Civilization 5 UI Tutorial

Part 1, The Basics

Version: 1.0

Status: Draft

Date: 11 May 2012

Author: William Howard

Contents

Overview 3
Create a Dialog Mod 3
Dialog 1 - Hello World! 3
Dialog 1b - Text Attributes and Anchors 5
Dialog 2 - Borders (Grids) 6
Dialog 2b - Anchors Revisited 10
Dialog 2c - Containing the Mouse 11
Dialog 3 - Adding an OK Button ..... **Error! Bookmark not defined.**
Dialog 4 - Static and Dynamic Decoration ..... **Error! Bookmark not defined.**
Dialog 4b - Changing the Icon ..... **Error! Bookmark not defined.**
Dialog 5 - More Buttons ..... **Error! Bookmark not defined.**
Dialog 5b - Easier More Buttons ..... **Error! Bookmark not defined.**
Dialog 6 - Odds and Ends ..... **Error! Bookmark not defined.**
Summary ..... **Error! Bookmark not defined.**

Overview

The objective of this tutorial is to introduce the basic Civ 5 User Interface (UI) elements and controls that are needed to construct dialogs for presenting information to the user and receiving their decisions (in the form of button clicks).

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 - 1 Basics" found under the "Other" category.

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

Create a Dialog Mod

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

To this mod add the two folders "UI" and "XML". In the UI folder create the two files "Dialog.xml" and "Dialog.lua" (delete the standard content added to these files). In the XML folder create the file "DialogText.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/DialogText.xml". On the "Content" tab, add an "InGameUIAddin" for "UI/Dialog.xml".

Save the project.

Dialog 1 - Hello World!

Computer tutorials from the year dot have always started with a simple "Hello World!" example, so let's not break with tradition.

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

```
<?xml version="1.0" encoding="utf-8" ?>
<Context>
  <Label Anchor="C,C" String="Hello World!"/>
</Context>
```

Save the file and build the mod. Start Civ 5, enable the mod, and start a new game. In the middle of the screen (probably right over your initial settler, so you may need to scroll the map a bit) you should see the text "Hello World!"

So what does the XML code do? The `<Label String="Hello World!"/>` bit should be obvious - place a label showing the text "Hello World!".

The `Anchor="C,C"` part places the text in the middle of the screen - or more accurately it centres it left-to-right and it centres it top-to-bottom (anybody use to other text layout languages will probably know this as alignment)

The `<Context></Context>` bit takes a little more explaining. All UI components - lines, boxes, buttons, menus, etc - must be contained within an XML <Context> element - referred to simply as "the context". You can think of the context as a special piece of transparent film the size of the screen. The UI components within the Context XML element are "drawn" onto the film. It is special in that if we pass a signal into it, the film is totally transparent (including the components we have drawn on it). If we remove the signal, while the film itself remains transparent, we can now see our components.

Start FireTuner, click on the "Lua Console" tab if necessary and from the drop-down select the "Dialog" context. In the command line area enter

```
ContextPtr:SetHide(true)
```

and press Return - the "Hello World!" text will vanish. This is the "signal" to the film to make what is drawn on it transparent. Enter the command

```
ContextPtr:SetHide(false)
```

and press Return and the text will re-appear.

As you will probably be typing those a lot, we will set up some short-cuts for them. In the "UI/Dialog.lua" file enter the following

```
function Show()
```

```
  ContextPtr:SetHide(false)
end

function Hide()
  ContextPtr:SetHide(true)
end
```

Exit Civ 5, rebuild the mod, start Civ 5, re-enable the mod, and start a new game - "Hello World!" should be back in the middle of the screen. From FireTuner's Lua Console reselect the "Dialog" context and enter `Hide()` - the text will vanish - and then `Show()` to bring it back.

Congratulations! Not only have you just created a Civ 5 UI mod, you've also written some Lua - that wasn't really too hard was it!

Dialog 1b - Text Attributes and Anchors

If we wrote all our text at that small size our users would soon stop using our mod due to eye strain.

So let's experiment with font attributes and anchoring. Change the "UI/Dialog.xml" to

```
<?xml version="1.0" encoding="utf-8" ?>
<Context>
  <Label Anchor="C,C" Font="TwCenMT24" FontStyle="Shadow"
ColorSet="Beige_Black_Alpha" String="Hello World!"/>

  <Label Anchor="L,T" Font="TwCenMT24" FontStyle="Shadow"
ColorSet="Beige_Black_Alpha" String="Top Left"/>
  <Label Anchor="C,T" Font="TwCenMT24" FontStyle="Shadow"
ColorSet="Beige_Black_Alpha" String="Top Centre"/>
  <Label Anchor="R,T" Font="TwCenMT24" FontStyle="Shadow"
ColorSet="Beige_Black_Alpha" String="Top Right"/>

  <Label Anchor="L,C" Font="TwCenMT24" FontStyle="Shadow"
ColorSet="Beige_Black_Alpha" String="Centre Left"/>
  <Label Anchor="R,C" Font="TwCenMT24" FontStyle="Shadow"
ColorSet="Beige_Black_Alpha" String="Centre Right"/>

  <Label Anchor="L,B" Font="TwCenMT24" FontStyle="Shadow"
ColorSet="Beige_Black_Alpha" String="Bottom Left"/>
  <Label Anchor="C,B" Font="TwCenMT24" FontStyle="Shadow"
ColorSet="Beige_Black_Alpha" String="Bottom Centre"/>
  <Label Anchor="R,B" Font="TwCenMT24" FontStyle="Shadow"
ColorSet="Beige_Black_Alpha" String="Bottom Right"/>
</Context>
```

save the changes, rebuild the mod, restart Civ 5, re-enable the mod and start a new game.

The "Hello World!" text should be a lot more legible and there are eight more phrases scattered around the edges of the screen.

The eight phrases indicate how the Anchor attribute changes the position of the text. Note that while in English we tend to say "bottom left" and "top right" the computer wants them the other way around "L,B" and "R,T" - like most graphical systems the x co-ordinate (left/right) comes before the y co-ordinate (top/bottom).

The ColorSet attribute determines the text and background colours to use. While Civ 5 defines a few colour sets the one used by the game UI is "Beige_Black_Alpha" and to be consistent with what our users are expecting, we should also use it.

The FontStyle="Shadow" attribute determines if the text has a "drop-shadow" (only visible on a coloured background). The only other option used by the game is FontStyle="Base" in the mid-screen popups.

The Font attribute controls the size of the font, the only values used by the game are TwCenMT14, TwCenMT16, TwCenMT18, TwCenMT20, TwCenMT22 and TwCenMT24 - the number on the end refers to the point size, so 14 is the smallest and 24 the largest. The letters at the front refer to the font family, but only one seems to be available.

OK, so our users can read our text, but it's still pretty hard to see!

Dialog 2 - Borders (Grids)

What we want to do is make our text jump out at the user, and the easiest (and standard) way to do this is to place it with a frame (commonly known as a dialog box)

Change the "UI/Dialog.xml" file to

```
<?xml version="1.0" encoding="utf-8" ?>
<Context>
  <Grid Size="400,270" Anchor="C,C" Style="Grid9DetailFive140">
```

```
    <Label Anchor="C,C" Font="TwCenMT24" FontStyle="Shadow"
ColorSet="Beige_Black_Alpha" String="TXT_KEY_TEST_DIALOG_MESSAGE"/>
  </Grid>
</Context>
```

and add the following to the "XML/DialogText.xml" file

```
<?xml version="1.0" encoding="utf-8"?>
<GameData>
  <Language_en_US>
    <Row Tag="TXT_KEY_TEST_DIALOG_MESSAGE">
      <Text>Hello World!</Text>
    </Row>
  </Language_en_US>
</GameData>
```

save the changes, rebuild the mod, restart Civ 5, re-enable the mod and start a new game.

The "Hello World!" text should be a lot more visible

So what's this TXT_KEY_TEST_DIALOG_MESSAGE thing?

We probably shouldn't assume that all our users can read English. Placing a fixed message of "Hello World!" inside the context is not a good idea, as it makes it impossible to replace it with alternative languages - German, French, etc. So we place a "text key" in the UI elements within the context and then provide translations of those "text keys" within the game database (via, in our case, the "XML/DialogText.xml" file)

To translate our UI we now just need to provide translations of the text keys (or someone else could provide them to us), eg

```
<?xml version="1.0" encoding="utf-8"?>
```

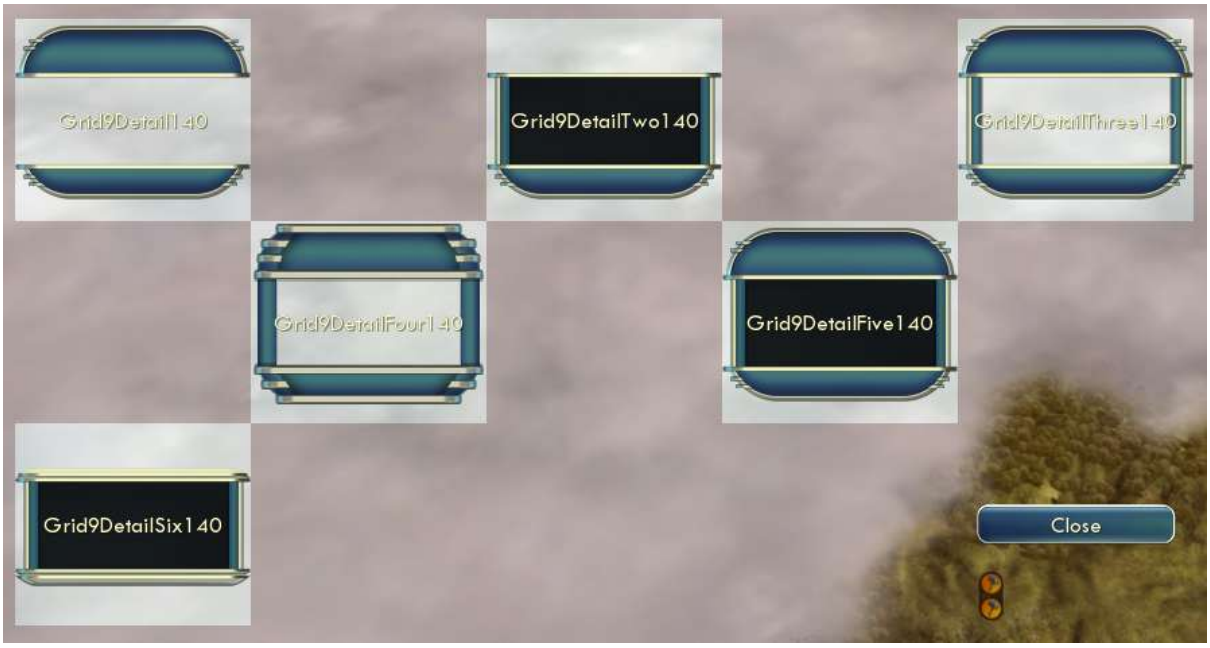
```
<GameData>
  <Language_en_US>
    <Row Tag="TXT_KEY_TEST_DIALOG_MESSAGE">
      <Text>Hello World!</Text>
    </Row>
  </Language_en_US>

  <Language_fr_FR>
    <Row Tag="TXT_KEY_TEST_DIALOG_MESSAGE">
      <Text>Bonjour Tout le Monde!</Text>
    </Row>
  </Language_fr_FR>

  <Language_de_DE>
    <Row Tag="TXT_KEY_TEST_DIALOG_MESSAGE">
      <Text>Hallo Welt!</Text>
    </Row>
  </Language_de_DE>
</GameData>
```

Text keys must always start with "TXT_KEY_" and must be in capitals (so "TXT_KEY_TEST_DIALOG_MESSAGE", not "TEST_DIALOG_MESSAGE" nor "TXT_KEY_Test_Dialog_Message") failing to start them with "TXT_KEY_" will mean that the UI treats them as plain text (so you'll see "TEST_DIALOG_MESSAGE" and not "Hello World!") and failing to use all capitals will generate one warning per lower case letter in the xml.log file (so "TXT_KEY_Test_Dialog_Message" will generate 14 warnings!)

The `<Grid Size="400,270" Anchor="C,C" Style="Grid9DetailFive140">` element creates the black box behind and edging around the text. The grid is centred in the screen (Anchor="C,C" remember) and is 400 pixels wide by 270 pixels tall (x then y in "computer speak"). The Style attribute determines which graphical elements are used to make up the borders - various different styles are available, the most common are



(from http://forums.civfanatics.com/showthread.php?t=408628)

So why is the <Label> element within the <Grid> element and does it make a difference?

```
  <Grid Size="400,270" Anchor="C,C" Style="Grid9DetailFive140">
    <Label Anchor="C,C" Font="TwCenMT24" FontStyle="Shadow"
ColorSet="Beige_Black_Alpha" String="TXT_KEY_TEST_DIALOG_MESSAGE"/>
  </Grid>
```

Yes, it does make a difference. UI components are drawn in the order they are encounter within the <Context> element. If two (or more) components overlap what you will see will depend on what was drawn last. For example,

```
  <Grid Size="400,270" Anchor="C,C" Style="Grid9DetailFive140"/>
  <Label Anchor="C,C" String="TXT_KEY_TEST_DIALOG_MESSAGE"/>
```

draws the grid first and then the text, so the text appears in the middle of the grid, whereas

```
  <Label Anchor="C,C" String="TXT_KEY_TEST_DIALOG_MESSAGE"/>
  <Grid Size="400,270" Anchor="C,C" Style="Grid9DetailFive140"/>
```

draws the text first and then draws the grid over the top of it thereby obscuring it.

Both are different to

```
<Grid Size="400,270" Anchor="C,C" Style="Grid9DetailFive140">
    <Label Anchor="C,C" Font="TwCenMT24" FontStyle="Shadow"
ColorSet="Beige_Black_Alpha" String="TXT_KEY_TEST_DIALOG_MESSAGE"/>
  </Grid>
```

as the next section demonstrates.

Dialog 2b - Anchors Revisited

So why is the <Label> element within the <Grid> element?

Change the "UI/Dialog.xml" file to be

```
<?xml version="1.0" encoding="utf-8" ?>
<Context>
  <Grid Size="400,270" Anchor="C,C" Style="Grid9DetailFive140">
    <Label Anchor="C,C" Font="TwCenMT24" FontStyle="Shadow"
ColorSet="Beige_Black_Alpha" String="TXT_KEY_TEST_DIALOG_MESSAGE"/>

    <Label Anchor="L,T" Font="TwCenMT20" FontStyle="Shadow"
ColorSet="Beige_Black_Alpha" String="L,T"/>
    <Label Anchor="C,T" Font="TwCenMT20" FontStyle="Shadow"
ColorSet="Beige_Black_Alpha" String="C,T"/>
    <Label Anchor="R,T" Font="TwCenMT20" FontStyle="Shadow"
ColorSet="Beige_Black_Alpha" String="R,T"/>

    <Label Anchor="L,C" Font="TwCenMT20" FontStyle="Shadow"
ColorSet="Beige_Black_Alpha" String="L,C"/>
    <Label Anchor="R,C" Font="TwCenMT20" FontStyle="Shadow"
ColorSet="Beige_Black_Alpha" String="R,C"/>

    <Label Anchor="L,B" Font="TwCenMT20" FontStyle="Shadow"
ColorSet="Beige_Black_Alpha" String="L,B"/>
    <Label Anchor="C,B" Font="TwCenMT20" FontStyle="Shadow"
ColorSet="Beige_Black_Alpha" String="C,B"/>
    <Label Anchor="R,B" Font="TwCenMT20" FontStyle="Shadow"
ColorSet="Beige_Black_Alpha" String="R,B"/>
  </Grid>
</Context>
```

(yes we're being "bad" using fixed text and not text keys, but they won't be there long!)

Save the changes, rebuild the mod, restart Civ 5, re-enable the mod and start a new game. Hello World is now joined by eight other abbreviations.

The abbreviations are clustered around the edge of the grid because their corresponding <Label> element was placed within the <Grid> element. Components within another component are anchored (aligned) relative to their containing component. If we changed the anchor of the grid to be "L,T", the grid would appear in the top left corner of the screen and the nine pieces of text would move with it. If the <Label> elements had been placed after the <Grid> element (and not within it), while the grid would have moved to the top-left corner, "Hello World!" would remain in the centre of the screen.

The same is true if we had changed the size of the grid - the eight abbreviations around the edges would have moved outwards or inwards with the edges of the grid.

In general, when creating a dialog box we will have a context containing a single grid which will then contain all the UI controls. That way, if we have to move the grid on the screen or change its size, all the controls will adjust their positions accordingly.

You may have noticed a problem while displaying these grids - and that's that the mouse pointer tells you what the tiles underneath the grid are! So we'd better fix that.

Dialog 2c - Containing the Mouse

Change the "UI/Dialog.xml" file to be

```
<?xml version="1.0" encoding="utf-8" ?>
<Context>
  <Grid Size="400,270" Anchor="C,C" Style="Grid9DetailFive140"
ConsumeMouse="1">
    <Label Anchor="C,C" Font="TwCenMT24" FontStyle="Shadow"
ColorSet="Beige_Black_Alpha" String="TXT_KEY_TEST_DIALOG_MESSAGE"/>
  </Grid>
```

```
    </Context>
```

(see, told you that fixed text wouldn't be there long!)

Save the changes, rebuild the mod, restart Civ 5, re-enable the mod and start a new game. If you pause the mouse over the grid it will no longer tell you about the tile underneath.

as opposed to

The ConsumeMouse="1" attribute tells the Civ 5 UI engine not to tell any component under this one about the mouse - not only does this stop the "tooltips" from appearing it also means you can't click on any buttons or units under the grid either.

In general, the main grid of the dialog box should always have the ConsumeMouse="1" attribute.

Great, so we have the bare bones of a dialog box to impart information to our mod's users, but how do they dismiss it?

Continued in Part 2