bean" />

<property name="...">
  <bean class="..." /> -- inner-bean
</property>
```

What would be the value of field str?

```
<bean>
  <constructor-arg value="aa" />
  <property name="str" value="bb" />
</bean>
```

Кафедра экономической информатики. Бгуир. 2017

78

## PISL14. Spring (введение)

### EventType

INFO,  
ERROR;

- *default* - use CacheFileEventListener
- INFO - use ConsoleEventListener
- ERROR - use all loggers  
(ConsoleEventListener & FileEventListener)

### CombinedEventListener

- loggers: Collection;

1 → \* *EventListener*

Кафедра экономической информатики. Бгуир. 2017

79

## PISL14. Spring (введение)

### LIST or ARRAY

```
<list>
  <value>...<value>      -- simple values
  <ref bean="..." />   -- other beans
  <bean class="...">   -- inner-beans
</list>
```

### SET

```
<set>
  ...
</set>
```

Кафедра экономической информатики. Бгуир. 2017

80



## PISL14. Spring (введение)

```
<bean id="combinedEventLogger" class="...">
  <constructor-arg>
    <list>
      <ref bean="consoleEventLogger"/>
      <ref bean="fileEventLogger"/>
    </list>
  </constructor-arg>
</bean>
```

## PISL14. Spring (введение)



## PISL14. Spring (введение)

### MAP

```
<map>
  <entry key="..." value="..."/>
  <entry key="..." value-ref="..."/>
  <entry key-ref="..." value-ref="..."/>
  <entry key="...">
    <bean class="..."/>
  </entry>
</map>
```

## PISL14. Spring (введение)

```
<bean id="app" class="...">
  <constructor-arg>
    <map>
      <entry key="INFO"
        value-ref="consoleEventLogger"/>
      <entry key="ERROR"
        value-ref="combineEventLogger"/>
    </map>
  </constructor-arg>
</bean>
```

## PISL14. Spring (введение)

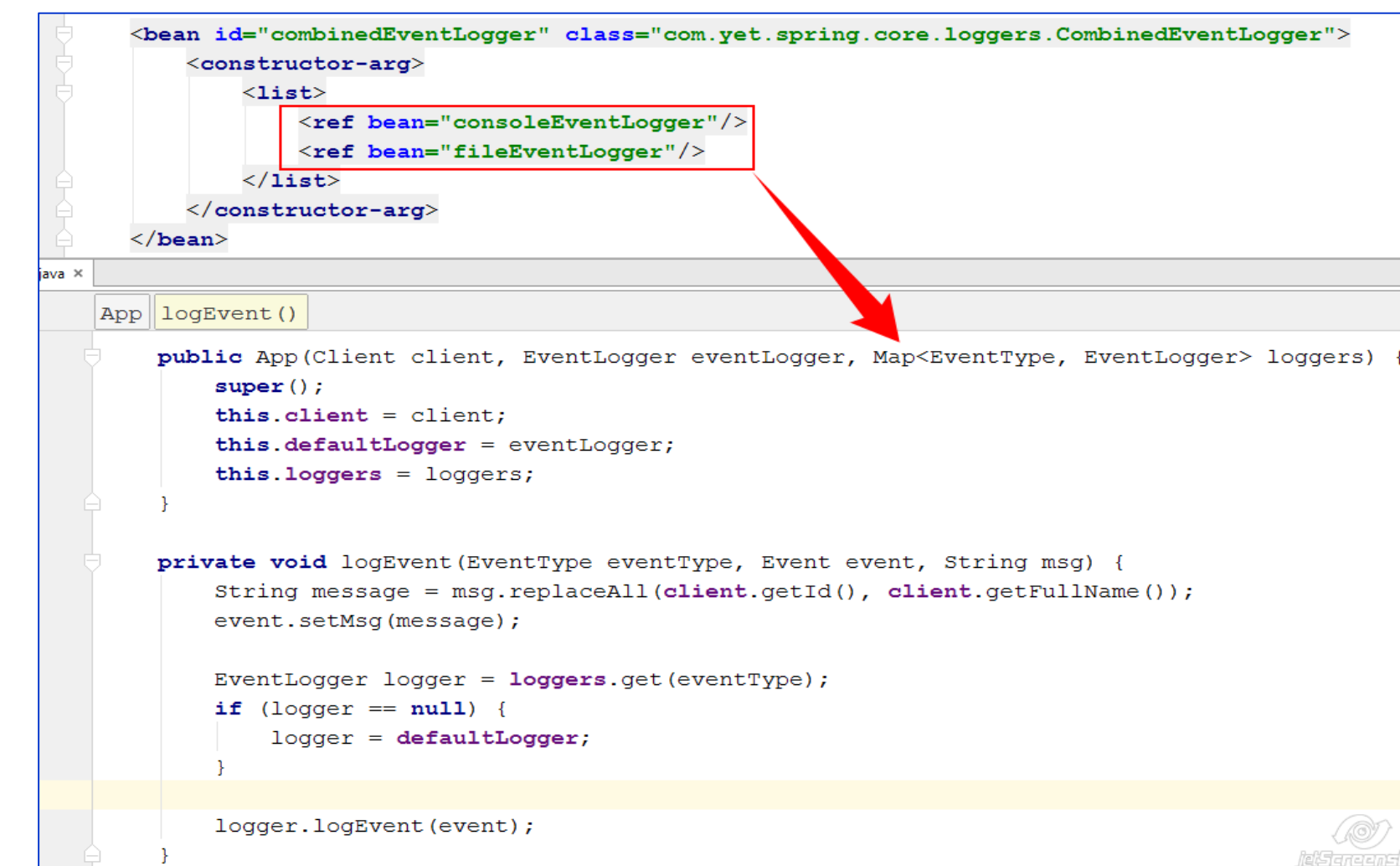
```
public App(Client client, EventLogger eventLogger,
           Map<EventType, EventLogger> loggers) {
    ...
}

private void logEvent(EventType type, String msg) {
    EventLogger logger = loggers.get(type);
    if (logger == null) {
        logger = defaultLogger;
    }
    logger.logEvent(event);
}
```

Кафедра экономической информатики. Бгуир. 2017

85

## PISL14. Spring (введение)



```
<bean id="combinedEventLogger" class="com.yet.spring.core.loggers.CombinedEventLogger">
  <constructor-arg>
    <list>
      <ref bean="consoleEventLogger"/>
      <ref bean="fileEventLogger"/>
    </list>
  </constructor-arg>
</bean>
```

```
public App(Client client, EventLogger eventLogger, Map<EventType, EventLogger> loggers) {
    super();
    this.client = client;
    this.defaultLogger = eventLogger;
    this.loggers = loggers;
}

private void logEvent(EventType eventType, Event event, String msg) {
    String message = msg.replaceAll(client.getId(), client.getFullName());
    event.setMsg(message);

    EventLogger logger = loggers.get(eventType);
    if (logger == null) {
        logger = defaultLogger;
    }
    logger.logEvent(event);
}
```

Кафедра экономической информатики. Бгуир. 2017

86

## PISL14. Spring (введение)

### JAVA.UTIL.PROPERTIES

```
<props>
  <prop key="...">...</prop>
  <prop key="...">...</prop>
</props>
```

Кафедра экономической информатики. Бгуир. 2017

87

## PISL14. Spring (введение)

### JAVA.UTIL.PROPERTIES

```
<props>
  <prop key="...">...</prop>
  <prop key="...">...</prop>
</props>
```

### NULL

```
<property name="someNullProperty">
  <null/>
</property>
```

Кафедра экономической информатики. Бгуир. 2017

88



## PISL14. Spring (введение)

- ⤵ Что почитать?
- ⤵ Левитин А.В. **Алгоритмы: введение в разработку и анализ.**  
– М.: Вильямс, 2006. – 576 с. (С. 267-271)
- ⤵ 2. Вирт Н. **Алгоритмы и структуры данных.**  
– М.: Мир, 1989. – 360 с. (С. 272-286)
- ⤵ 3. AVL-деревья // Сайт RSDN.ru. – URL:  
<http://www.rsdn.ru/article/alg/bintree/avl.xml>
- ⤵ 4. Любой поисковик: “avl tree” || “avl tree ext:pdf” || “avl tree ext:ppt”