# Exercise 1: Populating a Grid

In this exercise we practice using Vaadins data model, and more specifically, using DataProviders. The exercise stub contains a Layout that should contain a Grid which is populated by a DataProvider. Your task is to create a Grid and initialize a `ListDataProvider` containing four columns: name, email, age and birthday. You get a List of test data from the PersonService class.

| Name | Email | Age | Birthday |
|---|---|---|---|
| Elizabeth Smith | Elizabeth.Smith@example.com | 53 | 1964-11-21 |
| Dorothy White | Dorothy.White@example.com | 33 | 1984-08-05 |
| Mary Davis | Mary.Davis@example.com | 43 | 1974-04-19 |
| Dorothy Miller | Dorothy.Miller@example.com | 35 | 1982-06-07 |
| Linda Williams | Linda.Williams@example.com | 22 | 1995-09-21 |
| William Williams | William.Williams@example.com | 53 | 1964-02-03 |
| David Taylor | David.Taylor@example.com | 28 | 1989-01-22 |
| John Williams | John.Williams@example.com | 49 | 1968-03-03 |
| Dorothy Moore | Dorothy.Moore@example.com | 22 | 1995-10-28 |
| Elizabeth Jones | Elizabeth.Jones@example.com | 20 | 1997-01-05 |

You have a choice how to create the Columns in your Grid, between these two:

```
Grid<Person> grid = new Grid<>(Person.class)

// OR

Grid<Person> grid = new Grid<>();

grid.addColumn(Person::getAge);

...
```

Both options are equally valid, but remember that the first version might not order the columns like you want. There are ways to override that, of course.

**Links to Vaadin Docs:**
Grid: https://vaadin.com/docs8/-/part8/framework/components/components-grid.html
DataProvider: https://vaadin.com/docs8/-/part8/framework/datamodel/datamodel-providers.html

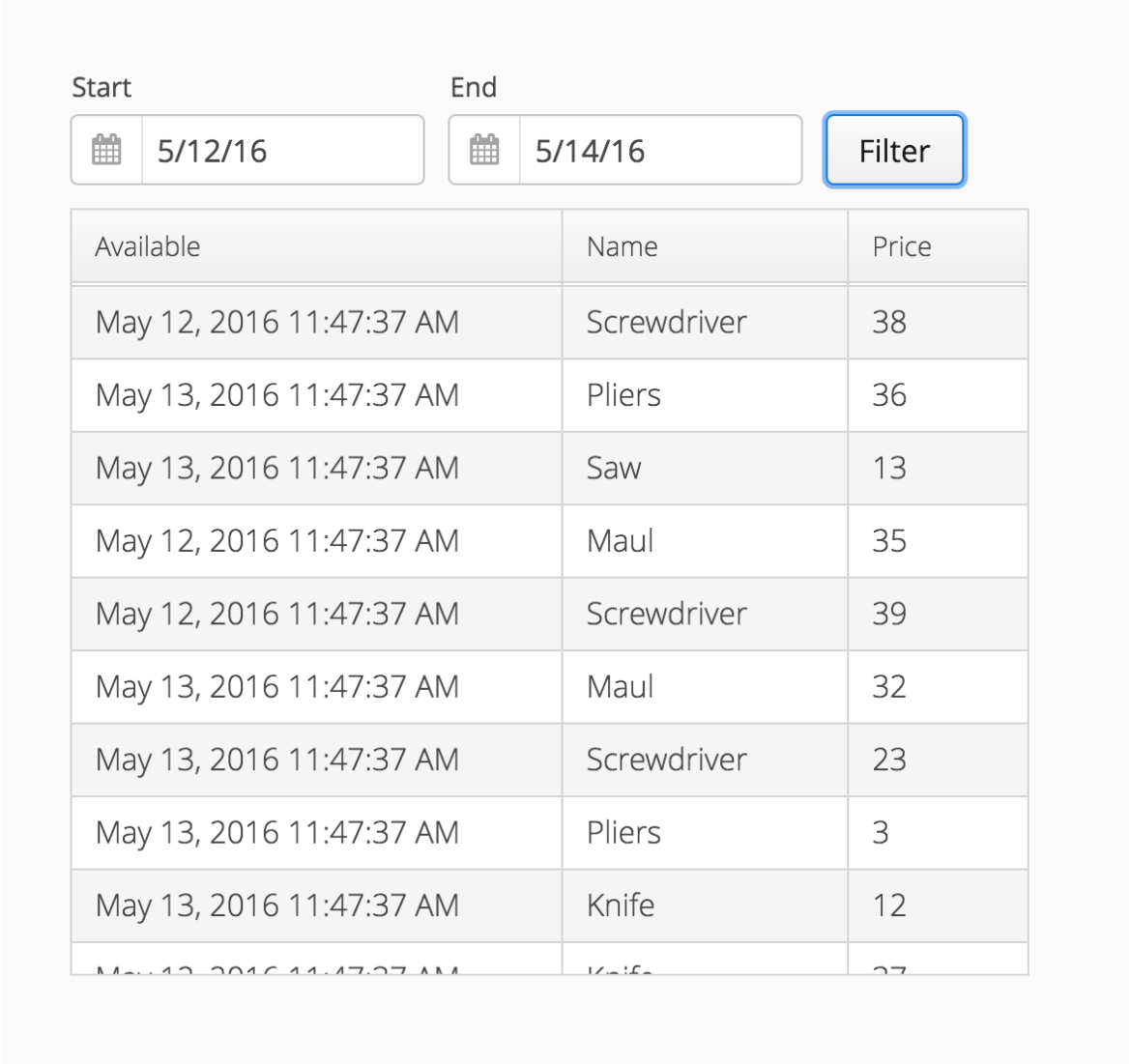
# Exercise 2: Filtering a DataProvider

The target of this exercise is to practice filtering values in an in-memory data provider. The view should have two `DateFields` in which you select a date range. When the filter-button is clicked, the grid's content should be filtered so, that only rows where the "available" property is between the given range are visible.

Note that filtering should be done in the DataProvider, not in the grid!

The layout for the view is not built for you, so you'll have to start by creating the view layout, it shouldn't be too hard for you at this point.

Once you've created the layout, start implementing the filtering with the help of the `filterProduct` method. As a hint, you should use `ListDataProvider.setFilter(SerializablePredicate<T> filter)`.



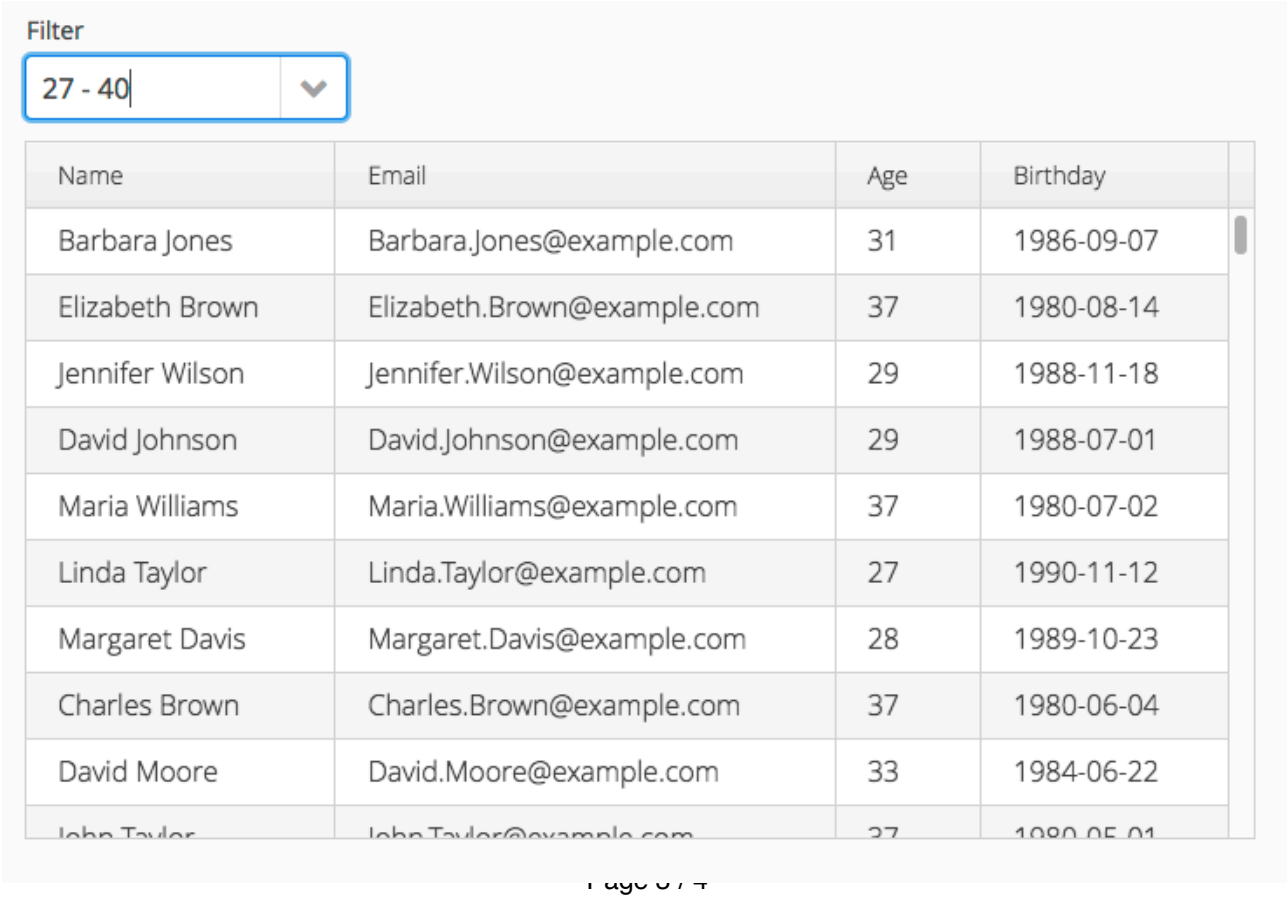

**REMEMBER** that your filter should be able to handle 4 situations; both dates being null, both dates being non-null, and either date being null and the other not. In other words, the

filter can be open-ended in either direction. You can handle each case, in turn, in the `filterProduct` method.

**Links to Vaadin Docs:**
Grid: https://vaadin.com/docs8/-/part8/framework/components/components-grid.html
DataProvider: https://vaadin.com/docs8/-/part8/framework/datamodel/datamodel-providers.html

# Exercise 3: Filtering a back end data provider

In the last exercise we will create a filtering BackEndDataProvider for a Grid. You are provided with a ComboBox, a Grid, and a Service for data. Your task is to create a DataProvider that fetches items lazily form the Service. In addition, the provided ComboBox should be able to filter the content of the Grid; when a user selects an age group, the grid should refresh itself with persons only from that group. This is done by providing a filter to the back end.



The lazy DataProvider can be done in the following steps:

1. Create the normal, non-filtering provider with `DataProvider.fromFilteringCallbacks()`. The first parameter is a call to `PersonService.getPersons()`, the second to `PersonService.countPersons()`. Save the resulting data provider to a `CallbackDataProvider` variable.

2. Add filtering with `CallbackDataProvider.withConfigurableFilter()`. You don't need any parameters for this one. Save the resulting `ConfigurableFilterDataProvider` to a variable as your actual data provider. It should have the following type:

   ```
   ConfigurableFilterDataProvider<Person, Void, AgeGroup>
   ```

3. call `grid.setDataProvider()` with your `ConfigurableFilterDataProvider`.

4. Add a ValueChangeListener to the ComboBox that gets the value from it and calls `dataProvider.setFilter(selectedAgeGroup)`

**Bonus:** If you clicked around in the solution, you might have noticed that columns do not sort properly anymore. This is because the Grid can't sort lazy data, that has to be one in the back end. As a bonus task, try to modify the Service class methods to support Sorting. The sort parameters can be found in the Query object.

**Links to Vaadin Docs:**
Grid: https://vaadin.com/docs8/-/part8/framework/components/components-grid.html
DataProvider: https://vaadin.com/docs8/-/part8/framework/datamodel/datamodel-providers.html