GWT - MENUBAR WIDGET

Introduction

The **MenuBar** widget represents a standard menu bar widget. A menu bar can contain any number of menu items, each of which can either fire a Command or open a cascaded menu bar.

Class declaration

Following is the declaration for **com.google.gwt.user.client.ui.MenuBar** class:

```
public class MenuBar
   extends Widget
      implements PopupListener, HasAnimation,
      HasCloseHandlers<PopupPanel>
```

CSS style rules

Following default CSS Style rules will be applied to all the MenuBar widget. You can override it as per your requirements.

```
.gwt-MenuBar {}
.gwt-MenuBar-horizontal {}
.gwt-MenuBar-vertical{}
.gwt-MenuBar .gwt-MenuItem {}
.gwt-MenuBar .gwt-MenuItem-selected {}
.gwt-MenuBar .gwt-MenuItemSeparator {}
.gwt-MenuBar .gwt-MenuItemSeparator .menuSeparatorInner {}
.gwt-MenuBarPopup .menuPopupTopLeft {}
.gwt-MenuBarPopup .menuPopupTopLeftInner {}
.gwt-MenuBarPopup .menuPopupTopCenter {}
.gwt-MenuBarPopup .menuPopupTopCenterInner {}
.gwt-MenuBarPopup .menuPopupTopRight {}
.gwt-MenuBarPopup .menuPopupTopRightInner {}
.gwt-MenuBarPopup .menuPopupMiddleLeft {}
.gwt-MenuBarPopup .menuPopupMiddleLeftInner {}
.gwt-MenuBarPopup .menuPopupMiddleCenter {}
.gwt-MenuBarPopup .menuPopupMiddleCenterInner {}
.gwt-MenuBarPopup .menuPopupMiddleRight {}
.gwt-MenuBarPopup .menuPopupMiddleRightInner {}
.gwt-MenuBarPopup .menuPopupBottomLeft {}
.gwt-MenuBarPopup .menuPopupBottomLeftInner {}
.gwt-MenuBarPopup .menuPopupBottomCenter {}
.gwt-MenuBarPopup .menuPopupBottomCenterInner {}
.gwt-MenuBarPopup .menuPopupBottomRight {}
.gwt-MenuBarPopup .menuPopupBottomRightInner {}
```

Class constructors

S.N.	Constructor & Description
1	**MenuBar** Creates an empty horizontal menu bar.
2	**MenuBar***booleanvertical* Creates an empty menu bar.

3

MenuBar*booleanvertical, MenuBar. MenuBarImagesimages*

Deprecated. replaced by MenuBar*boolean, Resources*

4

MenuBar*booleanvertical, MenuBar. Resourcesresources*

Creates an empty menu bar that uses the specified ClientBundle for menu images.

5

MenuBar*MenuBar. MenuBarImagesimages*

Deprecated. replaced by MenuBar*Resources*

6

MenuBar*MenuBar. Resourcesresources*

Creates an empty horizontal menu bar that uses the specified ClientBundle for menu images.

Class methods

S.N.	Function name & Description

1

HandlerRegistration addCloseHandler(CloseHandler handler)

Adds a CloseEvent handler.

2

MenuItem addItem*MenuItemitem*

Adds a menu item to the bar.

3

MenuItem addItem*SafeHtmlhtml, Commandcmd*

Adds a menu item to the bar containing SafeHtml, that will fire the given command when it is selected.

4

MenuItem addItem*SafeHtmlhtml, MenuBarpopup*

Adds a menu item to the bar, that will open the specified menu when it is selected.

5

MenuItem addItem*java. lang. Stringtext, booleanasHTML, Commandcmd*

Adds a menu item to the bar, that will fire the given command when it is selected.

6

MenuItem addItem*java. lang. Stringtext, booleanasHTML, MenuBarpopup*

Adds a menu item to the bar, that will open the specified menu when it is selected.

7

MenuItem addItem*java. lang. Stringtext, Commandcmd*

Adds a menu item to the bar, that will fire the given command when it is selected.

8

MenuItem addItem*java. lang. Stringtext, MenuBarpopup*

Adds a menu item to the bar, that will open the specified menu when it is selected.

9

MenuItemSeparator addSeparator

Adds a thin line to the MenuBar to separate sections of MenuItems.

10

MenuItemSeparator addSeparator*MenuItemSeparatorseparator*

Adds a thin line to the MenuBar to separate sections of MenuItems.

11

void clearItems

Removes all menu items from this menu bar.

12

void closeAllChildren*booleanfocus*

Closes this menu and all child menu popups.

13

void focus

Give this MenuBar focus.

14

boolean getAutoOpen

Gets whether this menu bar's child menus will open when the mouse is moved over it.

15

int getItemIndex*MenuItemitem*

Get the index of a MenuItem.

16

protected java.util.List getItems

Returns a list containing the MenuItem objects in the menu bar.

17

protected MenuItem getSelectedItem

Returns the MenuItem that is currently selected *highlighted* by the user.

18

int getSeparatorIndex*MenuItemSeparatoritem*

Get the index of a MenuItemSeparator.

19

MenuItem insertItem*MenuItemitem, intbeforeIndex*

Adds a menu item to the bar at a specific index.

20

MenuItemSeparator insertSeparator*intbeforeIndex*

Adds a thin line to the MenuBar to separate sections of MenuItems at the specified index.

21

MenuItemSeparator insertSeparator*MenuItemSeparatorseparator, intbeforeIndex*

Adds a thin line to the MenuBar to separate sections of MenuItems at the specified index.

22

boolean isAnimationEnabled

Returns true if animations are enabled, false if not.

23

boolean isFocusOnHoverEnabled

Check whether or not this widget will steal keyboard focus when the mouse hovers over it.

24

void moveSelectionDown

Moves the menu selection down to the next item.

25

void moveSelectionUp

Moves the menu selection up to the previous item.

26

void onBrowserEvent*Eventevent*

Fired whenever a browser event is received.

27

protected void onDetach

This method is called when a widget is detached from the browser's document.

28

protected void onEnsureDebugId*java. lang. StringbaseID*

Affected Elements: -item# = the MenuItem at the specified index.

29

void onPopupClosed*PopupPanelsender, booleanautoClosed*

Deprecated. Use addCloseHandler*CloseHandler* instead

30

void removeItem*MenuItemitem*

Removes the specified menu item from the bar.

31 **void removeSeparator***MenuItemSeparatorseparator*

Removes the specified MenuItemSeparator from the bar.

32 **void selectItem***MenuItemitem*

Select the given MenuItem, which must be a direct child of this MenuBar.

33 **void setAnimationEnabled***booleanenable*

Enable or disable animations.

34 **void setAutoOpen***booleanautoOpen*

Sets whether this menu bar's child menus will open when the mouse is moved over it.

35 **void setFocusOnHoverEnabled***booleanenabled*

Enable or disable auto focus when the mouse hovers over the MenuBar.

Methods inherited

This class inherits methods from the following classes:

- com.google.gwt.user.client.ui.UIObject
- com.google.gwt.user.client.ui.Widget
- java.lang.Object

MenuBar Widget Example

This example will take you through simple steps to show usage of a MenuBar Widget in GWT. Follow the following steps to update the GWT application we created in *GWT - Create Application* chapter:

Step	Description
1	Create a project with a name *HelloWorld* under a package *com.tutorialspoint* as explained in the *GWT - Create Application* chapter.
2	Modify *HelloWorld.gwt.xml*, *HelloWorld.css*, *HelloWorld.html* and *HelloWorld.java* as explained below. Keep rest of the files unchanged.
3	Compile and run the application to verify the result of the implemented logic.

Following is the content of the modified module descriptor **src/com.tutorialspoint/HelloWorld.gwt.xml**.

```
<?xml version="1.0" encoding="UTF-8"?>
<module rename-to='helloworld'>
  <!-- Inherit the core Web Toolkit stuff.                        -->
  <inherits name='com.google.gwt.user.User'/>
```

```
  <!-- Inherit the default GWT style sheet.                        -->
  <inherits name='com.google.gwt.user.theme.clean.Clean'/>

  <!-- Specify the app entry point class.                          -->
  <entry-point class='com.tutorialspoint.client.HelloWorld'/>

  <!-- Specify the paths for translatable code                     -->
  <source path='client'/>
  <source path='shared'/>

</module>
```

Following is the content of the modified Style Sheet file **war/HelloWorld.css**.

```
body{
   text-align: center;
   font-family: verdana, sans-serif;
}
h1{
   font-size: 2em;
   font-weight: bold;
   color: #777777;
   margin: 40px 0px 70px;
   text-align: center;
}

.gwt-MenuBar {
   cursor: default;
}
.gwt-MenuBar .gwt-MenuItem {
   cursor: default;
   font-family: Arial Unicode MS, Arial, sans-serif;
   font-size: 12px;
}
.gwt-MenuBar .gwt-MenuItem-selected {
   background: #E3E8F3;
}
.gwt-MenuBar-horizontal {
   background: #e3e8f3 url(images/hborder.png) repeat-x 0px -2003px;
   border: 1px solid #e0e0e0;
}
.gwt-MenuBar-horizontal .gwt-MenuItem {
   padding: 5px 10px;
   vertical-align: bottom;
   color: #000;
   font-weight: bold;
}
.gwt-MenuBar-horizontal .gwt-MenuItemSeparator {
   width: 1px;
   padding: 0px;
   margin: 0px;
   border: 0px;
   border-left: 1px solid #ccc;
   background: white;
}
.gwt-MenuBar-horizontal .gwt-MenuItemSeparator .menuSeparatorInner {
   width: 1px;
   height: 1px;
   background: white;
}
.gwt-MenuBar-vertical {
   margin-top: 0px;
   margin-left: 0px;
   background: white;
}
.gwt-MenuBar-vertical table {
   border-collapse: collapse;
}
```

```
.gwt-MenuBar-vertical .gwt-MenuItem {
   padding: 2px 40px 2px 1px;
}
.gwt-MenuBar-vertical .gwt-MenuItemSeparator {
   padding: 2px 0px;
}
.gwt-MenuBar-vertical .gwt-MenuItemSeparator .menuSeparatorInner {
   height: 1px;
   padding: 0px;
   border: 0px;
   border-top: 1px solid #ccc;
   overflow: hidden;
}
.gwt-MenuBar-vertical .subMenuIcon {
   padding-right: 4px;
}
.gwt-MenuBar-vertical .subMenuIcon-selected {
   background: #E3E8F3;
}
.gwt-MenuBarPopup {
   margin: 0px 0px 0px 3px;
}

.gwt-MenuBarPopup .menuPopupTopLeftInner {
   width: 5px;
   height: 5px;
   zoom: 1;
}
.gwt-MenuBarPopup .menuPopupTopRightInner {
   width: 8px;
   height: 5px;
   zoom: 1;
}
.gwt-MenuBarPopup .menuPopupBottomLeftInner {
   width: 5px;
   height: 8px;
   zoom: 1;
}
.gwt-MenuBarPopup .menuPopupBottomRightInner {
   width: 8px;
   height: 8px;
   zoom: 1;
}
.gwt-MenuBarPopup .menuPopupTopLeft {
   background: url(images/corner.png) no-repeat 0px -36px;
   -background: url(images/corner_ie6.png) no-repeat 0px -36px;
}
.gwt-MenuBarPopup .menuPopupTopRight {
   background: url(images/corner.png) no-repeat -5px -36px;
   -background: url(images/corner_ie6.png) no-repeat -5px -36px;
}
.gwt-MenuBarPopup .menuPopupBottomLeft {
   background: url(images/corner.png) no-repeat 0px -41px;
   -background: url(images/corner_ie6.png) no-repeat 0px -41px;
}
.gwt-MenuBarPopup .menuPopupBottomRight {
   background: url(images/corner.png) no-repeat -5px -41px;
   -background: url(images/corner_ie6.png) no-repeat -5px -41px;
}
html>body .gwt-MenuBarPopup {
}
* html .gwt-MenuBarPopup .menuPopupTopLeftInner {
   width: 5px;
   height: 5px;
   overflow: hidden;
}
* html .gwt-MenuBarPopup .menuPopupTopRightInner {
   width: 8px;
   height: 5px;
```

```
   overflow: hidden;
}
* html .gwt-MenuBarPopup .menuPopupBottomLeftInner {
   width: 5px;
   height: 8px;
   overflow: hidden;
}
* html .gwt-MenuBarPopup .menuPopupBottomRightInner {
   width: 8px;
   height: 8px;
   overflow: hidden;
}
```

Following is the content of the modified HTML host file **war/HelloWorld.html**.

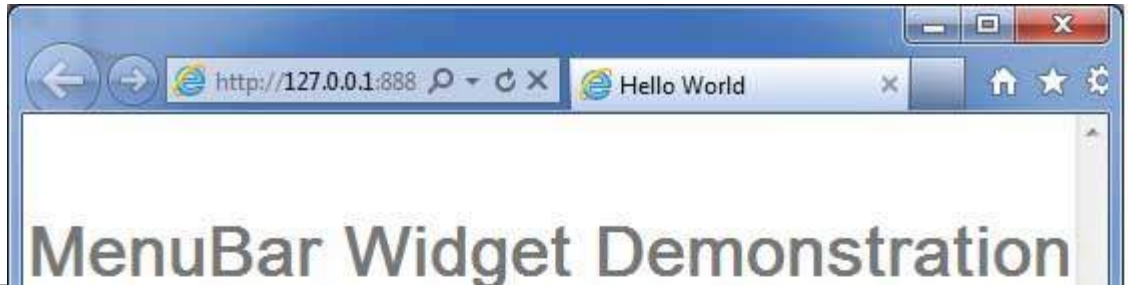
```
<html>
<head>
<title>Hello World</title>
   <link rel="stylesheet" href="HelloWorld.css"/>
   <script language="javascript" src="helloworld/helloworld.nocache.js">
   </script>
</head>
<body>

<h1>MenuBar Widget Demonstration</h1>
<div ></div>

</body>
</html>
```

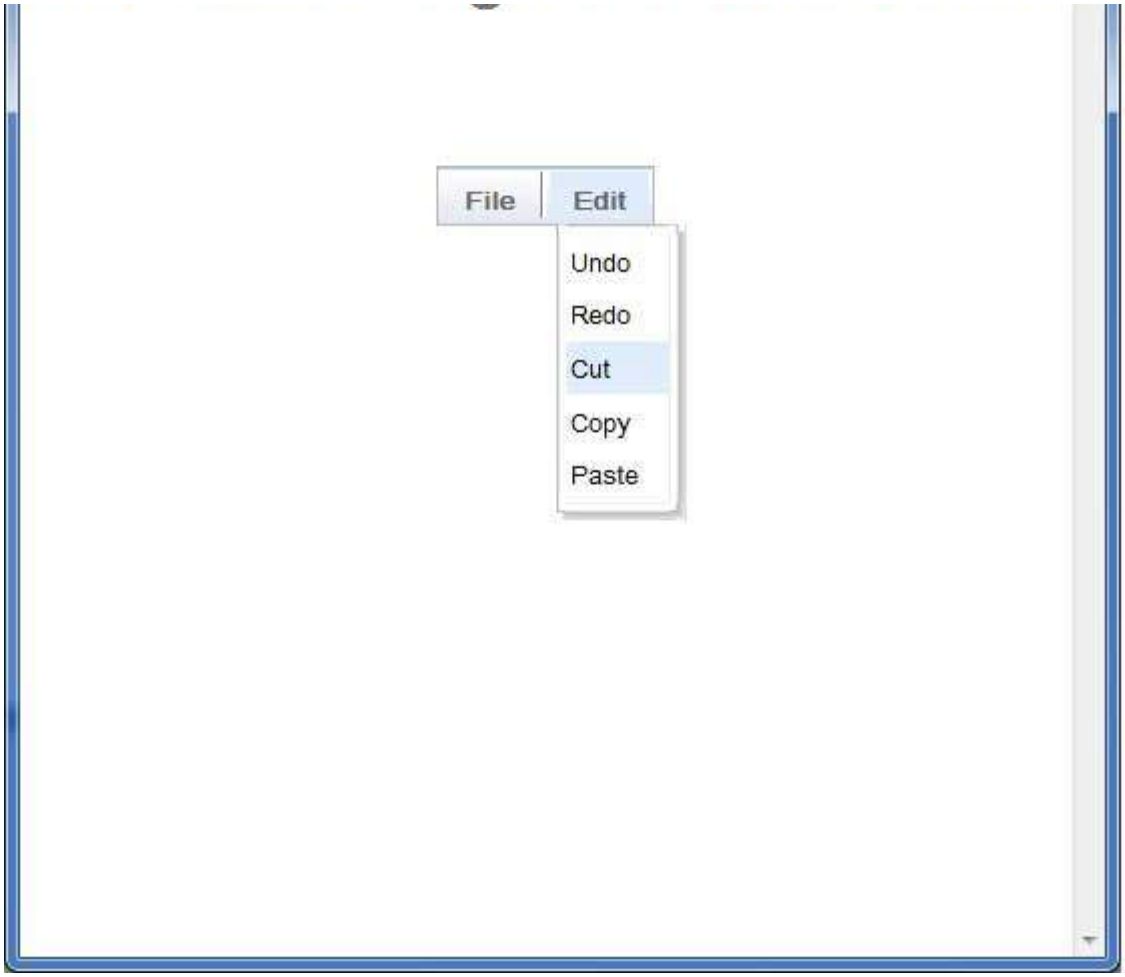
Let us have following content of Java file **src/com.tutorialspoint/HelloWorld.java** which will demonstrate use of MenuBar widget.

```
package com.tutorialspoint.client;

import com.google.gwt.core.client.EntryPoint;
import com.google.gwt.user.client.Command;
import com.google.gwt.user.client.Window;
import com.google.gwt.user.client.ui.MenuBar;
import com.google.gwt.user.client.ui.MenuItem;
import com.google.gwt.user.client.ui.RootPanel;

public class HelloWorld implements EntryPoint {

   private void showSelectedMenuItem(String menuItemName){
      Window.alert("Menu item: "+menuItemName+" selected");
   }

   public void onModuleLoad() {

   // Create a menu bar
   MenuBar menu = new MenuBar();
   menu.setAutoOpen(true);
   menu.setWidth("100px");
   menu.setAnimationEnabled(true);

   // Create the file menu
   MenuBar fileMenu = new MenuBar(true);
   fileMenu.setAnimationEnabled(true);

   fileMenu.addItem("New", new Command() {
      @Override
      public void execute() {
         showSelectedMenuItem("New");
      }
   });
   fileMenu.addItem("Open", new Command() {
      @Override
```

```
      public void execute() {
         showSelectedMenuItem("Open");
      }
   });
   fileMenu.addSeparator();
   fileMenu.addItem("Exit", new Command() {
      @Override
      public void execute() {
         showSelectedMenuItem("Exit");
      }
   });

   // Create the edit menu
   MenuBar editMenu = new MenuBar(true);
   editMenu.setAnimationEnabled(true);

   editMenu.addItem("Undo", new Command() {
      @Override
      public void execute() {
         showSelectedMenuItem("Undo");
      }
   });
   editMenu.addItem("Redo", new Command() {
      @Override
      public void execute() {
         showSelectedMenuItem("Redo");
      }
   });
   editMenu.addItem("Cut", new Command() {
      @Override
      public void execute() {
         showSelectedMenuItem("Cut");
      }
   });
   editMenu.addItem("Copy", new Command() {
      @Override
      public void execute() {
         showSelectedMenuItem("Copy");
      }
   });
   editMenu.addItem("Paste", new Command() {
   @Override
      public void execute() {
         showSelectedMenuItem("Paste");
      }
   });

   menu.addItem(new MenuItem("File", fileMenu));
   menu.addSeparator();
   menu.addItem(new MenuItem("Edit", editMenu));

   //add the menu to the root panel
   RootPanel.get("gwtContainer").add(menu);
   }
}
```

Once you are ready with all the changes done, let us compile and run the application in development mode as we did in GWT - Create Application chapter. If everything is fine with your application, this will produce following result:





Selecting any value in menubar, will popup an alert message showing the selected value.