

Table of Contents

Customization of the standard application frame is already described [here](#). But the menu can also be customized without extending the application itself. The menu has some features which aren't available without extending it.

Let's change the menu a little bit:

- Remove the SAVE and RELOAD buttons
- Add a custom button to the button area
- Completely remove the button area

Before you can use your own menu implementation, set the [application parameter](#):

Application.Menu.corporation.classname to the full qualified class name of your menu implementation, e.g.

```
<servlet>
  <servlet-name>VaadinUI</servlet-name>
  <servlet-class>com.sibvisions.rad.ui.vaadin.server.VaadinServlet</servlet-
class>

  ...

  <init-param>
    <param-name>Application.Menu.corporation.classname</param-name>
    <param-value>com.sibvisions.apps.vaadin.CustomCorporationMenu</param-
value>
  </init-param>

  ...
</servlet>
```

It's super easy to remove some buttons:

```
public class CustomCorporationMenu extends WebMenuCorporation
{
  public CustomCorporationMenu(IApplication pApplication)
  {
    super(pApplication);
  }

  @Override
  protected void configureMenu()
  {
    super.configureMenu();

    removeItem(EDIT_SAVE);
    removeItem(EDIT_RELOAD);
  }
}
```

To add a custom button, you need a little bit more code, because the button should be visible if authenticated only:

```
public class CustomCorporationMenu extends WebMenuCorporation
{
    public static final String OPTION_CLOSEALL = "Option/CloseAll";

    private UIButton butCloseAll;

    public CustomCorporationMenu(IApplication pApplication)
    {
        super(pApplication);
    }

    @Override
    protected void addMenuAreaButtons(IContainer pArea)
    {
        butCloseAll = new UIButton();
        butCloseAll.setImage(UIImage.getImage(IFontAwesome.ARROW_CIRCLE_LEFT_SMALL))
        ;

        //same style as all other buttons
        Style.addStyleNames(butCloseAll, "topbutton", "topcloseall");

        //ADDS the CloseAll Button before all other buttons!
        pArea.add(butCloseAll);

        super.addMenuAreaButtons(pArea);
    }

    @Override
    protected void putComponents()
    {
        super.putComponents();

        put(OPTION_CLOSEALL, butCloseAll);
    }

    @Override
    public void createStandardMenu()
    {
        super.createStandardMenu();

        setItemVisible(OPTION_CLOSEALL, false);
    }

    @Override
    public void doAfterLogin(RemoteApplication pApplication)
    {
        super.doAfterLogin(pApplication);

        setItemVisible(OPTION_CLOSEALL, true);
    }
}
```

And not so tricky is the removal of the button area:

```
public class CustomCorporationMenu extends WebMenuCorporation
{
    public CustomCorporationMenu(IApplication pApplication)
    {
        super(pApplication);
    }

    @Override
    protected void updateMenuPosition(LayoutMode pMode)
    {
        super.updateMenuPosition(pMode);

        //removes the options panel from the header
        UIPanel panel = getHeaderOptionsPanel();

        if (panel != null)
        {
            IContainer parent = panel.getParent();

            if (parent != null)
            {
                panel.getParent().remove(panel);
            }
        }
    }
}
```

From:

<http://doc.sibvisions.com/> - **Documentation**

Permanent link:

http://doc.sibvisions.com/vaadin/customize_menu



Last update: **2020/08/05 13:02**