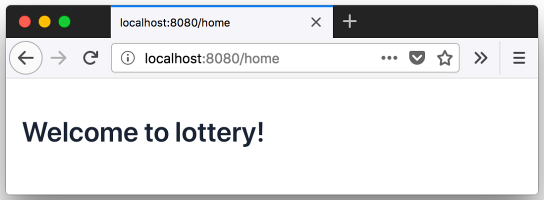
Exercises

In this exercise you are about to create a small lottery application. Even it's small, it has almost everything, two sub views HomeView and LotteryView, MainLayout, and even a LoginView. You will complete this exercise step by step.

Step1: Enable Routing

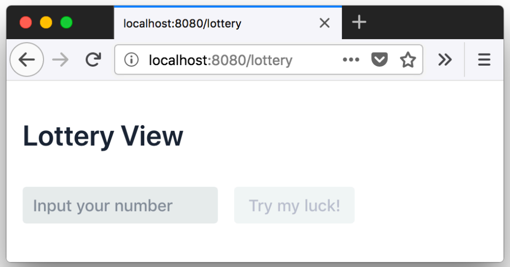
1. Find the HomeView class, you can see from the code that it extends from `Composite<Div>`, the content is quite simple, with only a `H2` element showing the text 'Welcome to Lottery!'.
2. Add annotation `@Route("home")` to HomeView
3. Now after you deploy the application, when you navigate to localhost:8080/home you should be able to see something like this:



localhost:8080/home

Welcome to lottery!

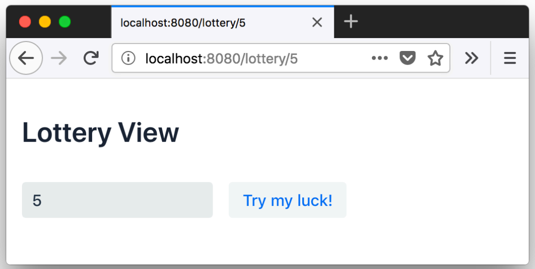
4. Do the same for LotteryView, so that when navigate to localhost:8080/lottery, you can see the lottery view.



localhost:8080/lottery

Lottery View

5. Add a route alias to the Home view, so that the home view can be shown when navigate to localhost:8080
6. Enable url parameters for Lottery View, so that when text field will be preset when navigating to lottery view with a parameter. e.g. when navigate to localhost:8080/lottery/5, you will see

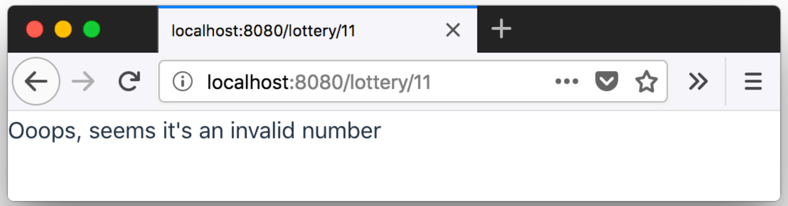


localhost:8080/lottery/5

Lottery View

5

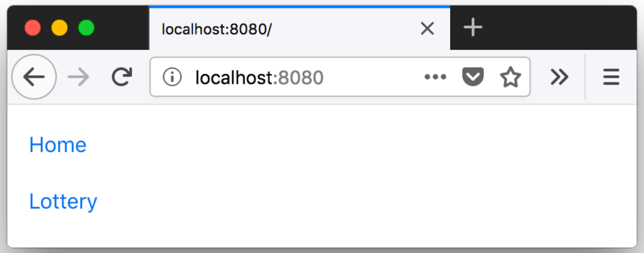
7. Handle errors during navigation. The valid number for lottery should be [1, 10], if user use an invalid number, then an exception will be thrown. Define an error view, so that when the number is invalid, the user will be redirected to the error view. The error can be triggered by calling the validate() to validate the parameter in setParameter method.



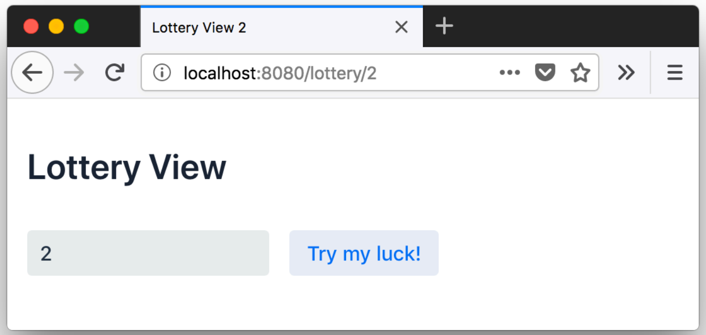
Ooops, seems it's an invalid number

Step 2: Add Navigation

1. Add a new MainView class to your application.
2. Add @Route("") to MainView class, so that by default it will go to this MainView when user navigates to localhost:8080. Note that you should also remove the @RouteAlias for HomeView, otherwise you will get an exception.
3. Add two router links to the MainView, one for HomeView, the other for LotteryView, so that user can navigate to corresponding view by clicking the link.

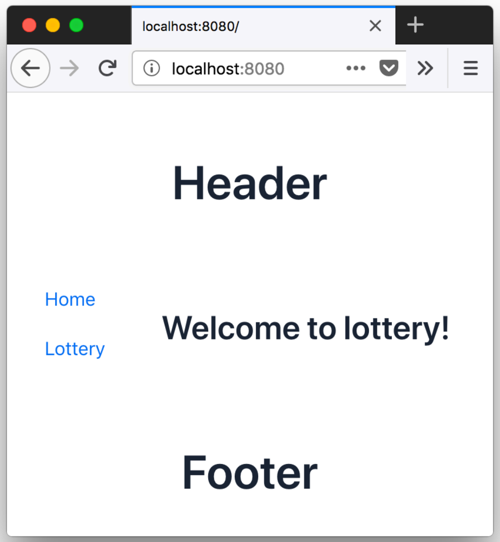


4. Add page titles to HomeView and LotteryView. The page title for HomeView is a static text “Home”, the page tile for LotteryView is dynamic, it’s “Lottery View” + the number that in the textfield, e.g. “Lottery View 2”



Step 3: Application Layout

1. Modify the MainView class, so that it will have a header, a footer, a menu bar and the child view. The header and footer could just be two H1 elements. A menu bar could be a vertical layout with 2 router links. Then a Div element which would be a wrapper for a child view.

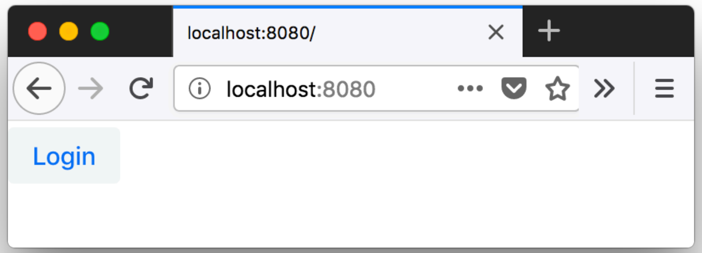


2. Make MainView class implements the RouterLayout interface. Implement the showRouterLayoutContent method by appending the content to the child wrapper element.
3. Change the @Route in HomeView and LotteryView class, to specify the MainView as their parent layouts.
4. Remove the @Route("") from MainView.
5. You can add the @RouteAlias("") back to HomeView, so that by default it will navigate to the home view, and home view will be displayed inside main view.

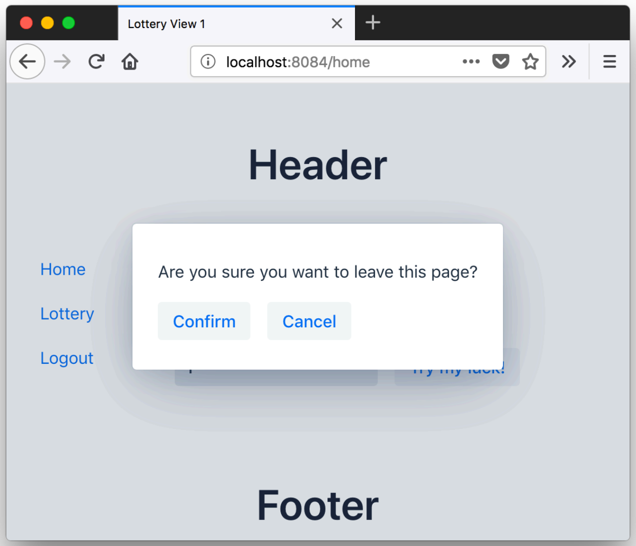
Step 4: Setup login

Let's add a login view to the application, so that the application will only visible to users who have already logged in, otherwise, it will redirect the user to the login view .

1. Create the `LoginView` class, which extends from `Composite<Div>` and implements `HasComponents`.
2. Add a button to `LoginView`, and inside the button click listener, set the `"userLoggedIn"` attribute to true in current Vaadin session, with `VaadinSession.getCurrent().setAttribute()`. Also navigate to home view with `UI.getCurrent().navigate("")`.
3. Let `MainView` implement the `BeforeEnterObserver` interface. In the `beforeEnter` method, check if the Vaadin session has the "`userLoggedIn"` attribute, and if not, take the user back to `LoginView` with `event.rerouteTo("login")`. So when the user is not logged in, they will be automatically redirected to login view.
4. Now when you go to localhost:8080, you get the login button, after you login, it will take you to home view:



5. Make the LotteryView implement the BeforeLeaveObserver, so that if user has input something in the text field, then trying to navigate to other view, a window pops up asking for user's confirmation, unless user confirms, it will stay in current view. Dialog component can be used for implementing the confirmation dialog. To open a dialog, call dialog.open(); to close a dialog, call dialog.close().



Bonus: logout

Setup a Logout view, which just shows some text and a link to take user to login view. In the constructor of the LogoutView, you can put the logout logic there, like destroy the http session with
`VaadinSession.getCurrent().getSession().invalidate();`

Also refresh the page with the following code, so that a new UI and new session will be created.
`UI.getCurrent().getPage().executeJavaScript("window.location.href=''");`