S02: Stranger Spring

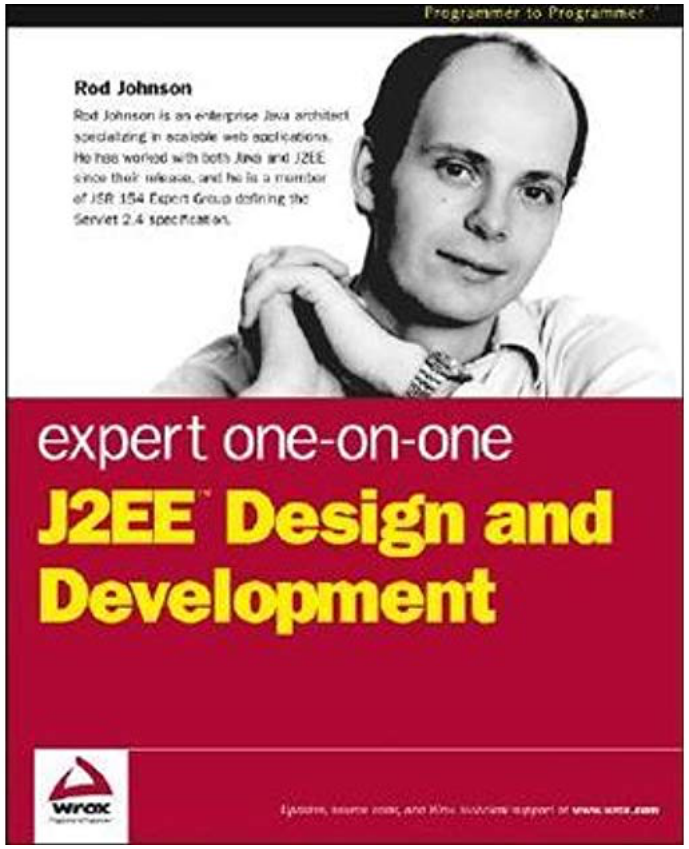
E01: S is for Spring

Tomek Adamczewski

15.10.2019

Historia

JAVA FAKTURA



The results of ***using J2EE in practice*** *are* ***often disappointing*** *– applications are often slow, unduly complex, and take too long to develop.* ***I believe that the problem lies not in J2EE itself****, but in that it is often used badly. Many J2EE publications advocate approaches that, while fine in theory, often* ***fail in reality, or deliver no real business value****.*

Rod Johnson

Założenia

JAVA FAKTURA

Kiedyś

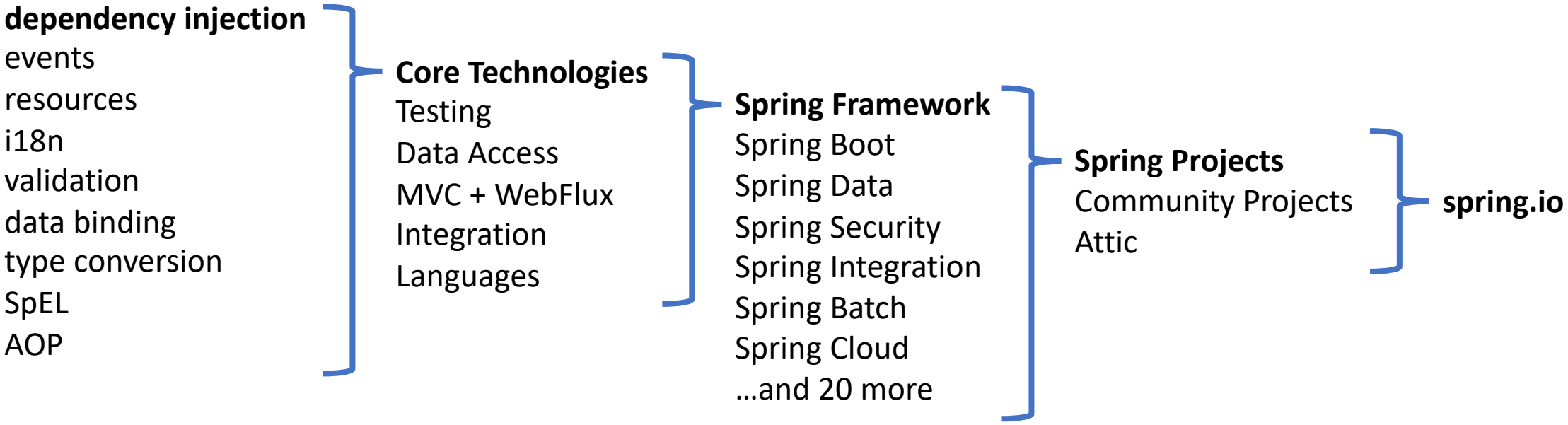
- Lekki
- Z niskim progiem wejścia
- Zwiększający produktywność
- Modularny
- Nieinwazyjny
- Wspierający testowalność
- Bazujący na dobrych specyfikacjach J2EE i bibliotekach

Teraz

- Pozwalający opóźniać decyzje projektowe
- Nie narzucający standardów
- Silnie kompatybilny wstecznie
- Zaprojektowany z dbałością o API
- Zaimplementowany w oparciu o wysokie standardy jakości kodu

Co znaczy “Spring”?

JAVA FAKTURA

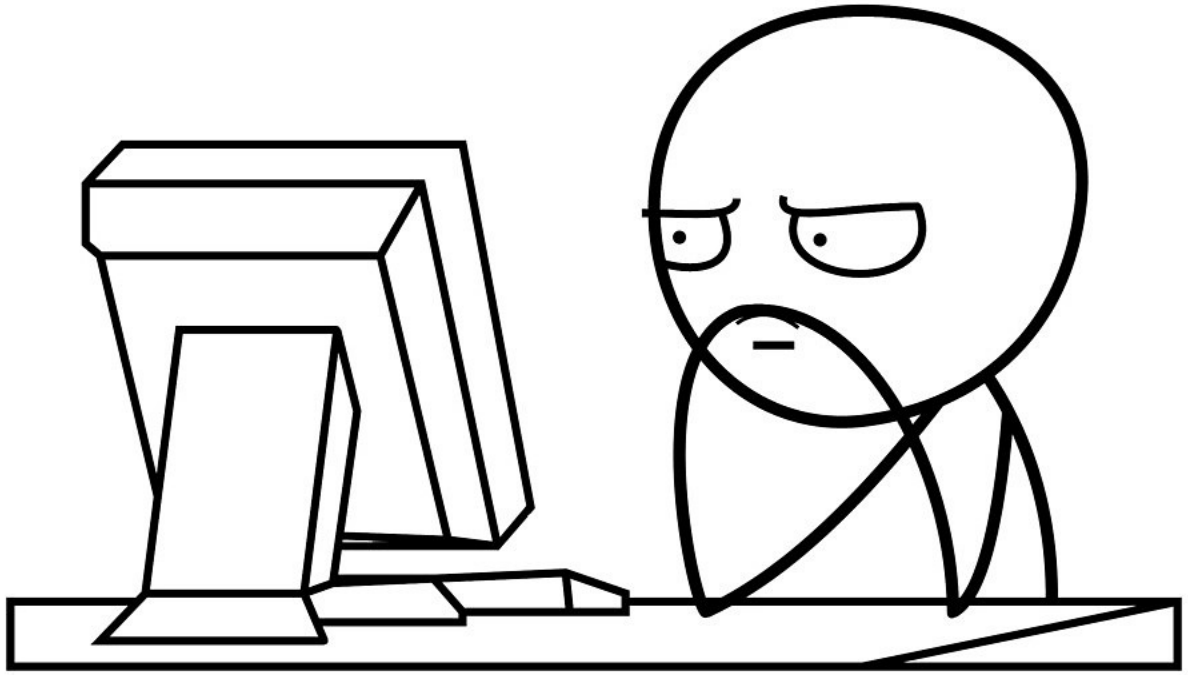


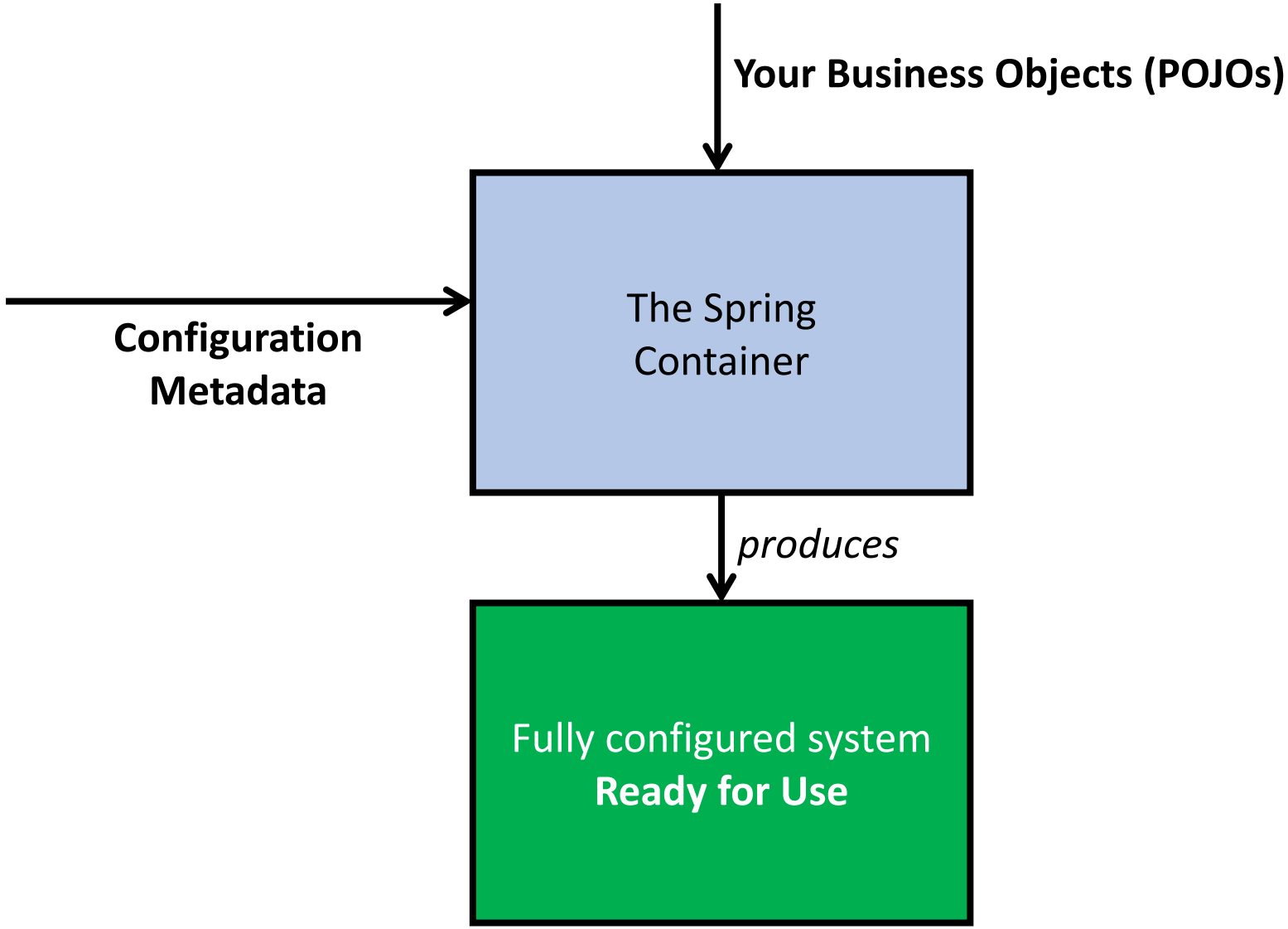
Jak poznać Springa?

JAVA FAKTURA

- docs.spring.io
- spring.io/guides
- github.com/spring-projects

Wzorzec Dependency Injection





https://docs.spring.io/spring-framework/docs/current/spring-framework-reference/core.html

Konfigurowanie beanów: context.xml

```xml
<beans>
  <bean id="myBean" class="…">
    <constructor-arg name="myDep" ref="myDep" />
    <property name="otherDep">
      <bean class="…"/>
    </property>
  </bean>

  <bean id="myDep" class="…"/>
</beans>
```

```
new ClassPathXmlApplicationContext("classpath*:/spring/*-context.xml");
```

Konfigurowanie beanów: @Adnotacje

```
1 @Component
2 class MyBean {
3
4   private Dependency myDep;
5
6   MyBean(Dependency myDep) {
7     this.myDep = myDep;
8   }
9 }
```

```
@ComponentScan("io.github.javafaktura"); // ~classpath*:/io/github/javafaktura/**/*.class
```

Konfigurowanie beanów: JavaConfig

```
1 @Configuration
2 class ModuleConfig {
3
4   @Bean Dependency myDep() {
5     return new Dependency();
6   }
7
8   @Bean MyBean myBean(Dependency myDep) {
9     return new MyBean(myDep);
10  }
11 }
```

```
new AnnotationConfigApplicationContext(ModuleConfig.class);
```

Konfigurowanie beanów: inne

- DSLe: Kotlin, Groovy
- @Configurable

Wstrzykiwanie zależności: field injection

```
1 @Component
2 class MyBean {
3
4   @Autowired
5   private Dependency myDep;
6 }
```

Wstrzykiwanie zależności: setter injection

```
1 @Component
2 class MyBean {
3
4   private Dependency myDep;
5
6   @Autowired
7   void setMyDep(Dependency myDep) {
8     this.myDep = myDep;
9   }
10 }
```

Dependency Injection: constructor injection

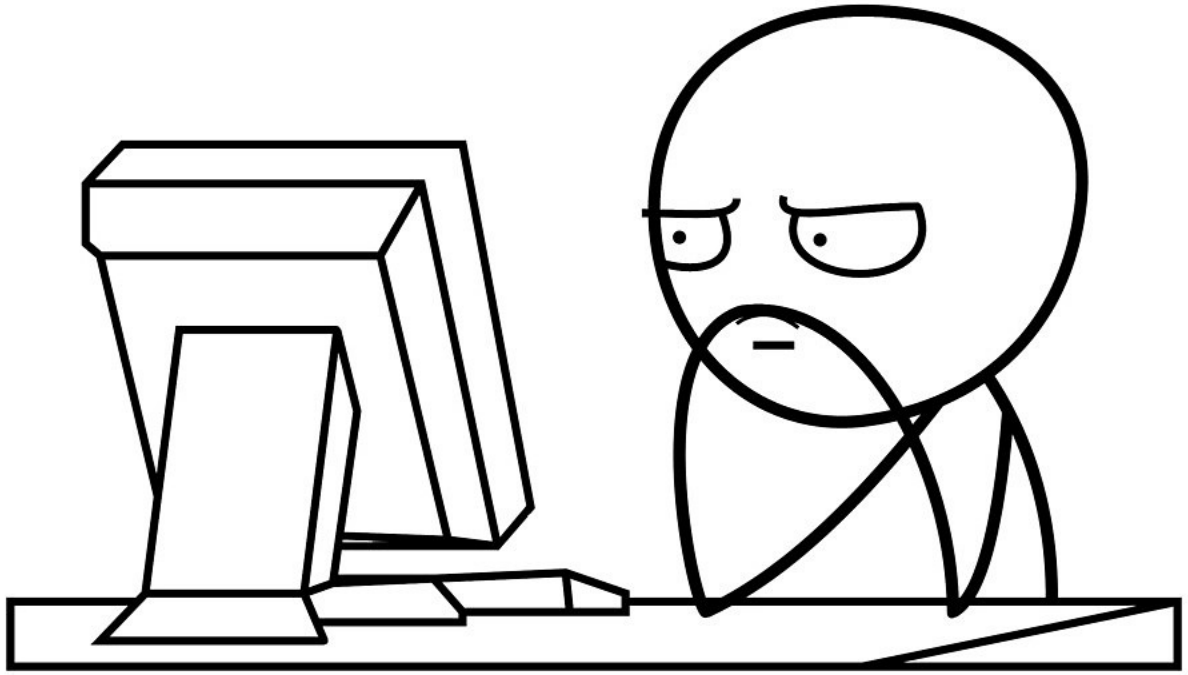
```
1  @Component
2  class MyBean {
3
4    private final Dependency myDep;
5
6    @Autowired
7    MyBean(Dependency myDep) {
8      this.myDep = myDep;
9    }
10 }
```

Dependency Injection

The Spring team *generally* ***advocates constructor injection****, as it lets you* ***implement application components as immutable objects*** *and* ***ensures that required dependencies are not null****. Furthermore, constructor-injected components are always returned to the client (calling) code in a* ***fully initialized state****.*
As a side note, ***a large number of constructor arguments is a bad code smell****, implying that the class likely has too many responsibilities and should be refactored to better address proper separation of concerns.*

https://docs.spring.io/spring/docs/current/spring-framework-reference/core.html

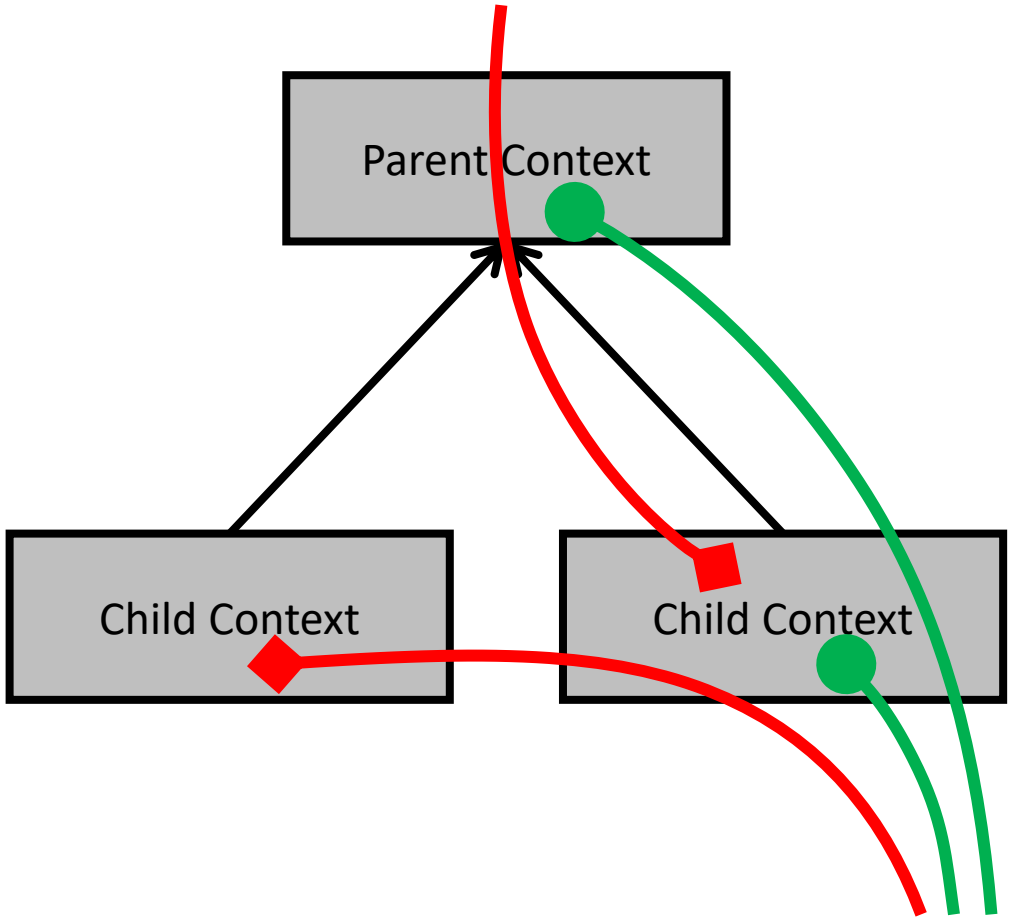
Spring IoC Container



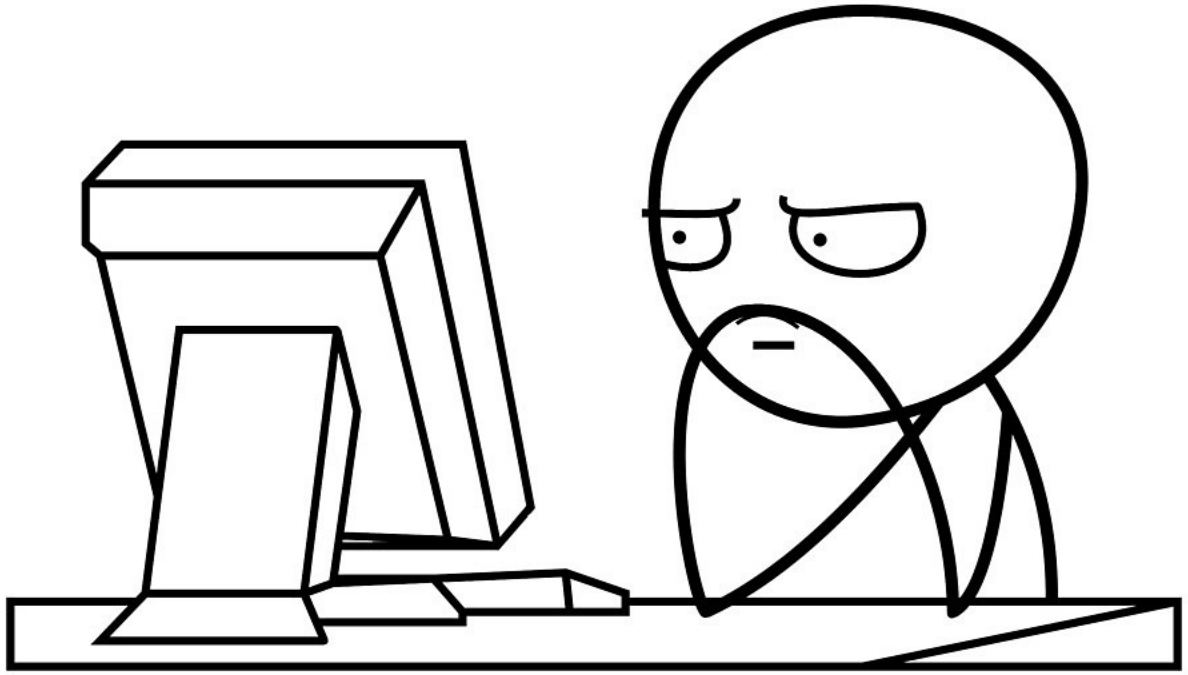
Komponowanie kontekstów

```
// Importowanie konfiguracji
@Configuration
@Import(NotificationsModuleConfig.class)
class AccountingModuleConfig

// Listowanie konfiguracji przy tworzeniu kontekstu
new AnnotationConfigApplicationContext(
            AccountingModuleConfig.class,
            NotificationsModuleConfig.class);

// Skanowanie classpathu
@ComponentScan("io.github.javafaktura")
```

Hierarchia kontekstów



Komponowanie kontekstów



Scope beanów

- Cykle standardowe:
  - Singleton
  - Prototype
- Cykle webowe:
  - Request
  - Session
  - Application
  - Websocket
- Cykle własne
- Method injection

Reagowanie na cykl życia beanów

- @PostConstruct, @PreDestroy
- InitializingBean, DisposableBean
- ApplicationContextAware

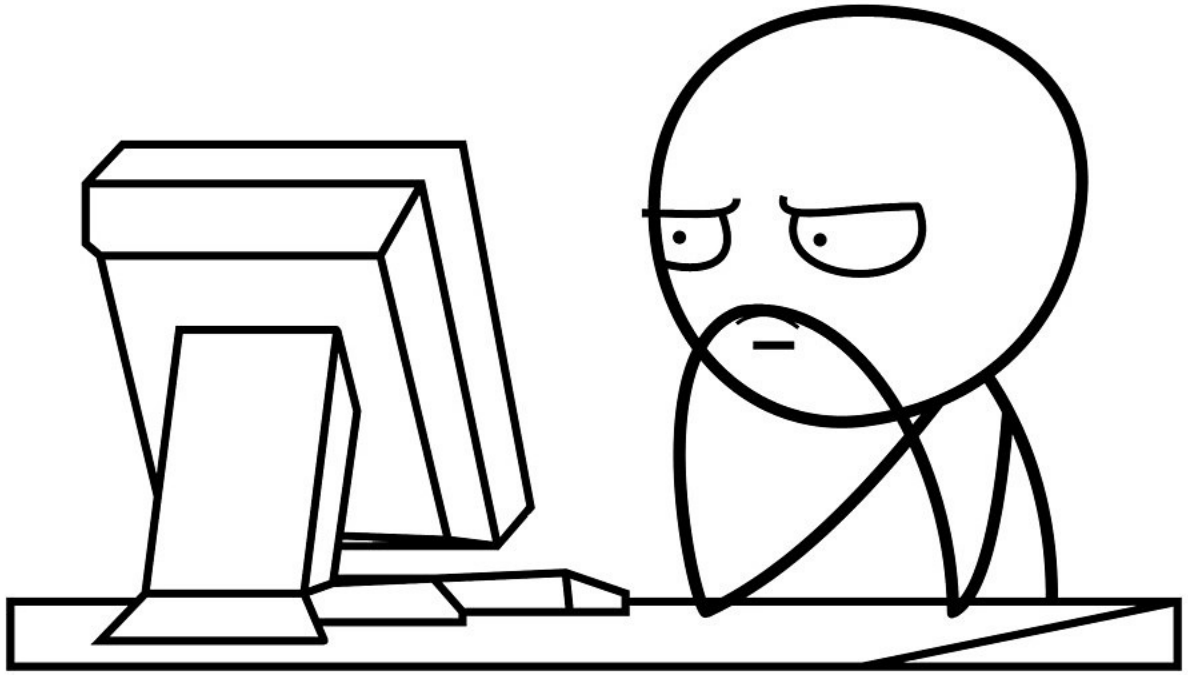
Profile

```
 1 @Configuration
 2 @Profile({"foo", "!bar"})
 3 class ModuleConfig
(…)
11 @ExtendWith(SpringExtension.class)
12 @ActiveProfiles("bar")
```

ApplicationEvents

- Eventy związane z cyklem życia kontekstu (start, stop, refresh)
- Definiowanie własnych eventów
- ApplicationListener
- ApplicationEventPublisher
- ApplicationEventMulticaster

Scope beanów, eventy



Skąd tyle hejtu?

JAVA FAKTURA

- Kompilacja vs runtime
- Adnotacje
- Powolna inicjalizacja kontekstu (classpath scanning, refleksja, duże aplikacje)
- Wymaga dyscypliny i stosowania dobrych wzorców projektowych

Duże aplikacje są skomplikowane – to nie wina Springa

Zaraz niespodzianka, a tymczasem…

JAVA FAKTURA

https://tinyurl.com/jfs02e01survey