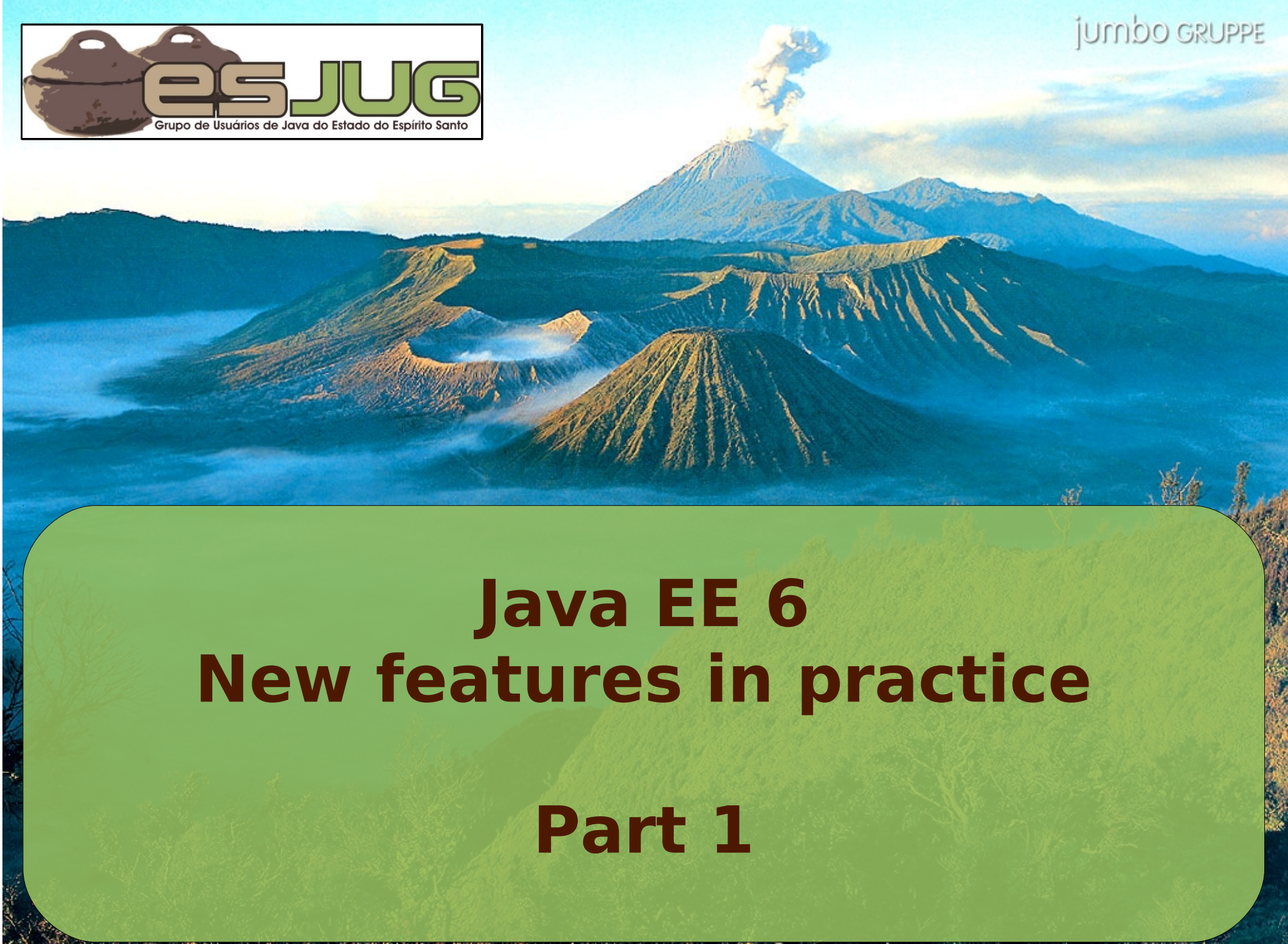


Java EE 6
New features in practice

Part 1

License for use and distribution

This material is available for non-commercial use and can be derived and/or redistributed, as long as it uses an equivalent license.

Attribution-Noncommercial-Share Alike 3.0 Unported

http://creativecommons.org/licenses/by-nc-sa/3.0/

You are free to share and to adapt this work under the following conditions: (a) You must attribute the work in the manner specified by the author or licensor (but not in any way that suggests that they endorse you or your use of the work); (b) You may not use this work for commercial purposes. (c) If you alter, transform, or build upon this work, you may distribute the resulting work only under the same or similar license to this one.

About the author – Vítor Souza

- Education:
  - Computer Science graduate, masters in Software Engineering – (UFES, Brazil), starting PhD at U. Trento.
- Java:
  - Developer since 1999;
  - Focus on Web Development;
  - Co-founder and coordinator of ESJUG (Brazil).
- Professional:
  - Substitute teacher at Federal University of ES;
  - Engenho de Software Consulting & Development.
- Contact: vitorsouza@gmail.com

JUG Trento / JUG Bolzano

- Website:
  - http://www.jugtrento.org/
  - http://www.jugbz.org/
- Mailing list (in Italian, mostly): http://groups.google.com/group/jugtaa
- If you like Java, subscribe and participate!

JUGTRENTO.ORG

Java User Group

Agenda

- What is Java EE?
- New features in Java EE 6:
  - Overview;
  - Tools for development (IDE, server);
  - Creating an application with NetBeans;
  - Domain classes with JPA 2.0;
  - Bean Validation;
  - Contexts and Dependency Injection (CDI).

Overview

- Java, Enterprise Edition:
  - Before version 5 it was called J2EE;
  - Platform for the development of enterprise application (scalability, security, accessibility, etc.);
  - Servlets, EJBs, container components with well-defined life-cycles, shared infrastructure;
  - Included technologies: JSP, JDBC, JPA, JSF, etc.
- First version: 1999;
- Version 6: December 2009 – JSR 316.

Main goals

- Flexibility:
  - Profiles. Web Profile already defined;
  - Pruning: optional technologies;
- Extensibility:
  - Extensibility points: automatic registration of non-standard frameworks;
- Ease of development:
  - Started in version 5 (POJOs vs. Entity Beans);
  - Many improvements in version 6.

Java EE Technologies

Technology (API)	Java EE 5	Java EE 6
Bean Validation	–	1.0
Common Annotations for the Java Platform	1.0	1.1
Contexts and Dependency Injection for the Java EE Platform	–	1.0
EJB (Enterprise Java Beans)	3.0	3.1 CP*
EL (Expression Language)	2.1	2.2
Interceptors	–	1.1
JACC (Java Authorization Service Provider Contract for Containers)	1.1	1.4
JASPIC (Java Authentication Service Provider Interface for Containers)	–	1.0
Java EE Deployment API	1.2	1.2 CP
Java EE Management API	1.1	1.1
JavaMail	1.4	1.4
JAX-RPC (Java API for XML-based RPC)	1.1	1.1 CP

CP = Candidates for pruning / * = Entity Beans only

Tecnologie Java EE

Technology (API)	Java EE 5	Java EE 6
JAX-RS (Java API for RESTful Web Services)	–	1.1
JAX-WS (Java API for XML Web Services)	2.0	2.2
JAXB (Java Architecture for XML Binding)	2.0	2.2
JAXR (Java API for XML Registries)	1.0	1.0 CP
JCA (Java EE Connector Architecture)	1.5	1.6
JMS (Java Messaging Service)	1.1	1.1
JPA (Java Persistence API)	1.0	2.0
JSF (JavaServer Faces)	1.2	2.0
JSP (JavaServer Pages)	2.1	2.2
JSTL (Standard Tag Library for JavaServer Pages)	1.2	1.2
JTA (Java Transaction API)	1.1	1.1
Managed Beans	–	1.0
Servlet	2.5	3.0
Web Services Metadata for the Java Platform	2.0	2.1

Tools

- Application Server:
  - GlassFish Enterprise Server v3;
  - TMAX JEUS 7 (commercial, end of 2010);
  - JBoss 6 (M3, still incomplete, not certified);
- Development environment (IDE):
  - NetBeans 6.9;
  - Eclipse with GlassFish Plugins.

NetBeans 6.9 + GlassFish v3

NetBeans IDE 6.9 Download

6.9 | Development | Archive

Email address (optional):

Subscribe to newsletters: ☑ Monthly ☐ Weekly

☑ NetBeans can contact me at this address

IDE Language: English

Platform: Linux (x86/x64)

Note: Greyed out technologies are not supported for this platform.

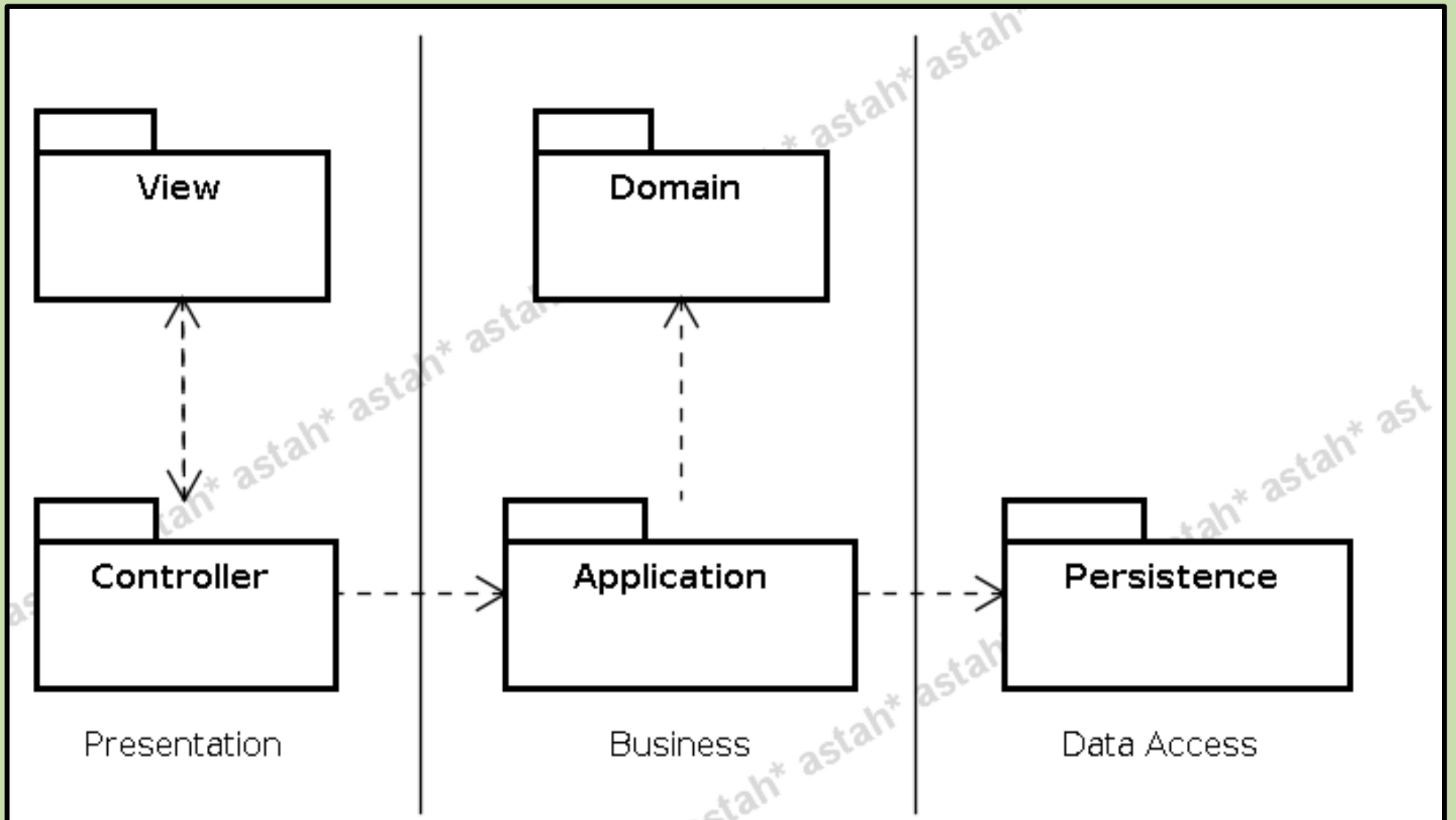
NetBeans IDE Download Bundles

Supported technologies *	Java SE	JavaFX	Java	Ruby	C/C++	PHP	All
NetBeans Platform SDK	●	●	●				●
Java SE	●	●	●				●
JavaFX		●					●
Java Web and EE			●				●
Java ME			●				●
Java Card™ 3 Connected			—				—
Ruby				●			●
C/C++					●		●
Groovy			●				●
PHP						●	●
Bundled servers							
GlassFish Server Open Source Edition 3.0.1			●	●			●
Apache Tomcat 6.0.26			●				●
	Download	Download	Download	Download	Download	Download	Download
	Free, 54 MB	Free, 91 MB	Free, 173 MB	Free, 86 MB	Free, 36 MB	Free, 31 MB	Free, 239 MB

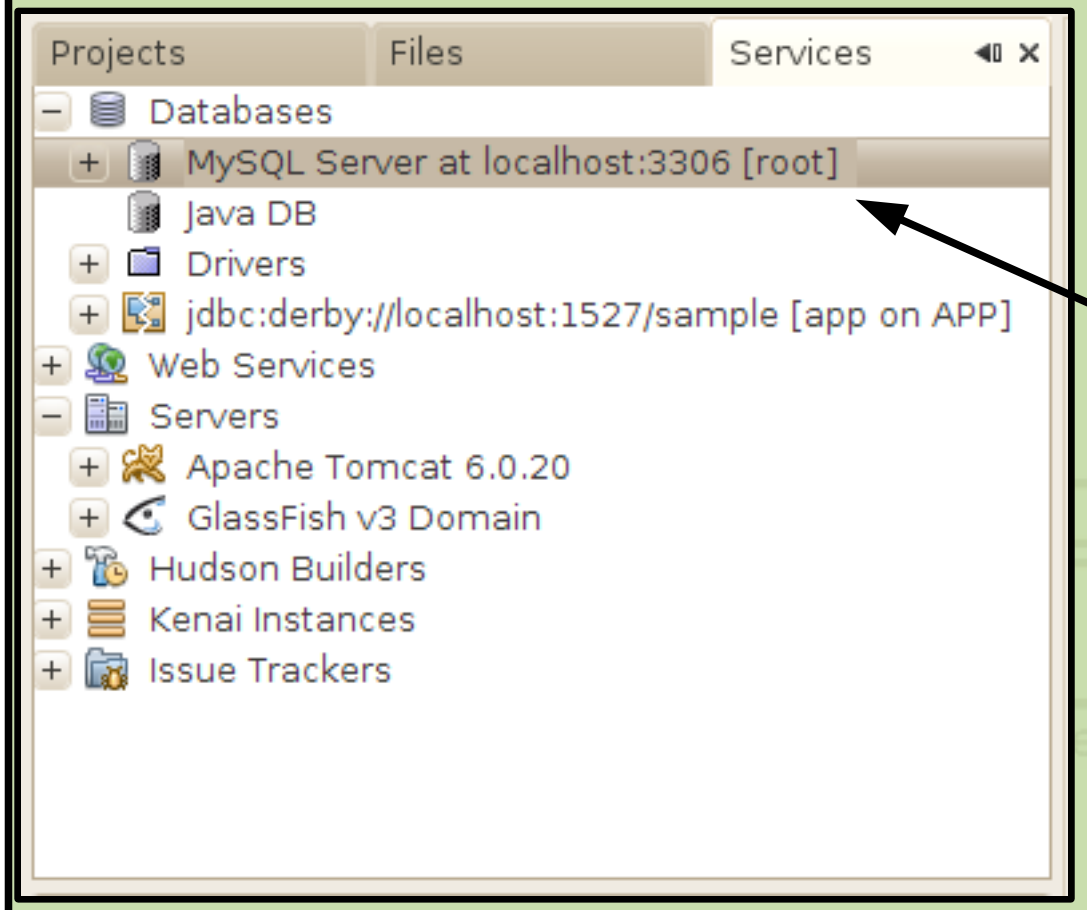
Demonstration – a real application

- Ambulance Dispatch System (ADS):
  - Citizens dial 911 (118, 190, ...) for emergencies;
  - Operator registers the call. Should identify duplicates and filter non-emergencies;
  - Dispatcher searches for a free ambulance as close as possible to emergency site and dispatches it;
  - Driver receives dispatch order and proceeds to emergency site. Ambulance location and status should eb updated in the system;
  - Infrastructure functionality (CRUD, login, etc.).

Architecture



Database



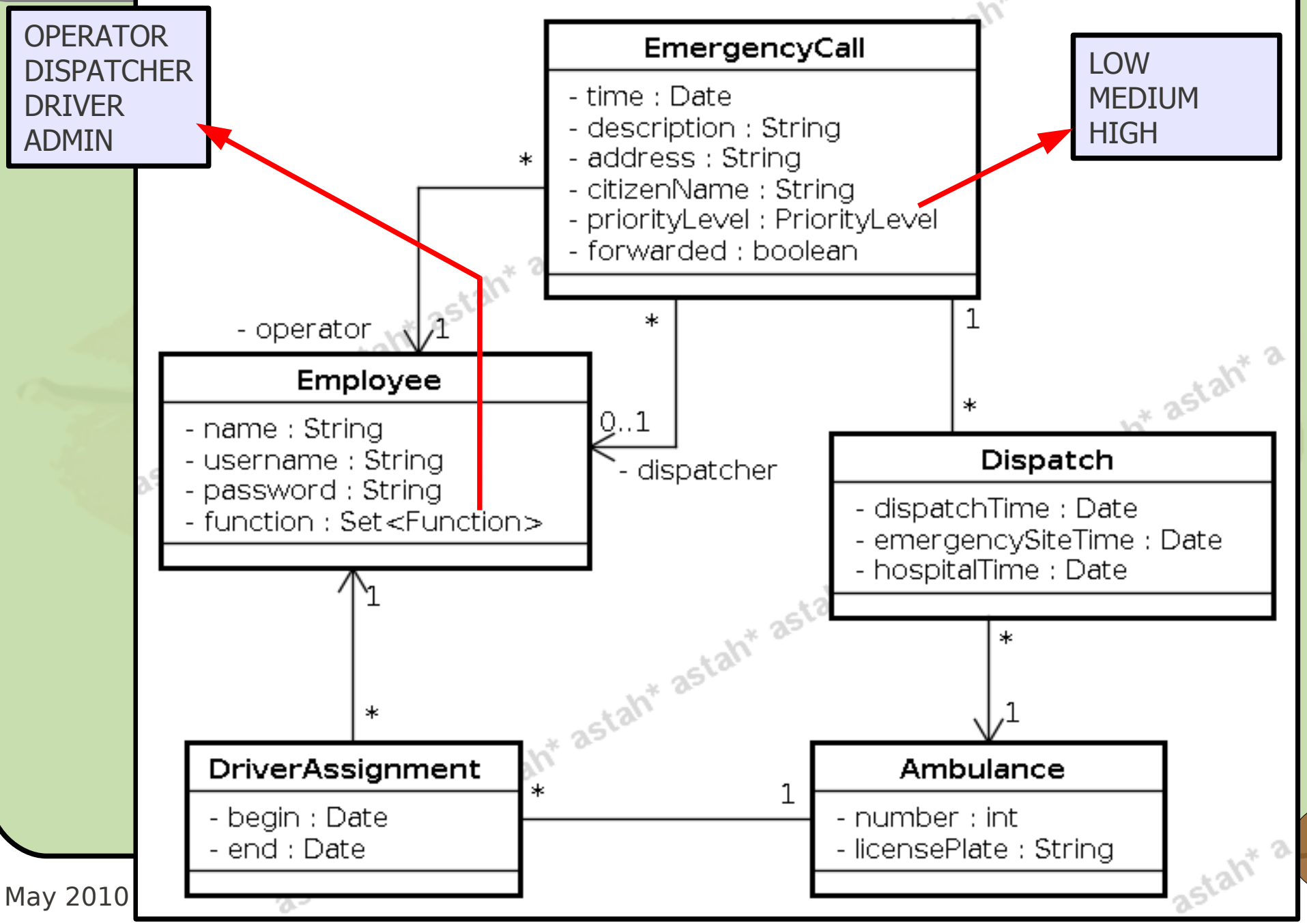
1 - Right-click > Start

2 - Right-click > Create Database...

Tables will be created by JPA automatically.

We will use MySQL in the demonstration.

Domain model



Small persistence framework (1)

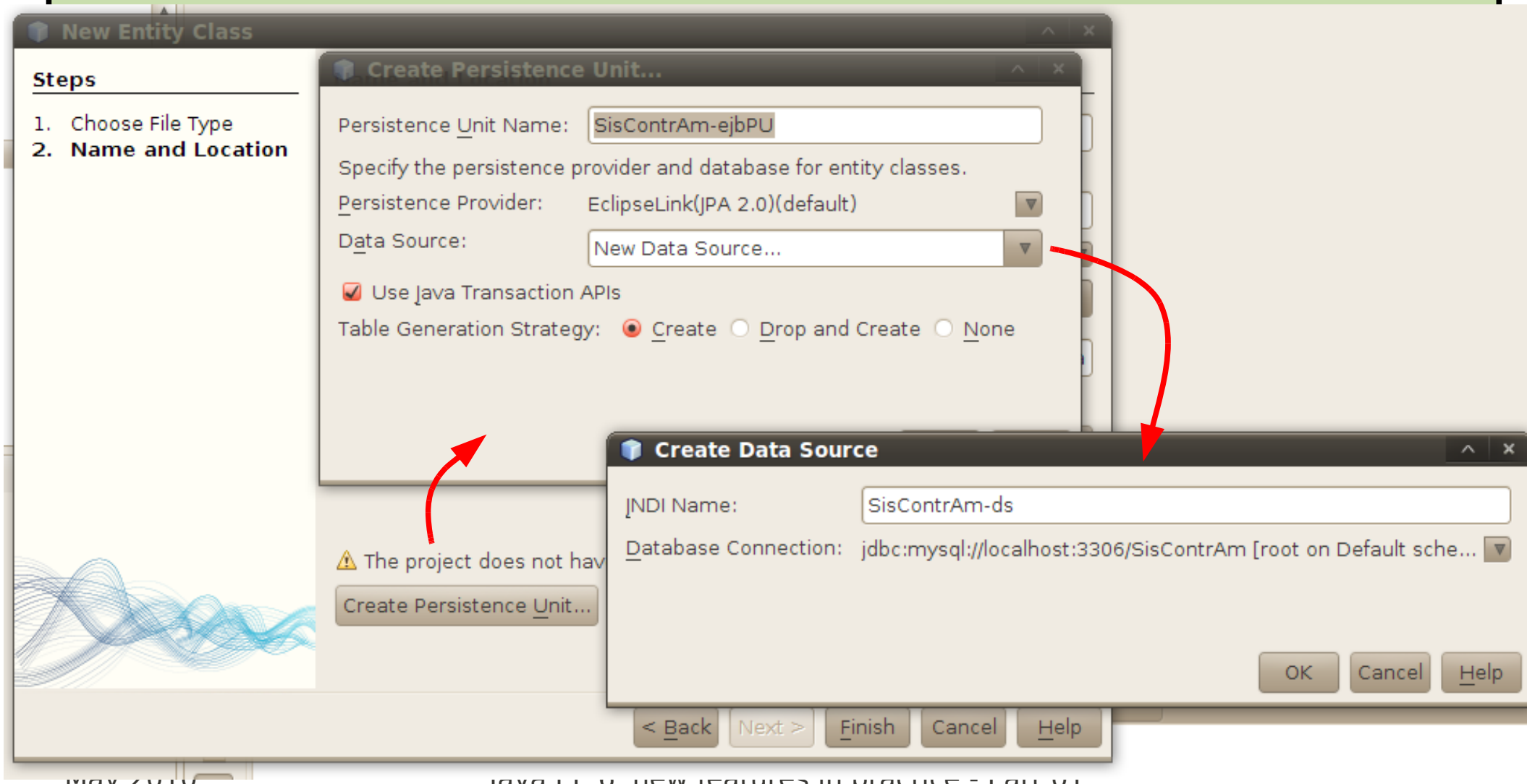
```
@MappedSuperclass
public abstract class PersistentObjectImpl
            implements PersistentObject, Serializable {
   private static final long serialVersionUID = 1L;

   @Basic @Column(nullable = false, length = 40)
   protected String uuid;

   @Id @GeneratedValue(strategy = GenerationType.AUTO)
   private Long id;

   @Version @Column(nullable = false)
   private Long version;

   public PersistentObjectImpl() {
      uuid = UUID.randomUUID().toString();
   }

   /* get/set methods, equals(), hashCode(), toString() */
}
```

Domain: POJOs e JPA

- Replaced Entity Beans in Java EE 5;
- Simple objects + annotations;
- New features in Java EE 6:
  - Mapping of collection of non-entity objects;
  - New JPQL operators: **`NULLIF`**, **`COALESCE`**, **`INDEX`**, **`TYPE`**, **`KEY`**, **`VALUE`**, **`ENTRY`**;
  - Criteria API;
  - Pessimistic locking support;
  - Integrated with Bean Validation.

Implementing the 1st domain class

- Right-click in the EJB project > *New* > *Entity Class...*



Demonstration

Implement the classes `Employee` and `EmergencyCall` using JPA. New features: `orphanRemoval` for the association with `Dispatch` and `@ElementCollection` for the functions attribute.

Bean Validation

- Transversal validation: from the form in the Web page to the persistence database;
- Centered in the domain layer, but without loosing focus of its purpose – annotations;
- Based on Hibernate Validator;

```
public class Ambulance extends PersistentObjectImpl {
   @NotNull
   private int number;

   @NotNull
   @Size(min = 8, max = 8)
   private String licensePlate;

   /* ... */
}
```

Some validation annotations

- `@AssertFalse`, `@AssertTrue` (for *boolean*);
- `@DecimalMax`, `@DecimalMin` (for real numbers, but works only with `BigDecimal`);
- `@Max`, `@Min` (for integer numbers);
- `@Digits` (only digits, string OK, can specify min/max digits of integer and decimal parts);
- `@Future`, `@Past` (for dates);
- `@Pattern` (regular expressions).

Customized validation (1)

```
import javax.validation.*;

public class PlateValidator implements
                          ConstraintValidator<Plate, String> {
  public void initialize(Plate constraintAnnotation) { }

  public boolean isValid(String value,
                          ConstraintValidatorContext context) {
    if (value.length() != 8) return false;
    boolean ascending = true;
    int previous = Character.digit(value.charAt(4), 10);
    for (int i = 5; ascending && i < 8; i++) {
      int current = Character.digit(value.charAt(i), 10);
      ascending = current > previous;
      previous = current;
    }
    return ascending;
  }
}
```

Customized validation (2)

```
import static java.lang.annotation.ElementType.*;
import java.lang.annotation.*;
import javax.validation.*;
import javax.validation.constraints.*;

@NotNull
@Pattern(regexp = "^[A-Z]{3} [0-9]{4}$")
@Constraint(validatedBy = PlateValidator.class)
@Documented
@Target({ANNOTATION_TYPE, METHOD, FIELD})
@Retention(RetentionPolicy.RUNTIME)
public @interface Plate {
  String message() default "Invalid license plate";
  String[] groups() default {};
  Class<? extends Payload>[] payload() default {};
}
```

```
public class Ambulance extends PersistentObjectImpl {
  @Plate
  private String licensePlate;

  /* ... */
```

Contexts and dependency injection

- Contexts and Dependency Injection for the Java EE Platform (CDI) – JSR 299;
- Annotations to define the context:
  - `@ApplicationScoped`, `@ConversationScoped`, `@SessionScoped`, `@RequestScoped`, `@Dependent`;
- Annotations to inject components:
  - `@PersistenceContext`, `@EJB`, `@Resource`, `@Inject`, etc.
- To reference components in JSF pages: `@Named`;
- All managed by the container (no setter needed)!

Stereotypes

- Combination of multiple annotations:

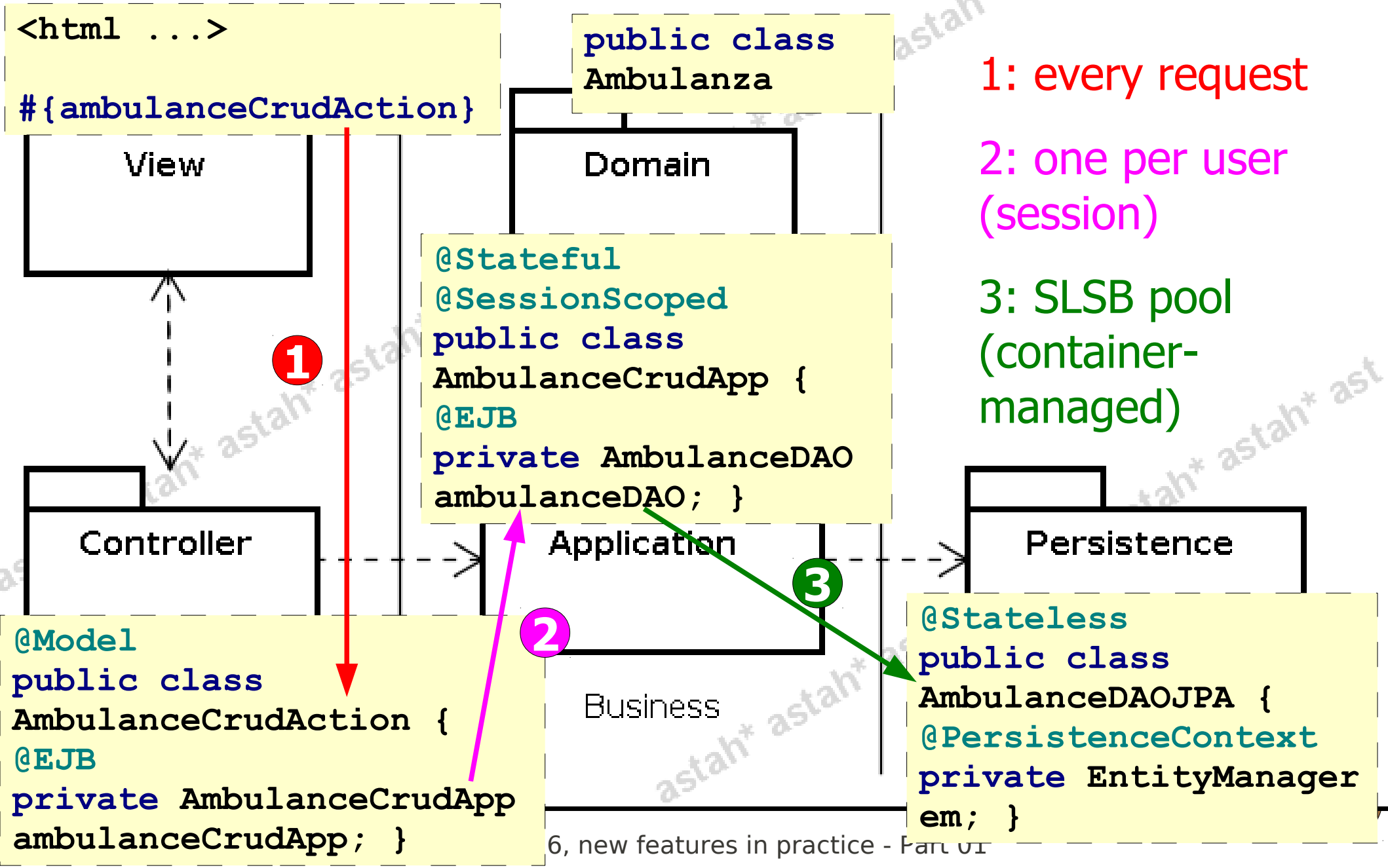
```
@Named
@RequestScoped
@Documented
@Stereotype
@Target(value={TYPE,METHOD,FIELD})
@Retention(value=RUNTIME)
public @interface Model
```

@Model = @Named + @RequestScoped

JSF Managed Beans

- Before, we had to declare them in the configuration file `faces-config.xml`;
- Now, all we need is `@Named`;
- Ambiguous annotations in JSF:
  - `@ManagedBean`, `@RequestScoped`, `@...Scoped`;
  - Also accepts CDI annotations, so we use those everywhere;
- It's also possible to access an EJB directly from the JSF page (would that be better?).

Example: Ambulance CRUD



Small persistence framework (2)

- Practically generated by NetBeans:
  - File > New File... > Java EE > Session Beans for Entity Classes;

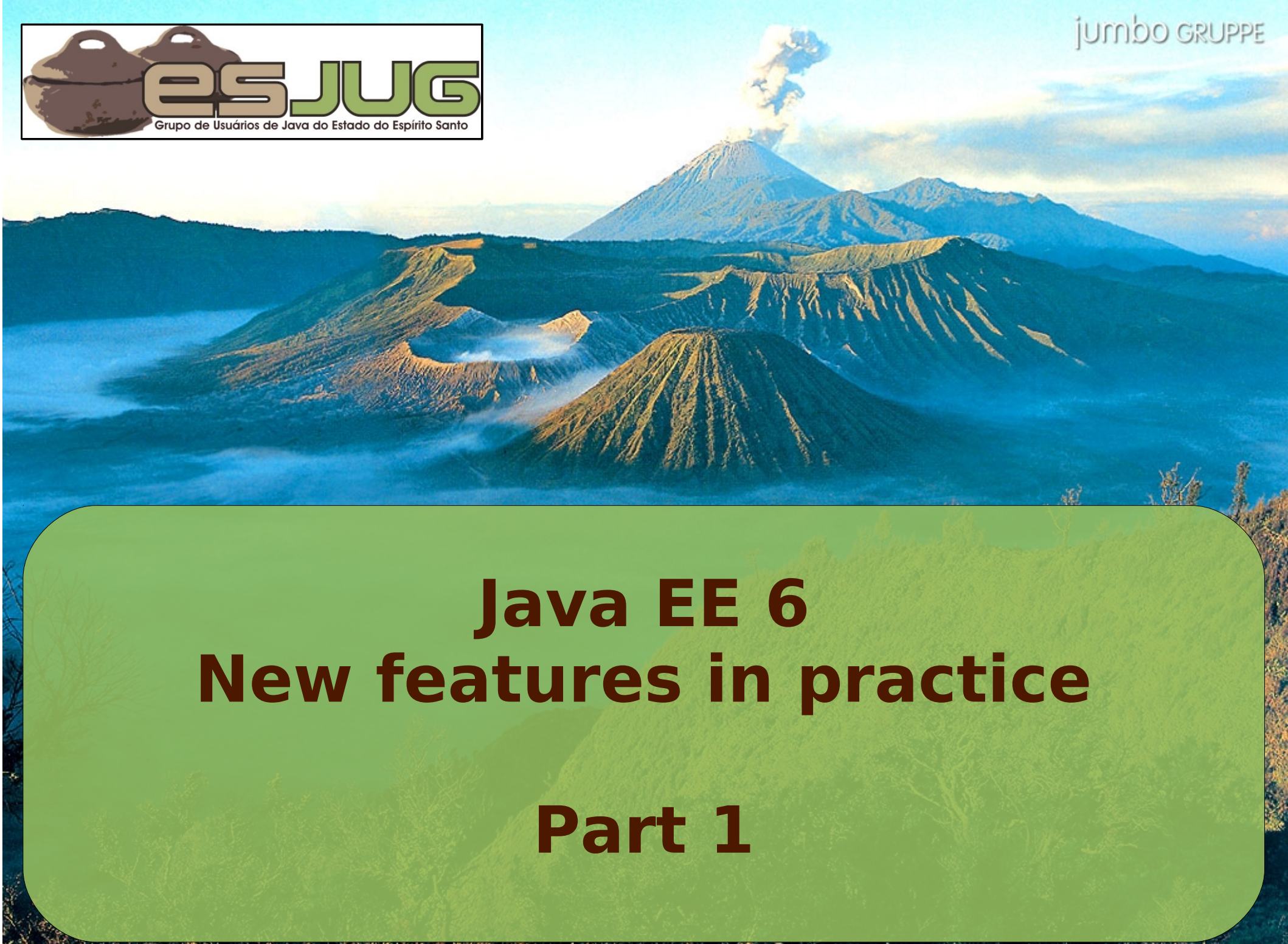
```
public abstract class BaseDAOJPA2<T extends
    PersistentObject> implements BaseDAO<T>, Serializable {

  protected abstract EntityManager getEntityManager();
  protected abstract Class<T> getDomainClass();

  public long retrieveCount() { /* ... */ }
  public List<T> retrieveAll() { /* ... */ }
  public List<T> retrieveSome(int[] range) {/* ... */}
  public T retrieveById(Long id) { /* ... */ }
  public void save(T object) { /* ... */ }
  public void delete(T object) { /* ... */ }
}
```

Demonstration

Implement the Ambulance CRUD, showing the dependency injection in contexts with CDI and the validation of the ambulance license plate.

Conclusions

- End of part 1. We've seen:
  - Develop with Java EE 6 is simpler than previous versions: less XML, more annotations;
  - Bean Validation guarantees data integrity;
  - Improved JPA 2, but there's more to be seen;
- In part 2:
  - Facelets replaces JSP as the default for JSF;
  - Criteria API (JPA 2) as alternative to JPQL;
  - Conversation management;
  - AJAX support in JSF 2.



Java EE 6
New features in practice

Part 1