Spring: ORM

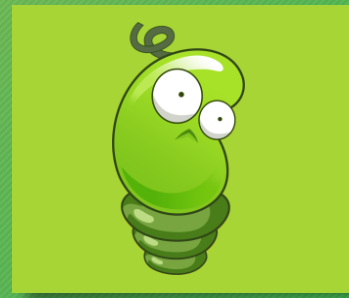
Peter Alagna Jr.

Hibernate Integration

Spring allows you to configure Hibernate to be used with **Contextual Sessions**.

- This **sessions** are created by **Hibernate** to manage users **concurrency**.
- A user **should not** be able to access another user's context.
- A **SessionFactory bean** is needed in order to get **sessions**.

Beans Needed



- **DataSource**.
  - **JDBC** instance.
    - Driver, Username, Password, URL.
- **SessionFactory**.
  - **Hibernate** instance.
  - Has data source as **member**.
  - Has the Hibernate configuration.
    - Packages to Scan, Dialect, Show_Sql, *hbm2ddl.auto*, *etc*.
- **TransactionManager**.
  - **Hibernate** instance.
  - Has session factory as a **member**.
  - Used and managed by Spring.
  - Used to associate a **contextual session** with a **transaction**.

REVATURE

@Transactional

```
<tx:annotation-driven/>
```

This annotation tells spring that a **class** (every method in it) or a specific **method** is going to be handled as a **Transaction**.

- **Contextual Sessions** should be seen as **Transactions**.
  - If tasks different than select are performed.
- This means we don't have to manually commit or rollback anymore.
  - Spring handles this for us adding behavior with **AOP**.

REVATURE

Exception Handling

By default, transactions are **rollbacked** by Spring if a **Runtime**, **unchecked** exception gets thrown.

- **HibernateException** is a **RuntimeException.**
  - We don't need try/catch blocks anymore.
- Exception is going to be automatically logged if **Log4j** is set up.
- To rollback with a specific **checked** exception:
  - @Transactional(rollbackFor = SpecificCheckedException.class).

REVATURE

JSR-303

- Standard API that Spring supports for **Bean** validation.
- We can use the **Validator** interface.
  - Providing validations to our business objects.
- It can be combined with **data binding** to dynamically bound user input to our model.

```
public class PersonValidator implements Validator {
```

```
public boolean supports(Class clazz) {
    return Person.class.equals(clazz);
}
```

```
public void validate(Object obj, Errors e) {
    ValidationUtils.rejectIfEmpty(e, "name", "name.empty");
    Person p = (Person) obj;
    if (p.getAge() < 0) {
        e.rejectValue("age", "negativevalue");
    } else if (p.getAge() > 110) {
        e.rejectValue("age", "too.darn.old");
    }
```

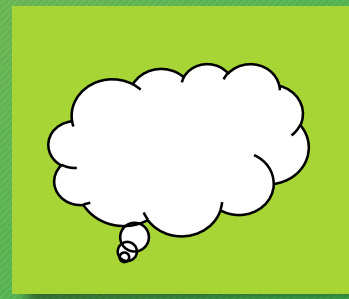
REVATURE

JndiObjectFactoryBean

- **JNDI** (Java Naming and Directory Interface).
  - Allows Java software clients to discover and look up data and objects via a **name**.
- **JndiObjectFactoryBean**.
  - Available in Spring's abstraction API.
  - Adds a layer of abstraction when obtaining external resources.
  - We can obtain a **container-managed** session factory.

REVATURE

Materials



- Spring ORM: https://docs.spring.io/spring/docs/current/spring-framework-reference/html/orm.html
- JSR-303: https://docs.spring.io/spring/docs/current/spring-framework-reference/html/validation.html
- JndiObjectFactoryBean: https://docs.spring.io/spring/docs/current/javadoc-api/org/springframework/jndi/JndiObjectFactoryBean.html