```
package fi.paatti.mobile.views.menuview;


import com.vaadin.ui.Component;
import com.vaadin.ui.NativeButton;
import fi.paatti.mobile.applicationinfo.CssStyleNames;
import fi.paatti.mobile.mobileview.MobileViewLogic;
import fi.paatti.mobile.views.mobilecontentview.MobileContentView;
import fi.paatti.mobile.paattiapplication.MobileViewHandler;
import java.util.HashMap;


/**
 * A view for a menu in mobile ui.
 * <p/>
 * @author Lauri Satokangas, lauri.n.satokangas@student.jyu.fi @date 10.3.2012
 * @since 0.0.1
 */
public abstract class MenuView extends MobileContentView {

  private MenuViewListener menuViewListener;
  private MenuViewLogic menuViewLogic;
  private final HashMap<Object, Component> menuComponentMap;
  private NativeButton topButton;
  private NativeButton centerButton;
  private NativeButton bottomButton;
  private final NativeButton[] buttonArray = new NativeButton[3];

  /**
   * First button in the menu.
   */
  public static final String MENU_BUTTON_TOP = "menutopbutton";

  /**
   * Second button in the menu.
   */
  public static final String MENU_BUTTON_CENTER = "menucenterbutton";

  /**
   * third button in the menu.
   */
  public static final String MENU_BUTTON_BOTTOM = "menubottombutton";
  private MobileViewHandler viewhandler;

  /**
   * Constructor.
   * <p/>
   * Creates a menu view with default layout. Contains a header label, three
   * menu buttons and three footer buttons.
   * <p/>
   * @param handler The applications view handler.
   */
  public MenuView(MobileViewHandler handler) {
    this.menuComponentMap = new HashMap<Object, Component>();
    viewhandler = handler;
    createUi();

  }

  /**
   * Constructor.
   * <p/>
   * Creates a menu view with default layout. Contains a header label, three
   * menu buttons and three footer buttons. Texts of the header and buttons
   * can be given as parameters.
   * <p/>
   * @param viewHandler  Handler for the view.
   * @param headerText   Text for the header
   * @param topButton    Text for the top button
   * @param centerButton Text for the center button
   * @param bottomButton Text for the bottom button
   * <p/>
   */
  public MenuView(MobileViewHandler viewHandler, String headerText,
      String topButton, String centerButton, String bottomButton) {
    this(viewHandler);
    getViewHandler().setHeaderText(headerText);
    this.topButton.setCaption(topButton);
    this.centerButton.setCaption(centerButton);
    this.bottomButton.setCaption(bottomButton);

  }

  /**
   * Create the user interface.
   */
  private void createUi() {

    setSizeFull();
    this.setSpacing(true);
    this.setMargin(true);
    this.setStyleName(CssStyleNames.MENU_LAYOUT);
```

```java
    topButton = new NativeButton();
    topButton.setStyleName(CssStyleNames.MENU_TOP_BUTTON);
    buttonArray[0] = topButton;

    centerButton = new NativeButton();
    centerButton.setStyleName(CssStyleNames.MENU_CENTER_BUTTON);
    buttonArray[1] = centerButton;

    bottomButton = new NativeButton();
    bottomButton.setStyleName(CssStyleNames.MENU_BOTTOM_BUTTON);
    buttonArray[2] = bottomButton;

    menuComponentMap.put(MENU_BUTTON_TOP, topButton);
    menuComponentMap.put(MENU_BUTTON_CENTER, centerButton);
    menuComponentMap.put(MENU_BUTTON_BOTTOM, bottomButton);

    for (NativeButton button : buttonArray) {
      button.addStyleName(CssStyleNames.MENU_BUTTON);
      button.setSizeFull();
      this.addComponent(button);
      this.setExpandRatio(button, 1f);
    }

  }

  /**
   * Set the text of the menu button.
   * <p/>
   * @param menuButton On which button the text will be set
   * @param text       The new text of the button
   */
  public void setMenuButtonText(Object menuButton, String text) {
    if (menuButton.equals(MENU_BUTTON_TOP)) {
      topButton.setCaption(text);
    } else if (menuButton.equals(MENU_BUTTON_CENTER)) {
      centerButton.setCaption(text);
    } else if (menuButton.equals(MENU_BUTTON_BOTTOM)) {
      bottomButton.setCaption(text);
    }
  }

  /**
   * {@inheritDoc}
   */
  @Override
  public void setContentViewLogic(MobileViewLogic logic) {
    viewhandler.setMobileViewLogic(logic);
    this.menuViewLogic = (MenuViewLogic) logic;

    if (this.menuViewListener != null) {
      for (NativeButton b : buttonArray) {
        for (Object listener : b.getListeners(NativeButton.ClickEvent.class)) {
          b.removeListener((NativeButton.ClickListener) listener);
        }

        b.removeListener(menuViewListener);
      }
    }

    this.menuViewListener = new MenuViewListener(menuViewLogic);

    for (NativeButton b : buttonArray) {
      b.addListener((NativeButton.ClickListener) menuViewListener);
    }
  }

  /**
   * Return the menu buttons.
   * <p/>
   * @return
   */
  public HashMap<Object, Component> getMenuComponents() {
    return menuComponentMap;
  }

  /**
   * @return
   */
  public MobileViewHandler getViewHandler() {
    return viewhandler;
  }
}
```