Dominik Chlebicki
4AHITM

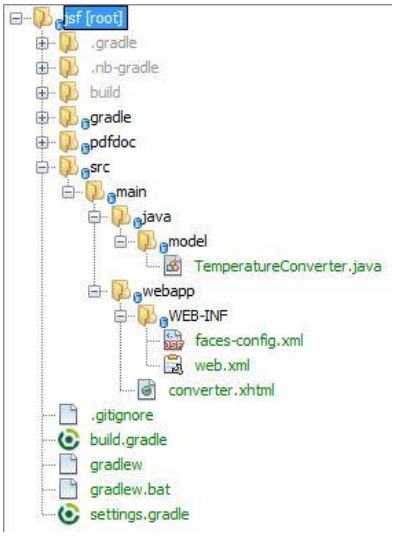
JSF Temp Converter

Protokoll

Aufgabe

Für den Anfang wurde das alte "hello_jsf"-Projekt kopiert und umbenannt. Anschließend konnte das Projekt in NetBeans (mit installierten Gradle-Plugin) importiert werden.
Files wie "hello.xhtml" werden nicht mehr verwendet und können deshalb gelöscht werden.

Danach werden die neuen Dateien für den Converter erstellt, dazu gehören "converter.xhtml" (die View) und "TemperatureConverter.java" (das Model).



Ordner/Projektstruktur

Dann wurde "web.xml" bearbeitet. Der Wert des <display-name>-Tags wurde zu "Temperature Converter" geändert. In das <welcome-file>-Tag wurde die Datei "converter.xhtml" geschrieben. Der Rest bleibt gleich.

In der Datei "faces-config.xml" stehen alle Daten über die Managed Beans. Diese werden ebenfalls geändert.

```
<?xml version="1.0" encoding="UTF-8"?>
<faces-config xmlns="http://xmlns.jcp.org/xml/ns/javaee"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
    http://xmlns.jcp.org/xml/ns/javaee/web-facesconfig_2_2.xsd"
    version="2.2">
    <managed-bean>
        <managed-bean-name>temperatureConverter</managed-bean-name>
        <managed-bean-class>model.TemperatureConverter</managed-bean-class>
        <managed-bean-scope>session</managed-bean-scope>
    </managed-bean>
</faces-config>
```
Managed Bean Konfiguration

Anschließend wird in "converter.xhtml" die ganze Struktur aufgebaut:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml" xmlns:h="http://java.sun.com/jsf/html"
      xmlns:f="http://java.sun.com/jsf/core" xmlns:ui="http://java.sun.com/jsf/facelets"
      xmlns:p="http://primefaces.org/ui">
    <h:head>
        <title>Temperature Converter</title>
    </h:head>
    <h:body>
        <h:form id="form">
            <p:panel header="Temperature Converter">
                <p:panelGrid columns="2">
                    <h:outputText value="Temp: " />
                    <p:inputText value="#{temperatureConverter.convert}" />
                </p:panelGrid>
                <br />
                <p:commandButton value="C -> F" ajax="false"
                                 action="#{temperatureConverter.celsiusToFahrenheit}" />
                <p:commandButton value="F -> C" ajax="false"
                                 action="#{temperatureConverter.fahrenheitToCelsius}" />
                <p:commandButton value="Reset" ajax="false"
                                 action="#{temperatureConverter.reset}" />
            </p:panel>
        </h:form>
        <p:panel id="result" header="Result"
                 rendered="#{not temperatureConverter.initial}">
            <h:outputLabel value="#{temperatureConverter.unit}: "></h:outputLabel>
            <h:outputLabel value="#{temperatureConverter.converted}"></h:outputLabel>
        </p:panel>
    </h:body>
</html>
```

Die View

Danach können im Model ("TemperatureConverter.java") die nötigen Funktionen erstellt werden:

```
package model;

import java.io.Serializable;

public class TemperatureConverter implements Serializable{
    private static final long serialVersionUID = 1L;
    private double convert;
    private double converted;
    private boolean initial;
    private String unit;

    public double getConvert() {
        return convert;
    }

    public void setConvert(double convert) {
        this.convert = convert;
    }

    public double getConverted() {
        return converted;
    }

    public String getUnit() {
        return unit;
    }

    public boolean getInitial() {
        return initial;
    }

    public void init(){
        initial = true;
        converted = 0;
        convert = 0;
        unit="";
    }

    public String reset() {
        init();
        return "reset";
    }

    public void celsiusToFahrenheit() {
        this.initial = false;
        this.unit="Fahrenheit";
        this.converted = (convert * 1.8) + 32;
    }

    public void fahrenheitToCelsius() {
        this.initial = false;
        this.unit = "Celsius";
        this.converted = (convert - 32) / 1.8;
    }
}
```

Das Model

Danach wird das Projekt mit "gradle jettyrun" in der Konsole gebuildet und ausgeführt, anschließend kann über den Browser darauf zugegriffen werden.

Erweiterung

Um neue Buttons hinzuzufügen müssen neue <p:commandButton> erstellt werden, jeweils mit der richtigen Beschriftung und Convert-Funktion:

```
<p:commandButton value="C -> F" ajax="false"
                 action="#{temperatureConverter.celsiusToFahrenheit}" />
<p:commandButton value="C -> K" ajax="false"
                 action="#{temperatureConverter.celsiusToKelvin}" />
<p:commandButton value="F -> C" ajax="false"
                 action="#{temperatureConverter.fahrenheitToCelsius}" />
<p:commandButton value="F -> K" ajax="false"
                 action="#{temperatureConverter.fahrenheitToKelvin}" />
<p:commandButton value="K -> C" ajax="false"
                 action="#{temperatureConverter.kelvinToCelsius}" />
<p:commandButton value="K -> F" ajax="false"
                 action="#{temperatureConverter.kelvinToFahrenheit}" />
<p:commandButton value="Reset" ajax="false"
                 action="#{temperatureConverter.reset}" />
```

converter.xml

Die Funktionen müssen auch der TemperatureConverter-Klasse zugefügt werden:

```
    public void celsiusToKelvin(){
        this.initial = false;
        this.unit = "Kelvin";
        this.converted = convert + 273.15;
    }

    public void fahrenheitToCelsius() {
        this.initial = false;
        this.unit = "Celsius";
        this.converted = (convert - 32) / 1.8;
    }

    public void fahrenheitToKelvin(){
        this.initial = false;
        this.unit = "Kelvin";
        this.converted = ((convert - 32) / 1.8) + 273.15;
    }

    public void kelvinToCelsius(){
        this.initial = false;
        this.unit = "Celsius";
        if(convert <= 0) {
            this.converted = -273.15;
        } else {
            this.converted = convert - 273.15;
        }
    }

    public void kelvinToFahrenheit(){
        this.initial = false;
        this.unit = "Fahrenheit";
        if(convert <= 0) {
            this.converted = -459.67;
        } else {
            this.converted = convert * 9 / 5 - 458.67;
        }
    }
```

TemperatureConverter.java

Anschließend kann das Projekt wieder mit "gradle jettyrun" gebuildet und ausgeführt werden.

Das Endergebnis

Dieses Projekt auf GitHub:
https://github.com/dchlebicki/tgm-aufgaben/tree/master/15-16/SEW/tempconv