Vaadin資料

Vaadinのコンポーネント(一部)

詳しくはこのリンクで調べてください。
https://vaadin.com/docs/latest/components

レイアウトコンポーネント(Layouts)

VerticalLayout
レイアウトのコンポーネント
縦並びにしたいときにこれを使う。

```
var verticalLayout = new VerticalLayout();
```

HorizontalLayout
レイアウトのコンポーネント
横並びにしたいときにこれを使う。

```
var horizontalLayout = new HorizontalLayout();
```

FormLayout
レイアウトのコンポーネント
フォームのレイアウトはこれでやるといいかも

データ入力コンポーネント(Data Entry)

TextField・MailField・PasswordField
入力のためのコンポーネント
パスワードの入力はパスワードフィールド、メールアドレスの入力はメールフィールド、それ以外はテキストフィールドで大丈夫だと思う。

CheckBox
チェックボックスのコンポーネント
チェックボックス・ラジオボタンなど複数の要素を含むものは、大体~~Boxというクラスがあり、それで管理します。

```
var checkboxGroup = new CheckboxGroup<>();
checkboxGroup.setItems("A", "B", "C", "D", "E", "F", "G", "H");
var checkbox = new Checkbox( labelText: "a");
checkboxGroup.add(checkbox);
add(checkboxGroup);
```

データ表示・インタラクションコンポーネント(Visualization & Interaction)

Button
ボタンのためのコンポーネント
addClickListenerメソッドのリスナーでクリックに対しての操作を記述可能

Grid
グリッド表示のためのコンポーネント

標準HTMLコンポーネント

https://vaadin.com/docs/latest/flow/create-ui/standard-html

H1~H6
H1~H6タグに対応するコンポーネント

Paragraph
pタグに対応するコンポーネント

Anchor
aタグに対応するコンポーネント

Element APIを使用すると、HtmlContainerを使用して任意の標準HTML要素を作成できます

例

```
HtmlContainer input = new HtmlContainer( tagName: "input");
```

他にもHtmlクラスを用いれば、文字列でHTMLを書くことも可能です。

例

```
Html html = new Html( outerHtml: "<p>pタグの作成</p>");
```

Vaadinのコンポーネントの機能

Vaadinのコンポーネントはadd(Component… components)といったメソッドを持っており、これを使用してコンポーネントを追加できます。

また、コンポーネントのset~~のメソッドでHTMLの属性やCSSを変えることができます。

コンポーネントのメソッドにないCSSを変えたいときは、下記のようにスタイルにセットします。

```
toHorizontalButton.getStyle().setBorder("3px solid #000");
```

```
toHorizontalButton.getStyle().set("border", "3px solid #000");
```

※どちらも同じ操作になります。

イベントハンドリング

コンポーネントは対応するリスナーを持っていて、それの処理をラムダ式・無名関数などで記述可能です。
※ 今回の処理は、ボタンをクリックすると他のページに遷移する処理になっています。

```
toHorizontalButton.addClickListener(clickEvent ->
        UI.getCurrent().getPage().setLocation("horizontal")
);
toHorizontalButton.addClickListener(new ComponentEventListener<>() {
    @Override  0 件の使用箇所
    public void onComponentEvent(ClickEvent<Button> clickEvent) {
        UI.getCurrent().getPage().setLocation("horizontal");
    }
});
```

また、引数として受け取ったイベントはイベントごとに必要な情報を保持してくれています。

```
password.addValueChangeListener(valueChangeEvent -> {
    valueChangeEvent.getValue();
    valueChangeEvent.getOldValue();
});
```

valueChangeEventの場合、今の値と変わる前の値などを保持してくれる。

ページの作成方法

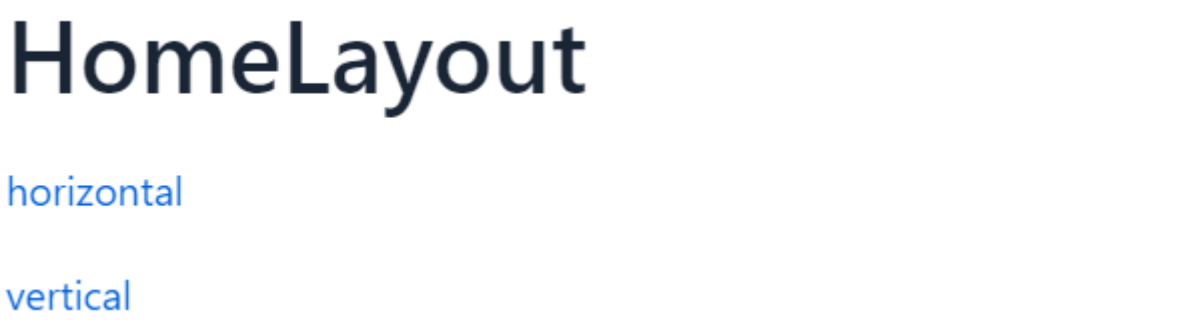
Vaadinのページを作るには、下記のようにVaadinのコンポーネントを継承して、それにRouteアノテーションをつければ、作ることができます。

```java
package com.example.demo.layout;

import com.vaadin.flow.component.html.Anchor;
import com.vaadin.flow.component.html.H1;
import com.vaadin.flow.component.orderedlayout.VerticalLayout;
import com.vaadin.flow.router.PageTitle;
import com.vaadin.flow.router.Route;

@PageTitle("Home")
@Route("")
public class HomeLayout extends VerticalLayout {
    public HomeLayout() {
        var layoutName = new H1( text: "HomeLayout");
        var toHorizontal = new Anchor( href: "horizontal", text: "horizontal");
        var toVertical = new Anchor( href: "vertical", text: "vertical");
        add(layoutName, toHorizontal, toVertical);
    }
}
```

※ PageTitleアノテーションはページを作るために必須ではないです。



次のようにH1などの大見出しでページを作ることもできますが、レイアウトコンポーネントのVerticalLayout、HorizontalLayoutを使った方が良いと思います。

```java
@PageTitle("Home")
@Route("")
public class HomeLayout extends H1 {
    public HomeLayout() {
        var layoutName = new H1( text: "HomeLayout");
        var toHorizontal = new Anchor( href: "horizontal",  text: "horizontal");
        var toVertical = new Anchor( href: "vertical",  text: "vertical");
        add(layoutName, toHorizontal, toVertical);
    }
}
```

ログイン機能

読んでください。
https://vaadin.com/docs/latest/flow/security/enabling-security