

Civilization 5 UI Tutorial

Part 3, Drop-Down Menus

Version: 1.0

Status: Draft

Date: 11 May 2012

Author: William Howard

Contents

Overview	3
Create a Menu Mod	3
Menu 1 - Simple Drop-Down	3
Menu 1b - Sorting the Entries.....	8
Menu 2 - Entry Appearance	Error! Bookmark not defined.
Menu 2b - Button Appearance	Error! Bookmark not defined.
Menu 3 - Scroll Area Appearance	Error! Bookmark not defined.
Summary	Error! Bookmark not defined.

Overview

The objective of this tutorial is to illustrate how to create Civ 5 User Interface (UI) drop-down menus.

The tutorial takes the practical approach - first we will write some "code", then we will look at what it does. At each step we will have working UI dialogs - which may or may not in themselves be useful - but which can be used as a working starting point for your own UI mods.

This tutorial assumes you have used ModBuddy and FireTuner to create your own simple non-UI mods, and also that you have a basic understanding of XML. Some Lua will be needed, but that will be explained in each step.

All the code in this tutorial can be downloaded from the in-game ModHub as "Test - UI Tutorial - 3 Menus" found under the "Other" category.

So, to start we need a ModBuddy project ...

Create a Menu Mod

Using ModBuddy, create a new mod called "Test - Menu" (or some such).

To this mod add the two folders "UI" and "XML". In the UI folder create the two files "Menu.xml" and "Menu.lua" (delete the standard content added to these files). In the XML folder create the file "MenuText.xml" (you can leave the standard content in this file).

In the mod's properties, on the "Mod Info" tab, uncheck "Affects Saved Games". On the "Actions" tab, add an "On Mod Activated - Update Database" entry for "XML/MenuText.xml". On the "Content" tab, add an "InGameUIAddin" for "UI/Menu.xml".

Save the project.

Menu 1 - Simple Drop-Down

In "Tutorial 2 - Dynamic Buttons", we created a dialog that contained a scrollable list of met City States. The problem with scrollable lists is that they take up a lot of space when most of the time we are only interested in the one entry the user selected. This is where drop-down menus come in. For most of the time they are no bigger than a button displaying what was last selected. However, when the user clicks the button a scrollable list of options is displayed for the user to choose from. When the user has made their choice the scrollable list vanishes and the button displays the selected option - all the functionality with only a small portion of the screen occupied!

This tutorial will build on the techniques developed in Tutorial 2, replacing the scrollable list with a drop-down menu.

Add the following to the "UI/Menu.xml" file

```
<?xml version="1.0" encoding="utf-8" ?>
<Context>
    <Box Style="BGBlock_ClearTopBar" />
```

UI Tutorial - Menus

```
<Instance Name="CityState">
    <Label ID="Label" Anchor="L,T" Font="TwCenMT20" FontStyle="Shadow"
ColorSet="Beige_Black_Alpha" />
</Instance>

<Grid Size="600,400" Anchor="C,C" Style="Grid9DetailFive140"
ConsumeMouse="1">
    <Label ID="Message" Anchor="C,T" Offset="0,50" Font="TwCenMT24"
FontStyle="Shadow" ColorSet="Beige_Black_Alpha"
String="TXT_KEY_TEST_MENU_CS_TITLE"/>

    <PullDown ID="CSMenu" Style="GenericPullDown"
ScrollThreshold="170" Size="190,27" SpaceForScroll="1" Anchor="C,T"
Offset="0,90"/>

    <GridButton ID="OK" Size="140,36" Anchor="C,B" Offset="0,50"
Style="BaseButton" ToolTip="TXT_KEY_TEST_MENU_BUTTON_OK_TT">
        <Label Anchor="C,C" Offset="0,-2"
String="TXT_KEY_TEST_MENU_BUTTON_OK" Font="TwCenMT24"
FontStyle="Shadow" ColorSet="Beige_Black_Alpha" />
    </GridButton>
</Grid>
</Context>
```

Add the following to the "UI/Menu.lua" file

```
function OnShowHide(bHide, bInit)
    if (not bHide) then
        UpdateCsList()
    end
end

ContextPtr:SetShowHideHandler(OnShowHide)

function OnOK()
    ContextPtr:SetHide(true)
end

Controls.OK:RegisterCallback(Mouse.eLClick, OnOK)

function OnCSSelected(iCS)
    local pCS = Players[iCS]
    Controls.CSMenu:GetButton():SetText(pCS:GetName())

    UI.LookAt(pCS:GetCapitalCity():Plot())
end

function UpdateCsList()
    local iTeam = Game.GetActiveTeam()
```

UI Tutorial - Menus

```
Controls.CSMenu:ClearEntries()

for iCS = GameDefines.MAX_MAJOR_CIVS, GameDefines.MAX_CIV_PLAYERS-1,
1 do
    local pCS = Players[iCS]
    if pCS:IsEverAlive() then
        if (pCS:IsAlive() and Teams[pCS:GetTeam()]:IsHasMet(iTeam)) then
            local sCsName = pCS:GetName()

            local entry = {}
            Controls.CSMenu:BuildEntry("InstanceOne", entry)

            entry.Button:SetVoid1(iCS)
            entry.Button:SetText(sCsName)
        end
    end
end

Controls.CSMenu:GetButton():LocalizeAndSetText(
"TXT_KEY_TEST_MENU_CS_CHOOSE")

Controls.CSMenu:CalculateInternals()
Controls.CSMenu:RegisterSelectionCallback(OnCSSelected)
end
```

Add the following to the "XML/MenuText.xml" file

```
<?xml version="1.0" encoding="utf-8"?>
<GameData>
    <Language_en_US>
        <Row Tag="TXT_KEY_TEST_MENU_BUTTON_OK">
            <Text>OK</Text>
        </Row>
        <Row Tag="TXT_KEY_TEST_MENU_BUTTON_OK_TT">
            <Text>Left click to dismiss the popup</Text>
        </Row>

        <Row Tag="TXT_KEY_TEST_MENU_CS_TITLE">
            <Text>City States</Text>
        </Row>
        <Row Tag="TXT_KEY_TEST_MENU_CS_CHOOSE">
            <Text>Select to view ...</Text>
        </Row>
    </Language_en_US>
</GameData>
```

Save the files and build the mod. Start Civ 5, enable the mod, and start a new game. In the middle of the screen you should see the "City States" dialog box with an empty drop-down. Click the OK

button to dismiss the dialog. Found your capital, build a Scout or two and wander about until you've met half-a-dozen or more City States. Using FireTuner, display the dialog again



In the context we replaced the scrollable list with

```
<PullDown ID="CSMenu" Style="GenericPullDown" ScrollThreshold="170"
Size="190,27" SpaceForScroll="1" Anchor="C,T" Offset="0,90"/>
```

This defines a generic drop-down menu. The main button is 190x27 pixels in size, the size of the scrollable area is 170 pixels high and the same width as the main button and space should be allocated within it for a vertical scroll bar. That's it. So long as we want the standard "look and feel" we have nothing else to do in the XML

The code to populate the list should mostly be familiar

```
function UpdateCsList()
    local iTeam = Game.GetActiveTeam()

    Controls.CSMenu:ClearEntries()

    for iCS = GameDefines.MAX_MAJOR_CIVS, GameDefines.MAX_CIV_PLAYERS-1,
    1 do
        local pCS = Players[iCS]
        if pCS:IsEverAlive() then
            if (pCS:IsAlive() and Teams[pCS:GetTeam()]:IsHasMet(iTeam)) then
                local sCsName = pCS:GetName()

                local entry = {}
```

```

        Controls.CSMenu:BuildEntry("InstanceOne", entry)

        entry.Button:SetVoid1(iCS)
        entry.Button:SetText(sCsName)
    end
end
end

Controls.CSMenu:GetButton():LocalizeAndSetText(
"TXT_KEY_TEST_MENU_CS_CHOOSE")

Controls.CSMenu:CalculateInternals()
Controls.CSMenu:RegisterSelectionCallback(OnCSSelected)
end

```

Before adding new entries to the drop-down list we remove all existing entries with

```
Controls.CSMenu:ClearEntries()
```

To add an entry we use

```
Controls.CSMenu:BuildEntry("InstanceOne", entry)
```

to create the entry ("InstanceOne" is the name of the default instance that comes "free" with the generic drop-down style), and

```

entry.Button:SetVoid1(iCS)
entry.Button:SetText(sCsName)

```

to set the text of the entry and the value our callback will receive when the user selects that entry ("Button" is the name of the button in the default "InstanceOne")

```

Controls.CSMenu:GetButton():LocalizeAndSetText(
"TXT_KEY_TEST_MENU_CS_CHOOSE")

```

Sets the initial text that will be displayed in the main button (rather than referring to the main button by name we call the GetButton() function to retrieve it from the drop-down menu)

```
Controls.CSMenu:CalculateInternals()
```

Tells the drop-down menu we have finished adding items so it needs to perform the "house keeping" to re-layout the content of the drop-down scroll panel, and

```
Controls.CSMenu:RegisterSelectionCallback(OnCSSelected)
```

registers our callback function for when a user selects an item in the drop-down.

Our callback function receives the value we associated with the entry in the drop-down (in our case the player id of the City State)

```

function OnCSSelected(iCS)
    local pCS = Players[iCS]

```

```
Controls.CSMenu:GetButton():SetText(pCS:GetName())

UI.LookAt(pCS:GetCapitalCity():Plot())
end
```

sets the name of the City State into the main button, and then scrolls the map to view the City State.

In our scrollable list of City States we sorted them alphabetically by name, so we'll add that functionality to our drop-down menu.

Menu 1b - Sorting the Entries

Change the <PullDown> element in the "UI/Menu.xml" file to

```
<PullDown ID="CSMenu" Style="GenericPullDown" ScrollThreshold="170"
Size="190,27" SpaceForScroll="1" Anchor="C,T" Offset="0,90">
  <StackData ID="CSMenuStack"/>
</PullDown>
```

Copy the following to the "UI/Button.lua" file (the bits in bold have been added)

```
local g_SortTable

function OnShowHide(bHide, bInit)
  if (not bHide) then
    UpdateCsList()
  end
end
ContextPtr:SetShowHideHandler(OnShowHide)

function OnOK()
  ContextPtr:SetHide(true)
end
Controls.OK:RegisterCallback(Mouse.eLClick, OnOK)

function SortByName(a, b)
  local sNameA = g_SortTable[tostring(a)].Name
  local sNameB = g_SortTable[tostring(b)].Name

  return sNameA < sNameB
end

function OnCSSelected(iCS)
  local pCS = Players[iCS]
  Controls.CSMenu:GetButton():SetText(pCS:GetName())

  UI.LookAt(pCS:GetCapitalCity():Plot())
end
```


UI Tutorial - Menus

```
function UpdateCsList()
    local iTeam = Game.GetActiveTeam()

    Controls.CSMenu:ClearEntries()
    g_SortTable = {}

    for iCS = GameDefines.MAX_MAJOR_CIVS, GameDefines.MAX_CIV_PLAYERS-1,
1 do
        local pCS = Players[iCS]
        if pCS:IsEverAlive() then
            if (pCS:IsAlive() and Teams[pCS:GetTeam()]:IsHasMet(iTeam)) then
                local sCsName = pCS:GetName()

                local entry = {}
                Controls.CSMenu:BuildEntry("InstanceOne", entry)
                g_SortTable[tostring(entry.Button)] = {Name=sCsName}

                entry.Button:SetVoid1(iCS)
                entry.Button:SetText(sCsName)
            end
        end
    end
end

Controls.CSMenuStack:SortChildren(SortByName)

Controls.CSMenu:GetButton():LocalizeAndSetText(
"TXT_KEY_TEST_MENU_CS_CHOOSE")

Controls.CSMenu:CalculateInternals()
Controls.CSMenu:RegisterSelectionCallback(OnCSSelected)
end
```

save the changes, rebuild the mod, restart Civ 5, re-enable the mod and start a new game. Dismiss the dialog, wander around a bit to discover several City States and then display the dialog again (via FireTuner)



The City States now appear in alphabetical order.

All the Lua code we added should be familiar from Tutorial 2

The `<StackData>` element added to the `<PullDown>` element

```
<StackData ID="CSMenuStack"/>
```

serves to give the default stack a name, so we can access it to sort it.

```
Controls.CSMenuStack:SortChildren(SortByName)
```

If all you want are standard look-and-feel drop-down menus you can stop here.

Continued in Part 2