```java
package fi.paatti.mobile.mobileview;


import com.vaadin.terminal.ThemeResource;
import com.vaadin.ui.Alignment;
import com.vaadin.ui.Button;
import com.vaadin.ui.Component;
import com.vaadin.ui.ComponentContainer;
import com.vaadin.ui.Embedded;
import com.vaadin.ui.HorizontalLayout;
import com.vaadin.ui.Label;
import com.vaadin.ui.NativeButton;
import com.vaadin.ui.Panel;
import com.vaadin.ui.VerticalLayout;
import com.vaadin.ui.themes.Runo;
import fi.paatti.mobile.applicationinfo.CssStyleNames;
import java.util.HashMap;


/**
 * Layout for all the mobile views.
 * <p/>
 * Has a place for header, footer and the content.
 * <p/>
 * @author Lauri Satokangas, lauri.n.satokangas@student.jyu.fi @date 6.3.2012
 */
public class MobileView extends VerticalLayout implements MobileViewButtons, MobileViewHeaderAndFooterActions {

  private static final long serialVersionUID = 1L;
  private Panel pageContent;
  private Label headerLabel;
  private NativeButton leftFooterButton;
  private NativeButton middleFooterButton;
  private NativeButton rightFooterButton;

  /*
   * All the footer buttons in an array. Can be used when performing the same
   * action to all the buttons.
   */
  private final NativeButton[] footerButtonArray = new NativeButton[3];
  private HorizontalLayout headerLayout;
  private HorizontalLayout footerLayout;
  private MobileViewListener mobileViewListener;
  private final HashMap<Object, Component> footerComponents;
  // These heights are used when calculating the right size for the whole layout.
  private final int DEFAULT_HEIGHT_OF_HEADER = 48;
  private final int DEFAULT_HEIGHT_OF_FOOTER = 70;
  private int viewHeight = -1;
  private MobileViewLogic mobileViewLogic;
  private Embedded headerIcon;

  /**
   * Constructor.
   */
  public MobileView() {
    footerComponents = new HashMap<Object, Component>();
    createLayout();
  }

  /**
   * Creates the layout and sets all the settings etc.
   */
  private void createLayout() {
    this.setWidth("100%");

    this.setStyleName(CssStyleNames.MOBILE_VIEW);
    HorizontalLayout header = new HorizontalLayout();

    this.addComponent(header);

    // Create the header
    headerLayout = new HorizontalLayout();

    headerIcon = new Embedded();
    headerIcon.setStyleName("headericon");
    headerIcon.setSizeUndefined();

    headerLabel = new Label();
    headerLabel.setStyleName(CssStyleNames.HEADER_TEXT);
    headerLabel.setContentMode(Label.CONTENT_XHTML);
    headerLabel.setSizeUndefined();

    headerLayout.addComponent(headerIcon);
    headerLayout.addComponent(headerLabel);
    headerLayout.setComponentAlignment(headerLabel, Alignment.MIDDLE_CENTER);
    headerLayout.setExpandRatio(headerLabel, 1f);

    headerLayout.setSizeUndefined();
    this.setHeader(headerLayout);
    this.setComponentAlignment(headerLayout, Alignment.MIDDLE_CENTER);

    // Create the content
    pageContent = new Panel();
    pageContent.addStyleName(Runo.PANEL_LIGHT);
    pageContent.addStyleName(CssStyleNames.PAGE_CONTENT);

    this.addComponent(pageContent, 1);
    this.setExpandRatio(pageContent, 1f);

    // HorizontalLayout footer = new HorizontalLayout();
    // this.addComponent(footer);

    // Create the footer
    footerLayout = new HorizontalLayout();
    footerLayout.setSpacing(true);
    footerLayout.setMargin(true);
    footerLayout.setStyleName(CssStyleNames.FOOTER_LAYOUT);

    leftFooterButton = new NativeButton();
    leftFooterButton.setStyleName(CssStyleNames.LEFT_FOOTER_BUTTON);

    footerComponents.put(FOOTER_BUTTON_LEFT, leftFooterButton);
    footerButtonArray[0] = leftFooterButton;

    middleFooterButton = new NativeButton();
    middleFooterButton.setStyleName(CssStyleNames.MIDDLE_FOOTER_BUTTON);
    footerComponents.put(FOOTER_BUTTON_MIDDLE, middleFooterButton);
    footerButtonArray[1] = middleFooterButton;

    rightFooterButton = new NativeButton();
    rightFooterButton.setStyleName(CssStyleNames.RIGHT_FOOTER_BUTTON);
    footerComponents.put(FOOTER_BUTTON_RIGHT, rightFooterButton);
    footerButtonArray[2] = rightFooterButton;

    // Settings that are same for all the footer buttons.
    // Also add all the buttons to the footer layout.
    for (NativeButton button : footerButtonArray) {
      button.addStyleName(CssStyleNames.FOOTER_BUTTON);
      button.setDisableOnClick(true);
      button.setSizeFull();
      footerLayout.addComponent(button);
      footerLayout.setExpandRatio(button, 1f);
    }

    this.setFooter(footerLayout);
  }

  /**
```

```
   * Set the main content.
   * <p/>
   * @param contentToBeSet Needs to be ComponentContainer because Panel
   * doesn´t accept single Components.
   */
  public void setContent(ComponentContainer contentToBeSet) {
    pageContent.setContent(contentToBeSet);
  }

  /**
   * Set the header component.
   * <p/>
   * @param headerToBeSet
   */
  private void setHeader(Component headerToBeSet) {
    this.removeComponent(getComponent(0));
    headerToBeSet.setHeight(DEFAULT_HEIGHT_OF_HEADER + "px");
    headerToBeSet.setWidth("100%");
    headerToBeSet.setStyleName(CssStyleNames.HEADER_LAYOUT);
    this.addComponent(headerToBeSet, 0);
    this.setExpandRatio(headerToBeSet, 0);
  }

  /**
   * Set the footer component.
   * <p/>
   * @param footerToBeSet
   */
  private void setFooter(Component footerToBeSet) {
    if (this.getComponentCount() == 3) {
      this.removeComponent(getComponent(2));
    }
    footerToBeSet.setHeight(DEFAULT_HEIGHT_OF_FOOTER + "px");
    footerToBeSet.setWidth("100%");
    footerToBeSet.setStyleName(CssStyleNames.FOOTER_LAYOUT);
    this.addComponent(footerToBeSet, 2);
    this.setExpandRatio(footerToBeSet, 0);
  }

  /**
   * Set the height of the layout (which should be the height of the mobile browser).
   * <p/>
   * @param height The new height
   */
  public void setViewHeight(int height) {
    this.viewHeight = height;
    pageContent.setHeight(height - DEFAULT_HEIGHT_OF_HEADER - DEFAULT_HEIGHT_OF_FOOTER + "px");
  }

  /**
   * Get the component in the footer.
   * <p/>
   * @return
   */
  public HashMap<Object, Component> getFooterComponents() {
    return footerComponents;
  }

  /**
   * Set the logic that handles the footer button clicks.
   * <p/>
   * @param logic The logic to be set.
   */
  public void setViewLogic(MobileViewLogic logic) {

    if (mobileViewListener != null) {
      for (NativeButton b : footerButtonArray) {
        for (Object listener : b.getListeners(Button.ClickEvent.class)) {
          b.removeListener((Button.ClickListener) listener);
        }

        b.removeListener(mobileViewListener);
      }
    }
    mobileViewLogic = logic;
    mobileViewListener = new MobileViewListener(mobileViewLogic);
    for (NativeButton b : footerButtonArray) {
      b.addListener((Button.ClickListener) mobileViewListener);
    }

  }

  /**
   * {@inheritDoc}
   */
  @Override
  public void setHeaderText(String text) {
    headerLabel.setValue(text);
  }

  /**
   * {@inheritDoc}
   */
  @Override
  public void setHeaderIcon(ThemeResource iconResource) {
    headerIcon.setSource(iconResource);
  }

  /**
   * {@inheritDoc}
   */
  @Override
  public void setFooterButtonText(Object footerButton, String text) {

    if (footerButton.equals(FOOTER_BUTTON_LEFT)) {
      leftFooterButton.setCaption(text);
    } else if (footerButton.equals(FOOTER_BUTTON_MIDDLE)) {
      middleFooterButton.setCaption(text);
    } else if (footerButton.equals(FOOTER_BUTTON_RIGHT)) {
      rightFooterButton.setCaption(text);
    }

  }

  /**
   * {@inheritDoc}
   */
  @Override
  public void setFooterButtonEnabled(Object footerButton, boolean enabled) {

    if (footerButton.equals(FOOTER_BUTTON_LEFT)) {
      leftFooterButton.setEnabled(enabled);
    } else if (footerButton.equals(FOOTER_BUTTON_MIDDLE)) {
      middleFooterButton.setEnabled(enabled);
    } else if (footerButton.equals(FOOTER_BUTTON_RIGHT)) {
      rightFooterButton.setEnabled(enabled);
    }
  }

  @Override
  public void setFooterButtonPressed(Object footerButton, boolean pressed) {
    NativeButton button = null;

    if (footerButton.equals(FOOTER_BUTTON_LEFT)) {
      button = leftFooterButton;
    } else if (footerButton.equals(FOOTER_BUTTON_MIDDLE)) {
      button = middleFooterButton;
    } else if (footerButton.equals(FOOTER_BUTTON_RIGHT)) {
      button = rightFooterButton;
    }

    if (pressed) {
      button.addStyleName(CssStyleNames.BUTTON_PRESSED);
    } else {
      button.removeStyleName(CssStyleNames.BUTTON_PRESSED);
    }
  }

  /**
   * {@inheritDoc}
   */
  @Override
  public void setFooterButtonIcon(Object footerButton, ThemeResource iconResource) {

    if (footerButton.equals(FOOTER_BUTTON_LEFT)) {
      leftFooterButton.setIcon(iconResource);
    } else if (footerButton.equals(FOOTER_BUTTON_MIDDLE)) {
      middleFooterButton.setIcon(iconResource);
    } else if (footerButton.equals(FOOTER_BUTTON_RIGHT)) {
      rightFooterButton.setIcon(iconResource);
    }
  }

  /**
   * {@inheritDoc}
   */
  @Override
  public void resetFooterButtons() {
    for (NativeButton button : footerButtonArray) {
      button.setCaption(null);
      button.setIcon(null);
      button.removeStyleName(CssStyleNames.BUTTON_PRESSED);
    }
    setFooterButtonsEnabled(true);
    setFooterButtonsVisible(true);
  }

  /**
   * {@inheritDoc}
   */
  @Override
  public void setFooterButtonsEnabled(boolean enabled) {
    for (NativeButton button : footerButtonArray) {
      button.setEnabled(enabled);
```

```java
    }
  }

  /**
   * {@inheritDoc}
   */
  @Override
  public void setFooterButtonsVisible(boolean visible) {
    footerLayout.setVisible(visible);
    footerLayout.setEnabled(visible);

    if (visible) {
      setViewHeight(viewHeight);
    } else {
      pageContent.setHeight(viewHeight - DEFAULT_HEIGHT_OF_HEADER + "px");
    }
  }
}
```