Adap CW#12 Homework

Project name: Urlaub

Project repository: https://github.com/PVahldiek/wahlzeit

This week's tag: adap-hw12 on main

This week's diff: https://github.com/PVahldiek/wahlzeit/compare/adap-hw11...PVahldiek:adap-hw12

Frage: Wie stehen Objekt-Collaborations in Beziehung zu Design Patterns:

Collaborations sind meiner Meinung nach eine "Erweiterung" von Design Patterns. Design Patterns geben Klassenbeziehungen an und verfolgen einen bestimmten Zweck (z.B. Objekte instanziieren in Fabriken). Collaborations hingegen erweitern die Möglichkeiten. Bei Design Patterns werden Klassenbeziehungen dargestellt, in Collaborations können Beziehungen verschiedener Objekte (aus z.B. der gleichen Klasse) modelliert werden. Einzelne Teilnehmer (Rollen) können nun zum Beispiel auch auf Objektebene modelliert werden.

HolidayPhoto Holiday Collaboration

Zweck: Die Collaboration ist zuständig für das Sicherstellen der gesamten Domänenfunktionalität. Dazu gehört unter anderem:

- Ändern von Objektattributen über Setter Methoden
- Anzeigen von Objektattributen über Getter Methoden
- Sicherstellen, dass Objekte in sich konsistent sind (über Assertions)

Rollentypen: HolidayPhoto(Client), Holiday(Service); **Maintenance Collaboration**. Syntax:

```
public collaboration HolidayPhotoHoliday{
	public role HolidayPhoto{
		public Holiday getHoliday();
		public void setHoliday(Holiday h);
	}
	public role Holiday{
		public int getDays();
		public void setDays(int newDays);
		public int getCosts();
		public void setCosts(int newCosts);
		public String getCountry();
		public void setCountry(String newCountry);
		protected void assertClassInvatiants();
	}
}
public class HolidayPhoto binds HolidayPhotoHoliday.Holiday{...}
```

Holiday HolidayType Collaboration:

Zweck: Die Collaboration ermöglicht eine Hierarchie aus HolidayTypes über Holiday-Objekten zur Laufzeit. Dafür wurde das Type-Object Pattern verwendet.

- Erstellen mehrerer HolidayTypes für ein Holiday
- Kreieren eines Holiday-Objekts aus einem HolidayType (Client notwendig).

Rollentypen: Client, Holiday(Base Object); **Secondary Service Collaboration. Syntax:**

```
public collaboration HolidayHolidayType{
	public role HolidayType{
		public Holiday createInstance();
		public void setHoliday(Holiday h);
		public boolean hasInstance(Holiday holiday);
	}
	public role Holiday{
		public Holiday(HolidayType holidayType);
		public HolidayType getType();
	}
}
public class Client binds HolidayHolidayType.HolidayType{...}
public class Client binds HolidayHolidayType.Holiday{...}
```

Holiday DataObject Collaboration:

Zweck: Enthält Persistenz-Logiken für die Holiday-Objekte

- Lesen und Schreiben von Holiday-Objekten in die Datenbanken (über readFrom(), writeOn())

Rollentypen: Client, DataObject(Parent), Holiday(Child); **Primary Service Collaboration. Syntax:**

```
public collaboration HolidayDataObject{
	public role Holiday{
		public void readFrom(ResultSet rset);
		public void writeOn(ResultSet rset);
		public void writeId(PreparedStatement stmt, int pos);
	}
	public role DataObject implements Persistent{
		public final boolean resetWriteCount();
		public final void incWriteCount();
	}
}
public class Client binds HolidayDataObject.Holiday{...}
```