



We will build a reusable Tabbed Menu using J2ME.



View it in action

You can find a midlet showing this component in action [here](#).

Our component will support:

- Full styling of tabs (bg color, fore color, font face, margin, paddings and corner radius)
- Automatic horizontal scrolling, so that you can put as many tab you want, without caring about screen width

Building the tabbed menu

Let's start from the customizable variables, whose aim is self-explaining:

```
int background = 0xffffffff;
int bgColor = 0xcccccc;
int bgFocusedColor = 0x0000ff;
int foreColor = 0x000000;
int foreFocusedColor = 0xffffffff;
int cornerRadius = 4;
int padding = 2;
int margin = 2;
Font font = Font.getDefaultFont();
int scrollStep = 20;
```

Then we'll define some internal variables, used to maintain the tabbed menu state:

```
int selectedTab = 0; //selected tab index
int[] tabsWidth = null; //width of single tabs
int[] tabsLeft = null; //left X coordinate of single tabs
int tabHeight = 0; //height of tabs (equal for all tabs)
String[] tabs = null; //tab labels
int menuWidth = 0; //total menu width

int viewportWidth = 0; //visible viewport width
int viewportX = 0; //current viewport X coordinate
```

Now, we'll define a simple constructor accepting tab labels and viewport width as parameters:

View it in action

Tabbed_Menu_in_Java_ME

```
public TabMenu(String[] tabs, int width)
{
    this.tabs = tabs;

    this.viewportWidth = width;

    initialize()
}
void initialize()
{
    tabHeight = getHeight() + cornerRadius + 2 * padding;

    menuWidth

    tabsWidth = int[tabs.length];
    tabsLeft = int[tabs.length];

    for(int i = 0; i < tabsWidth.length; i++)
    {
        [i] = fontWidth * tabs[i].length() + 2 * padding + 2 * cornerRadius;

        [i] = menuWidth -

        += tabsWidth[i];

    if(i > 0)
    {
        += margin;    menuWidth
    }
    }
}
```

Now, let's define an utility method to check if a tab is visible or not.

```
private boolean isTabVisible(int tabIndex)
{
    return tabsLeft[tabIndex] < viewportX + viewportWidth &&
        [tabIndex] + tabsWidth[tabIndex] >= viewportX;
}
```

And now we'll implement the tab switching/scrolling methods, that will use the isTabVisible method defined above:

```
public void goRight()
{
    (+1); go
}
public void goLeft()
{
    (-1); go
}
private void go(int delta)
{
    int newTab = Math.max(0, Math.min(tabs.length - 1, selectedTab + delta));

    boolean scroll = true;

    if(newTab != selectedTab && isTabVisible(newTab))
    {
        = newTab; selectedTab
    }
}
```

Tabbed_Menu_in_Java_ME

```
if((delta > 0 && tabsLeft[selectedTab] + tabsWidth[selectedTab] > viewportX + viewportWidth) ||
(delta < 0 && tabsLeft[selectedTab] < viewportX))
{
    scroll = true;
}
else
{
    scroll = false;
}
if(scroll)
{
    viewportX = Math.max(0, Math.min(menuWidth - viewportWidth, viewportX + delta * scrollStep));
}
}
```

And now, we're ready to paint our menu :)

```
public void paint(Graphics g)
{
    int currentX = - viewportX;

    setCliq(0, 0, viewportWidth, tabHeight);

    setColgr(background);
    fillRegt(0, 0, viewportWidth, tabHeight);

    for(int i = 0; i < tabs.length; i++)
    {
        setColor(i == selectedTab ? bgFocusedColor : bgColor);

        fillRoundRect(currentX, 0, tabsWidth[i], tabHeight + cornerRadius, 2 * cornerRadius, 2 * cornerRadius);

        setColor(i == selectedTab ? foreFocusedColor : foreColor);

        drawString(tabs[i], currentX + cornerRadius + padding, cornerRadius + padding, Graphics.LEFT |
            Graphics.TOP,
            currentX += tabsWidth[i] + margin;
    }
}
```

Download source code

You can download source code of TabMenu here:

- [TabMenu.java main class](#)
- [sample usage of TabMenu class](#)