Arbeitsblatt 3 (27.10.2015)
GWT

In dieser Übung lernen Sie die Programmierung mit GWT kennen:
- ✓ Wiederholen Sie den Import von Eclipse-Projekten.
- ✓ Lernen Sie die Programmierung einer Webanwendung mit GWT.
- ✓ Lernen Sie das Ausführen einer Webanwendung mit GWT.

Für dieses Arbeitsblatt benötigen Sie ein vorkonfiguriertes Projekt, welches eine kleine GWT-Webanwendung enthält. Die Webanwendung enthält eine Klasse `Book`, welche eine einfache Repräsentation eines Buches mit den notwendigen Eigenschaften darstellt. Zudem enthält die Webanwendung die Klasse `BookExample`, die Sie im Verlauf dieses Arbeitsblattes zum Anzeigen von `Books` erweitern werden. Das Projekt „03-gwt-book-example.zip“ finden Sie in Moodle. Für die Bearbeitung dieser Aufgabe benötigen Sie GWT. Eine Installationsanleitung von GWT finden Sie in Moodle („03-Installation-Erste-Schritte-GWT“).

Aufgabe 1: Projekt importieren

Starten Sie zunächst Eclipse und importieren Sie das bereitgestellte Projekt. Machen Sie sich mit dem Inhalt des Projektes vertraut.

1. **Eclipse starten**
   - Starten Sie die Entwicklungsumgebung Eclipse.
2. **Wechseln in die Java Perspective**
   - Falls Sie sich noch nicht in der Java Perspective befinden sollten, wechseln Sie bitte in diese Perspective.
   - Eclipse Menu „Window“ -> „Open Perspective“ -> „Other“ -> „Java (default)“
3. **Importieren Sie das vorkonfigurierten Projekts**
   - Eclipse Menu „File“ -> „Import“
   - Projekt auswählen und importieren.
4. **Projektstruktur**
   - Machen Sie sich mit der Struktur des Projekts vertraut und schauen Sie sich die Klassen `Book.java` und `BookExample.java` im Ordner `src` im package `de.uhd.bookexample.client` an.

Das Projekt hat 2 Ordner: `src` und `war`. Im Ordner `src` enthält den Quelltext der Webanwendung. Der Ordner `war` enthält alle Dateien, die zur Ausführung der Webanwendung benötigt werden, z.B. die HTML-Seite `book-example.html`.

In den folgenden Schritten werden Sie schrittweise eine Webanwendung erstellen und ausführen.

Aufgabe 2: Implementierung verschiedener Widgets

Die Klasse `BookExample` stellt den Einstiegspunkt für Ihre Webanwendung dar. Das ist daran zu erkennen, dass die Klasse die Schnittstelle `EntryPoint` implementiert und eine Implementierung der Operation `onModuleLoad()` bereitstellen muss. Die Operation `onModuleLoad()` entspricht der `main()` Operation einer herkömmlichen Java-Anwendung.

Zunächst implementieren Sie zwei Panels, ein HTML-Widget, ein Label, eine TextBox und einen Button. Fügen Sie dazu den folgenden Quelltext in die `onModuleLoad()` Operation der Klasse `BookExample` ein.

```
// Zugriff auf <td id="content"></td> in der HTML-Datei um darin alle weiteren
// Inhalte einfuegen zu koennen
RootPanel rootPanel = RootPanel.get("content");

// Zuerst legen wir ein neues Panel an. In das Panel legen wir unsere
// Inhalte ab. Es gibt unterschiedliche Panels, z.B. vertikal und horizontal.
VerticalPanel vp = new VerticalPanel();

// Wir wollen die Ueberschrift 'BookExample' hinzufuegen. Dazu legen wir
// ein Widget vom Typ HTML an. In solch ein Widget lassen sich herkoemmliche
// HTML-Tags verwenden, z.B. <h1>.
HTML weppageHeader = new HTML("<h1>BookExample</h1>");

// Die Ueberschrift wird zum Panel hinzugefuegt.
vp.add(weppageHeader);

// Erstellung eines horizontalen Panels.
HorizontalPanel hp = new HorizontalPanel();

// Erzeugung eines Texts (Label) und hinzufuegen des Labels zum horizontalen Panel.
Label newBookName = new Label("Bookname:");
hp.add(newBookName);

// Um den Namen eines Buches einzugeben, brauchen wir eine TextBox.
// Die TextBox wird wieder zum horizontalen Panel hinzugefuegt.
TextBox bookName = new TextBox();
hp.add(bookName);

// Um ein neues Buch einzufuegen brauchen wir einen Button.
// Den Button fuegen wir wieder zum horizontalen Panel hinzu.
Button addBook = new Button("Add");
hp.add(addBook);

// Das horizontale Panel fuegen wir zu dem vertikalen Panel hinzu.
vp.add(hp);

// Wir fuegen das Panel zum RootPanel hinzu
rootPanel.add(vp);
```

Aufgabe 3: Hinzufügen einer Tabelle

In dieser Aufgabe werden Sie eine Tabelle zur Anzeige von Büchern implementieren. Fügen Sie dazu den folgenden Quelltext wieder in die `onModuleLoad()` Operation der Klasse `BookExample` ein. Achten Sie darauf, dass dieser Quelltext nach dem Quelltext aus Aufgabe 2 hinzugefügt wird.

```
// Erstellen einer Liste mit Buechern
ArrayList<Book> bookList = new ArrayList<Book>();
bookList.add(new Book(1234, "Krieg und Frieden", "Leo Tolstoi"));
bookList.add(new Book(4567, "Die Tore der Welt", "Ken Follet"));
bookList.add(new Book(6789, "Illuminati", "Dan Brown"));

// Diese Liste wollen wir in einer Tabelle (CellTable) anzeigen.
CellTable<Book> bookTable = new CellTable<Book>();
vp.add(bookTable);

// Anlegen der Spalten der Tabelle
// Zuerst die Spalte fuer die ISBN
// Info: Eine TextCell ist nicht bearbeitbar!
Column<Book, String> isbnColumn = new Column<Book, String>(new TextCell()) {

	@Override
	public String getValue(Book object) {
		// Hier muss der Wert zurueckgegeben werden, der
		// in der Spalte der Tabelle angezeigt werden soll.
		return "" + object.getIsbn();
	}

};

// Die ISBN Spalte zu der Tabelle hinzufuegen.
bookTable.addColumn(isbnColumn, "ISBN");

// Anlegen einer Spalte fuer den Titel
// Info: In einer EditTextCell kann man den Text bearbeiten!
Column<Book, String> titleColumn = new Column<Book, String>(new EditTextCell()) {

	@Override
	public String getValue(Book object) {
		return object.getTitle();
	}
};

// Auch diese Spalte wieder der Tabelle hinzufuegen.
bookTable.addColumn(titleColumn, "Title");

// Anlegen einer Spalte fuer den Author
Column<Book, String> authorColumn = new Column<Book, String>(new EditTextCell()) {

	@Override
	public String getValue(Book object) {
		return object.getAuthor();
	}
};

// Auch die Spalte wieder der Tabelle hinzufuegen.
bookTable.addColumn(authorColumn, "Author");

// Um die oben erstellten Liste von Buechern in die Tabelle zuladen
// erstellen wir einen ListDataProvider. Dem ListDataProvider teilen wir mit welche
// Tabelle mit welchen Daten befuellt werden soll.
ListDataProvider<Book> bookDataProvider = new ListDataProvider<Book>();
bookDataProvider.addDataDisplay(bookTable);
bookDataProvider.setList(bookList);
```

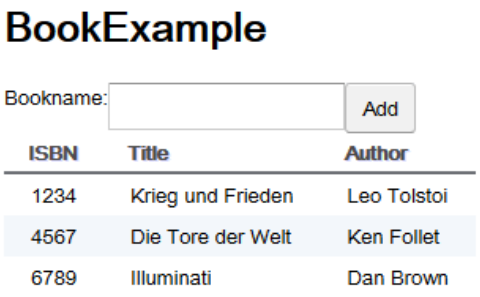
Aufgabe 4: Sortieren der Spalten einer Tabelle

In dieser Aufgabe lernen Sie, wie Sie die Inhalte der einzelnen Spalten sortieren können. Fügen Sie dazu den folgenden Quelltext wieder in die `onModuleLoad()` Operation der Klasse `BookExample` ein. Achten Sie darauf, dass der Quelltext unter dem Quelltext aus Aufgabe 3 hinzugefügt wird.

```
// Um die Inhalte sortieren zu koennen, muessen die Spalten sortierbar sein.
isbnColumn.setSortable(true);
titleColumn.setSortable(true);
authorColumn.setSortable(true);

// Erzeugung eines Handler, der den Event zum Sortierens abarbeiten kann.
// Der Handler bekommt die Liste des DataProviders uebergeben.
// Den Handler an die Tabelle anfuegen.
ListHandler<Book> sortHandler = new ListHandler<Book>(bookDataProvider.getList());
bookTable.addColumnSortHandler(sortHandler);

// Um die Sortierung durchzufuehren muss der Handler die Buecher vergleichen.
// Dazu verwendet man einen Comparator, der die ISBN zweier Buecher vergleicht.
sortHandler.setComparator(isbnColumn, new Comparator<Book>() {

	@Override
	public int compare(Book o1, Book o2) {
		// Vergleich der ISBN Nummern
		if (o1.getIsbn() > o2.getIsbn()) {
			return 1;
		}
		if (o1.getIsbn() < o2.getIsbn()) {
			return -1;
		}
		return 0;
	}
});

// Verwendung eines Comparator, der die Titel zweier Buecher vergleicht.
sortHandler.setComparator(titleColumn, new Comparator<Book>() {

	@Override
	public int compare(Book o1, Book o2) {
		// Vergleich der Titel zweier Buecher.
		// Der Datentyp String bietet bereits eine Vergleichsoperation.
		return o1.getTitle().compareTo(o2.getTitle());
	}
});

// Verwendung eines Comparators, der die Authoren zweier Buecher vergleicht.
sortHandler.setComparator(authorColumn, new Comparator<Book>() {

	@Override
	public int compare(Book o1, Book o2) {
		// Vergleich der Authoren zweier Buecher.
		return o1.getAuthor().compareTo(o2.getAuthor());
	}
});
```

Aufgabe 5: Webanwendung ausführen

Nachdem Sie Ihre Webanwendung implementiert haben, müssen Sie diese noch ausführen. Machen Sie einen Rechtsklick auf dem Projekt und wählen Sie „Debug As“ und dann „3 Web Application“ aus. Warten Sie bis in Eclipse die View „Development Mode“ geöffnet wird. Klicken Sie anschließend doppelt auf den dort abgebildeten Link, um die Webanwendung im Webbrowser anzusehen. Nach Öffnen des Link, kompiliert GWT die Klassen in dem Ordner „src“ von Java nach JavaScript und zeigt die resultierende Webanwendung an.



Damit haben Sie Ihre erste Webanwendung mit GWT implementiert!