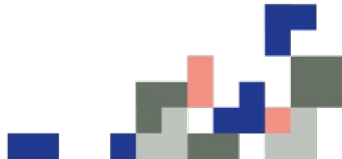
JUG SAXONY DAY

Roland Weisleder
Softwareentwicklung & Beratung

Softwarefehler mit ArchUnit aufspüren

@Ro_Wei
roland@rweisleder.de



Sicherheitslücke

Kein Logging

Überlastung durch
fehlende Caches

Gescheitertes Projekt

NullPointerExceptions
durch falsche Serialisierung

Datenkorruption

Nicht-funktionale Anforderungen nicht bekannt oder falsch umgesetzt

Kann euch das auch passieren?

Codebasis ändert sich

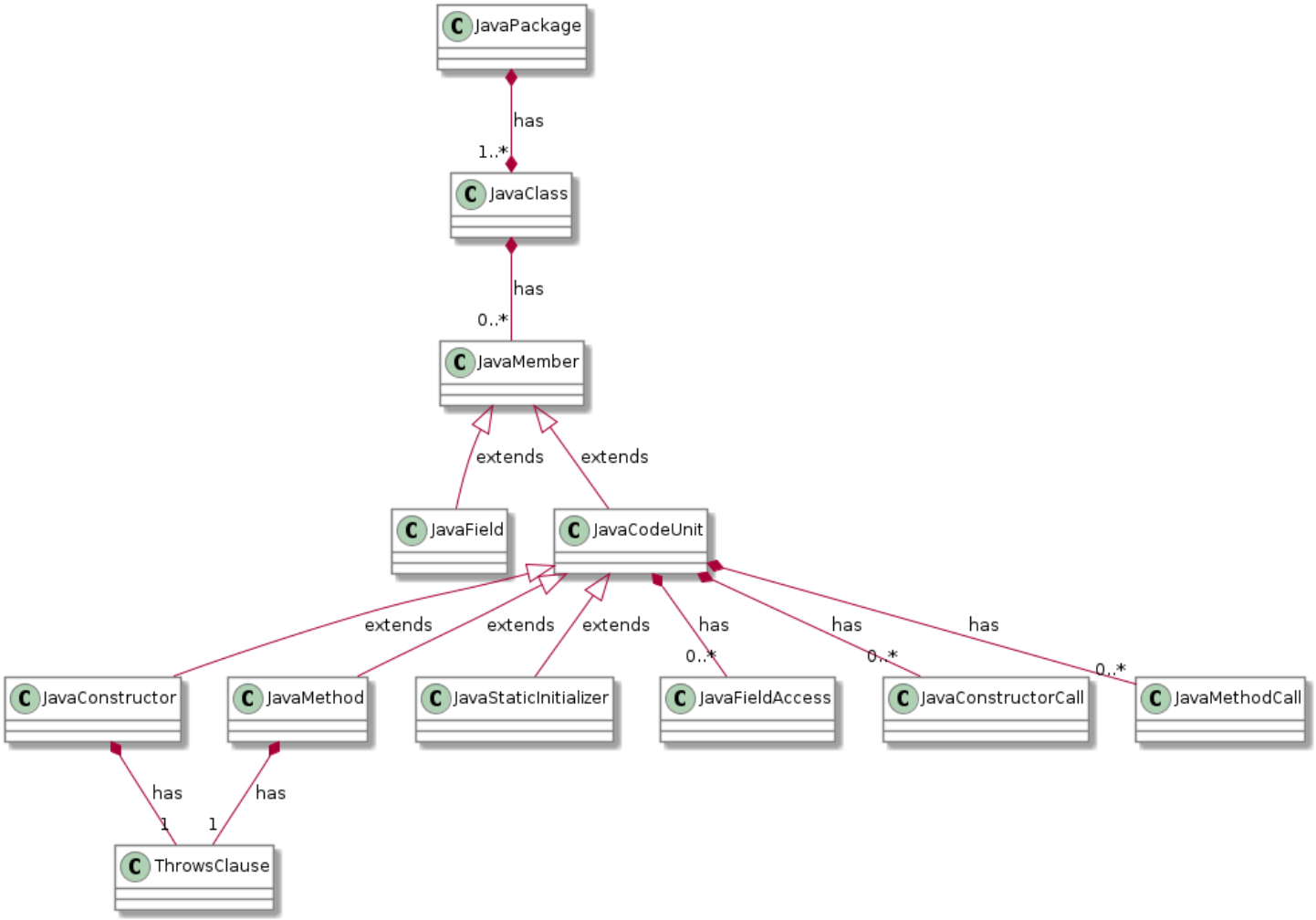
Team ändert sich

Mehr automatisiert testen?

Anforderungen ändern sich

“ArchUnit is a free, simple and extensible library for checking the architecture of your Java code using any plain Java unit test framework. That is, ArchUnit can check dependencies between packages and classes, layers and slices, check for cyclic dependencies and more. It does so by analyzing given Java bytecode, importing all classes into a Java code structure.”

https://www.archunit.org/


JavaPackage
has
1..*
JavaClass
has
0..*
JavaMember
extends
extends
JavaField
JavaCodeUnit
extends
extends
extends
has
has
has
0..*
0..*
0..*
JavaConstructor
JavaMethod
JavaStaticInitializer
JavaFieldAccess
JavaConstructorCall
JavaMethodCall
has
has
1
1
ThrowsClause

```
@AnalyzeClasses
public class ArchitectureTest {

    @ArchTest
    public static final ArchRule SERVICE_NAMING_RULE =
        classes()
            .that().areAnnotatedWith(Service.class)
            .should().haveSimpleNameEndingWith("Service");
}
```

```
classes()
    .that().areAnnotatedWith(Service.class)
    .should().haveSimpleNameEndingWith("Service");
```

Architecture Violation - Rule 'classes that are annotated with @Service should have simple name ending with 'Service" was violated:

Class <de.rweisleder.example.users.business.UserNameservice> does not have simple name ending with 'Service' in (UserNameservice.java:0)

Domain

Flexibler Zugriff auf komplette Struktur

Language

Definition von Regeln mit Fluent-API inkl. Erweiterbarkeit

Library

Vorgefertigte Architektur-Regeln

Codestruktur mit ArchUnit testen

Fehleranfällige Strukturen mit ArchUnit finden?

```java
@Service
public class UserService {

    @Secured("ROLE_ADMIN")
    public void addUser(String name) { ... }

    @Secured("ROLE_ADMIN")
    public void deleteUserWithName(String name) { ... }

    public void deleteAllUsers() { ... }
}
```

@Secured fehlt

```
methods()
    .that().arePublic()
    .and().areDeclaredInClassesThat()
              .areAnnotatedWith(Service.class)
    .should().beAnnotatedWith(Secured.class)
```

Architecture Violation - Rule 'methods that are public and are declared in classes that are annotated with @Service should be annotated with @Secured' was violated:

Method <de.rweisleder.example.users.business.UserService.deleteAllUsers()> is not annotated with @Secured in (UserService.java:35)

```
noClasses()
    .should().dependOnClassesThat()
    .resideInAPackage("org.apache.logging.log4j..")
    .as("classes should not depend on Log4J")
```

Architecture Violation - Rule 'classes should not depend on Log4J' was violated:

Field <de.rweisleder.example.users.persistence.UserRepository.log> has type <org.apache.logging.log4j.Logger> in (UserRepository.java:0)

Method <de.rweisleder.example.users.persistence.UserRepository.addUser(de.rweisleder.example.users.persistence.UserEntity)> calls method <org.apache.logging.log4j.Logger.debug(java.lang.String)> in (UserRepository.java:21)

Static Initializer <de.rweisleder.example.users.persistence.UserRepository.<clinit>()> calls method <org.apache.logging.log4j.LogManager.getLogger(java.lang.Class)> in (UserRepository.java:12)

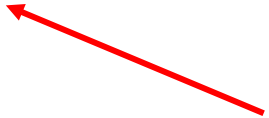
```java
import org.codehaus.jackson.annotate.JsonProperty;

public class UserDto {

    @JsonProperty("id")
    private long id;

    @JsonProperty("displayName")
    private String name;

}
```



verimportiert

```
{
    "id": 1,
    "name": "John Doe"
}
```

```java
import org.codehaus.jackson.annotate.JsonProperty;
import com.fasterxml.jackson.annotation.JsonProperty;

public class UserDto {

    @JsonProperty("id")
    private long id;

    @JsonProperty("displayName")
    private String name;

}
```

```
noClasses()
    .should().dependOnClassesThat()
    .resideInAPackage("org.codehaus.jackson..")
    .as("classes should not depend on Jackson v1")
    .because("Jackson v2 should be used instead");
```

Architecture Violation - Rule 'classes should not depend on Jackson v1, because Jackson v2 should be used instead' was violated:

Field <de.rweisleder.example.users.boundary.UserDto.id> is annotated with <org.codehaus.jackson.annotate.JsonProperty> in (UserDto.java:0)

Field <de.rweisleder.example.users.boundary.UserDto.name> is annotated with <org.codehaus.jackson.annotate.JsonProperty> in (UserDto.java:0)

Fehlerhafte Strukturen mit ArchUnit finden

- Fehlender/Überflüssiger Code
- Fehlerhafte Imports
- Fehlerhafte Verwendung von Librarys/Frameworks
- Ungewollte Abhängigkeiten zwischen Klassen
- Abweichungen von der Architektur

Warum ArchUnit dafür einsetzen?

- Niemand hat ständig alle Regeln im Kopf
- Niemand wird ständig alle Regeln nachlesen
- Jemand wird Tests vergessen
- Regel können sich ändern
- Umsetzung von nicht-funktionalen Anforderungen zentral testen
- Vorhandener und neuer Code wird getestet

“Testing can detect only the presence of errors, not their absence.”

Sofwarefehler mit ArchUnit aufspüren

Roland Weisleder
Softwareentwicklung & Beratung

Code:

https://github.com/rweisleder/find-bugs-with-archunit-examples

https://github.com/TNG/ArchUnit-Examples

Doku:

https://www.archunit.org/

Beratung & Unterstützung in eurem Legacy-System benötigt?

roland@rweisleder.de

rweisleder.de

@Ro_Wei