

Exercise 1: Polling

This exercise is simple enough to get familiar of using simple components and events in Vaadin, but also demonstrates the synchronous model.

The goal is to create an application that contains three buttons and a progress indicator. One button is for showing a notification with the text “Hello” in it. The two other buttons should start a long running operation on the server, which simulates a time consuming process on the server.

First of the buttons should start the operation synchronously, so that the operation is ran in the same thread. There is a helper method, `startOperation(boolean)`, that will actually execute the long running operation. The second button should start the same process, except that the process is ran asynchronously in a separate thread. Again, use the same helper method to start the process.

1. Create a Button instance for all the three buttons in the application
2. Set the width of the buttons to 170px so that they are equally wide
3. Add a Button.ClickListener to each of the Buttons (or you can use a lambda expression)
    a. In ClickListener, call `startOperation()` with either the parameter `true` or `false`
    b. When the notification button is clicked, open a notification with the text “Hello” using the Notification.show(“”) method.

Helpful links to Vaadin documentation

- Button
- Handling events with listeners
- Notifications

Exercise 2: Creating a Custom Component

Server-side composition of existing Vaadin components is a powerful way to create reusable custom components which can also be used in other projects and by other developers. Wrapping functionality into an extension of the CustomComponent class also provides a way to protect the implementation details while providing a meaningful API for your custom component.

Your task is to create a server-side composition containing a TextField and a Label. Please start with the `WrappedTextField` class which already extends the CustomComponent class.

1. Create an instance of TextField and Label and assign them to the fields provided in the WrappedTextField class
2. Create a new HorizontalLayout
3. Set the HorizontalLayout you created as the composition root of WrappedTextField
4. Add the TextField and Label you created in step 1 to the HorizontalLayout

Helpful links to Vaadin documentation

- Composition with CustomComponent
- HorizontalLayout