eXtended Objects
Interfaces are better than POJOs

Java User Group Saxony
12/2014

AGENDA

- The POJO problem
- Interfaces: Less Code, More Flexibility
- eXtended Objects
- Live Demo
- Beyond Graphs

eXtended Objects - Interfaces are better than POJOs

The POJO problem

- POJOs are commonly used for mapping
  - Domain objects
  - Persistent data structures

```
public class … {
  long id; // 1
  String name; // „Object“
  boolean abstract; // false
}
```

ID number(19,0)	NAME varchar(255)	ABSTRACT char(1)
1	Object	n

- Java Persistence API

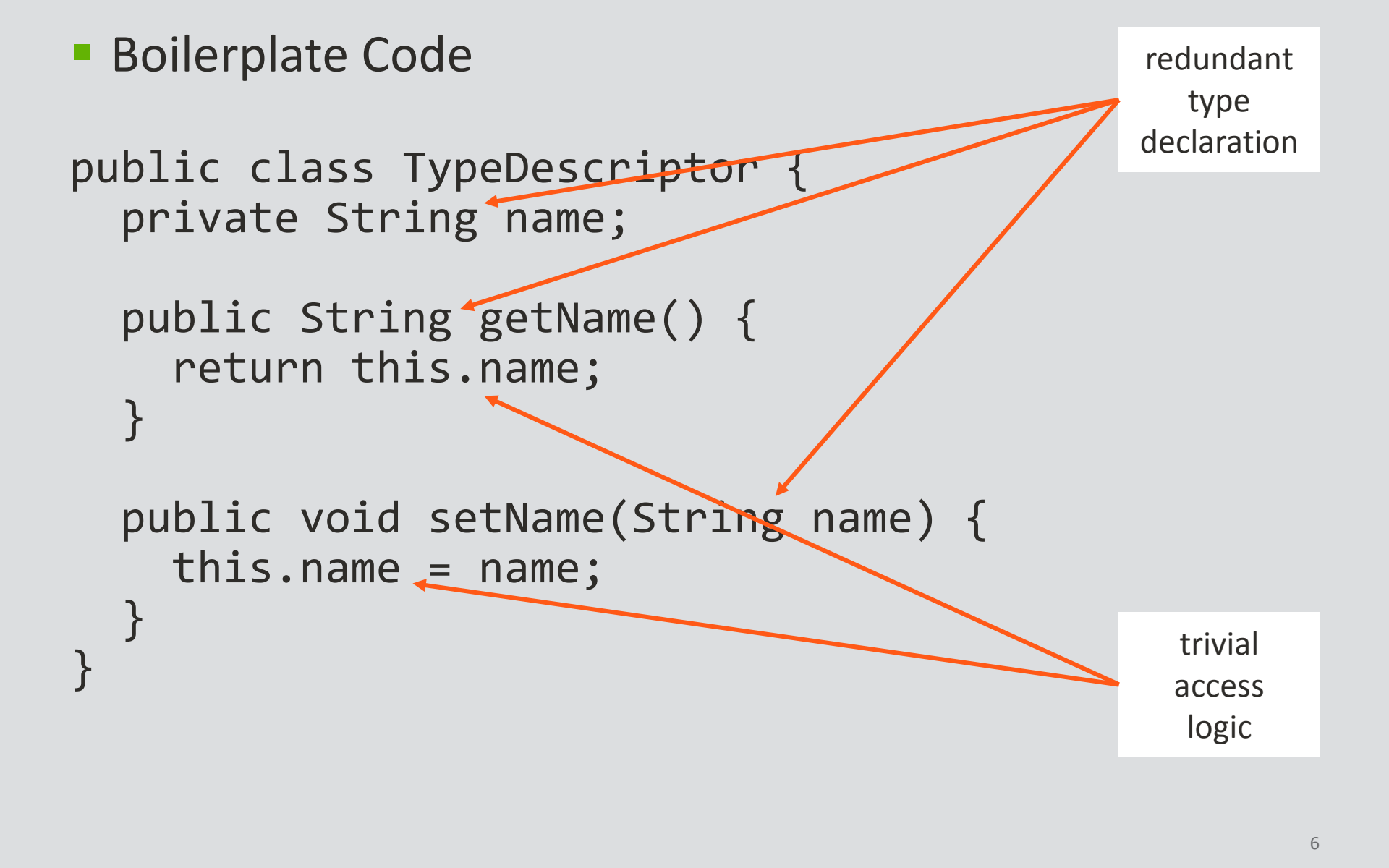
Are POJOs the perfect solution?

- Boilerplate Code

```
public class TypeDescriptor {
  private String name;

  public String getName() {
    return this.name;
  }

  public void setName(String name) {
    this.name = name;
  }
}
```



- Unclear semantics of non-Accessor-Methods

```
public class TypeDescriptor {

  public int hashCode() { … }

  public boolean equals(Object other) { … }

  public String toString() { … }

}
```

- Expected behavior often depends on usage context

- Lack of flexibility
  - Example: properties of Java elements

	Name	Visibility	Abstract
Package	X		
Type	X	X	X
Field	X	X	
Method	X	X	X

  - POJOs only allow single inheritance
    - Is there common base class?
  - Define common properties in interfaces
    - And re-implement them x times for every POJO?

eXtended Objects - Interfaces are better than POJOs

Interfaces: Less Code, More Flexibility

- POJO

```
public class TypeDescriptor {
  private String name;

  public String getName() {
    return this.name;
  }

  public void setName(String name) {
    this.name = name;
  }
}
```

- Interface

```
public interface TypeDescriptor {

   public String getName();


   public void setName(String name);

}
```

- Just declare properties!

- Composition

```
public interface NamedDescriptor {
  String getName();
  void setName(String name);
}

public interface TypeDescriptor extends
  NamedDescriptor,
  AbstractDescriptor, … {
 …
}
```

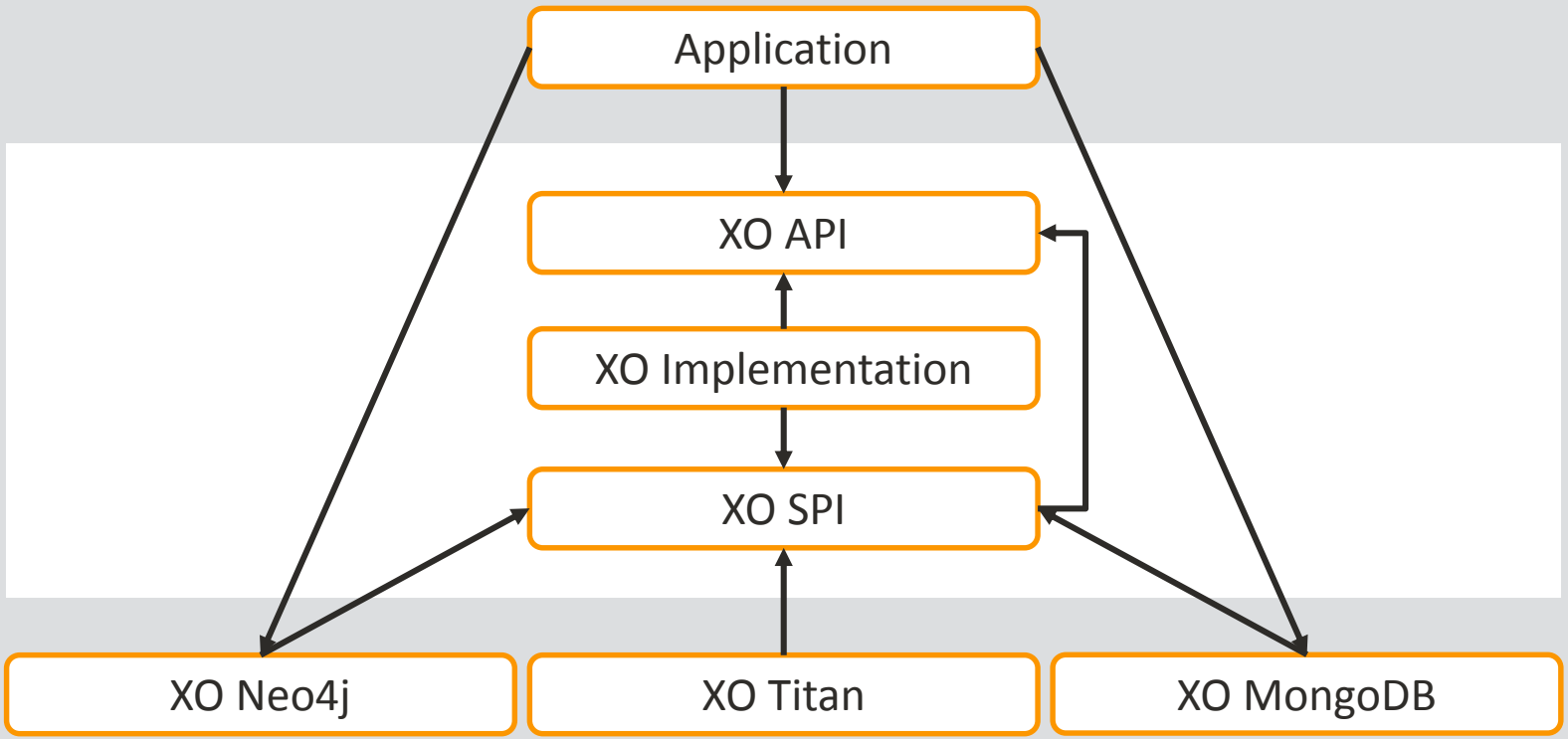
- Separate roles and compose data types!

eXtended Objects - Interfaces are better than POJOs

eXtended Objects

- Project Homepage
  - http://extended-objects.org
  - https://github.com/buschmais/eXtendedObjects
- Open Source
  - Apache Software License 2.0
- Current Release
  - 0.4.5

Architecture



- Features
  - Lightweight
    - Low overhead
    - Only a few 3rd-party dependencies
  - Datastore agnostic API, SPI and implementation
  - Bootstrapping and API similar to JPA
  - Bean Validation
  - Instance Callbacks
  - Configurable Transaction/Concurrency Management
  - Example API (Java 8 → Lambda expressions)
  - Pluggable query languages (e.g. Lucene, Gremlin)
  - OSGi compatible

- Graph Datastores
  - Neo4j
    - https://github.com/buschmais/extended-objects
  - OrientDB
    - https://github.com/SMB-TEC/xo-orientdb
  - Tinkerpop
    - https://github.com/SMB-TEC/xo-tinkerpop-blueprints
  - Titan
    - https://github.com/PureSolTechnologies/extended-objects-titan

eXtended Objects - Interfaces are better than POJOs

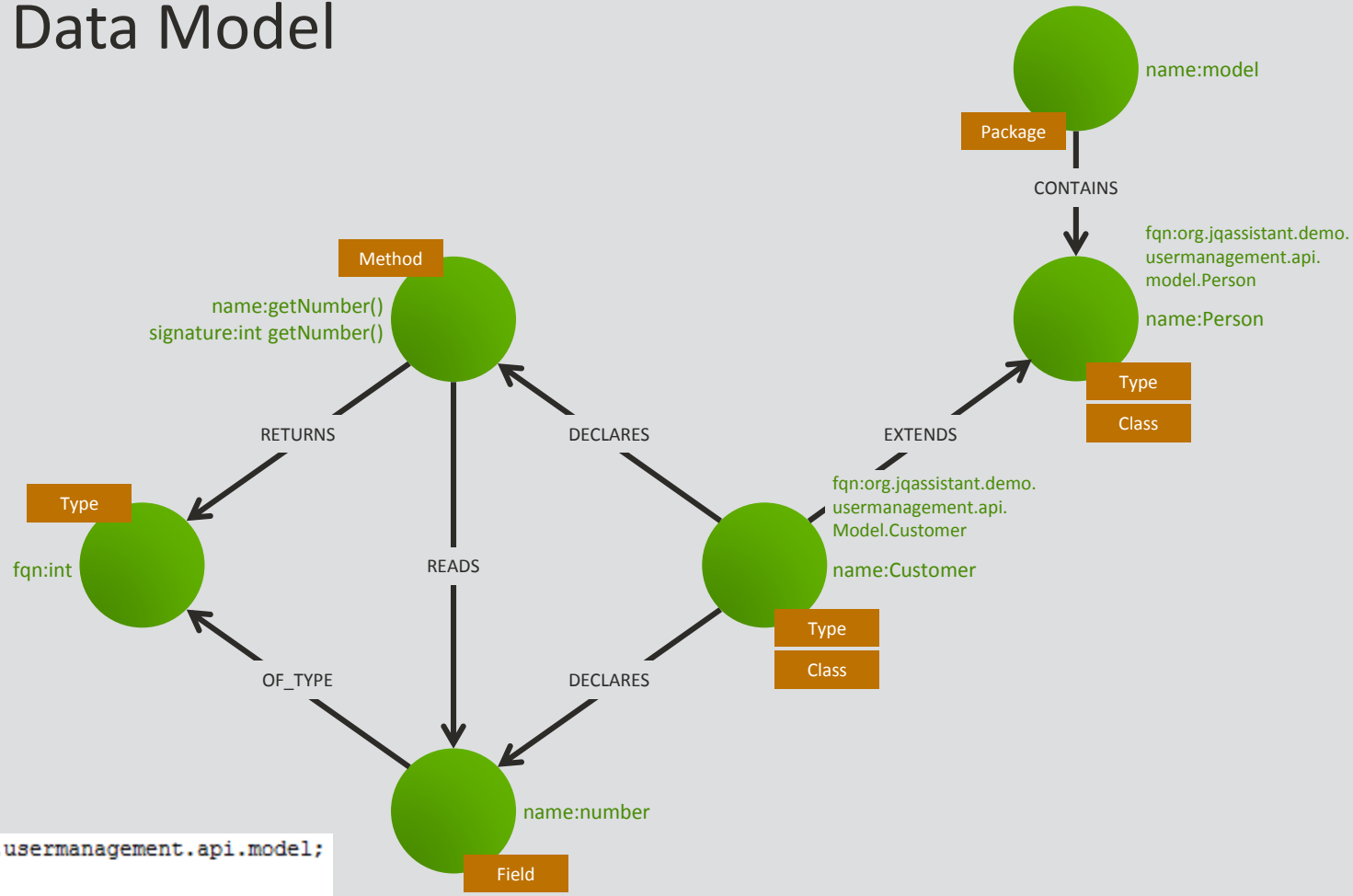
Live Demo

- Entities
  - Static Composition
    - Entities and Templates
  - Dynamic Composition
    - CompositeObject
  - Type Migration

- Relations
  - Unqualified
  - Qualified
  - Typed

- Queries
  - Simple
  - Typed
- Active Properties
  - @Implemented By
  - @ResultOf
- Repositories
  - Generic
  - Datastore Specific
- Example API

Example Data Model




```
package org.jqassistant.demo.usermanagement.api.model;

import javax.persistence.Entity;
import javax.persistence.Id;

@Entity
public class Person {
```

eXtended Objects - Interfaces are better than POJOs

Beyond Graphs

- MongoDB datastore
  - GitHub: https://github.com/SMB-TEC/xo-mongodb
- Documents are map-like structures
  - JSON/BSON
- Better support needed by eXtended Objects SPI
  - Arbitrary depth of primitive values
  - Requires a @Embedded-like concept

- Example

```
@Document
public interface Article {

    String getTitle();
    void setTitle(String title);

    @Indexed(unique = true)
    String getURL();
    void setURL(String url);

    @DBRef
    Author getAuthor();
    void setAuthor(Author author);

    @Property("blog_comments")
    @DBRef
    Set<Comment> getComments();

}
```

Thank you! Questions?

Mail: info@extended-objects.org
Web: extended-objects.org
Twitter: @extendedObjects