Validace

Bean Validation vs. JSF validation

- V současnosti existují dva způsoby, jak provést validaci:
  - Bean Validation (JSR 303)
    - Obecný způsob jak validovat jakýkoli JavaBean objekt. Není vůbec závislé na JSF nebo Java EE, je dokonce možné používat tuto validaci v Java SE aplikacích.
    - Validační logika je definována pomocí anotací, existuje pár hotových, je možné si vytvářet vlastní anotace.
    - Nejpoužívanější implementace JSR 303 je Hibernate Validator: http://hibernate.org/validator/
    - Velice vhodné používat, zejména pokud Vaše aplikace ukládá data do databáze (validace se poté provádí například i před vyvoláním SQL INSERT).
    - Validace je součást logiky aplikace a té by mělo být v UI co nejméně. Validace se provádí na serveru.
  - JSF validation: obdobné jako Bean Validation (výsledek je stejný), ale validace je pouze na úrovni UI.

Bean Validation: Anotace

- V JavaBean se validační anotace definují následovně:

```
public class Item {

  @NotNull(message = "Cannot be empty!")

  private String name;


  private String description;


  @Min(value = 1, message = "Price must not be zero!")

  private double price;

}
```

Dostupné anotace

- Standardně jsou k dispozici tyto anotace:
  - http://docs.oracle.com/javaee/6/tutorial/doc/gircz.html
- Hibernate Validator jich má ještě víc:
  - https://docs.jboss.org/hibernate/validator/5.1/api/org/hibernate/validator/constraints/package-summary.html

pom.xml

- Pokud nepoužíváte aplikační server, pak přidejte do `pom.xml`:

```
<dependency>
   <groupId>org.hibernate</groupId>
   <artifactId>hibernate-validator</artifactId>
   <version>5.0.1.Final</version>
</dependency>
```

Bean Validation & JSF

- Validace formuláře se v JSF provede automaticky. Pro zobrazení výsledku validace příslušného atributu je nutné definovat atribut `id`, který použijete v tagu `h:message`:

```
<h:message for="name" style="display:block;color:red"/>

<h:inputText value="#{itemManager.item.name}" title="name:" id="name" />
```

Tohle by bylo lepší přes CSS ;-)

- Tímto tagem zobrazíte seznam všech chyb:

```
<h:messages />
```

Validace & PrimeFaces

- Místo `h:message` použijete `p:message`, místo `h:messages` použijete `p:message`, jinak je to prakticky stejné (jenom lepší) :-)
- Praktický příklad viz. přednáška „navigace“.
- Místo nebo společně s `p:message` můžete použít `p:growl`:

```
<p:growl autoUpdate="true" showDetail="true" life="2000" />
```

  - http://www.primefaces.org/showcase/ui/growl.jsf
  - http://stackoverflow.com/questions/15598192/how-to-use-pgrowl-only-for-confirmation-not-validation-jsf2-primefaces

Místo prázdného textu NULL

- Pokud ve formuláři v HTML komponentách (jako je textfield, textarea apod.) nevyplníte žádnou hodnotu, pak se do atributů beany, která je s ním svázaná, uloží prázdný text.
- Jak toto změnit? Přidejte do `web.xml`:

```
<context-param>
   <param-name>javax.faces.INTERPRET_EMPTY_STRING_SUBMITTED_VALUES_AS_NULL</param-name>
   <param-value>true</param-value>
</context-param>
```

Formát datumu

- Tímto způsobem nastavíte formát datumu:

```
<p:inputText value="#{userOrderManager.userOrder.orderDate}">
   <f:convertDateTime pattern="dd.MM.yyyy"/>
</p:inputText>
```