Civilization 5 UI Tutorial

Part 7, Animations

Version: 1.0

Status: Draft

Date: 23 May 2012

Author: William Howard

Contents

Overview

The objective of this tutorial is to cover the Civ 5 User Interface (UI) input elements and controls that relate to animations.

As this tutorial requires textures (DDS files) you will need to download the associated mod from the in-game ModHub "Test - UI Tutorial - 7 Animations" found under the "Other" category.

There are five types of animation available to the Civ V UI modder

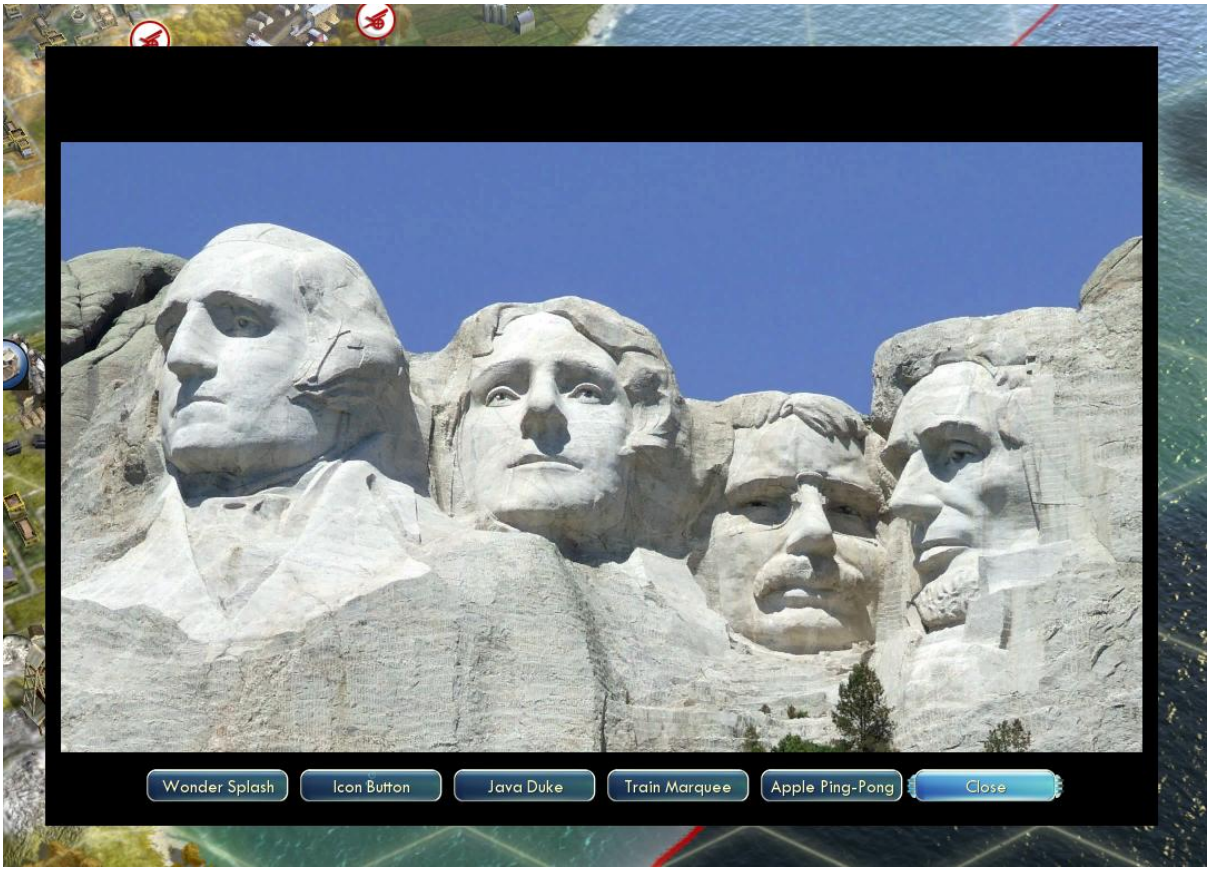
- Slide - where a component moves across or up/down the screen, for example, the civilization's details panel as the game starts or the credits
- Zoom - where the view on an image zooms in or out, for example, the wonder splash screen
- Scroll - where an image is "played" over a component, for example, the bubbles on the tech buttons
- Alpha - where a component is faded up/down or appears to glow, for example, when the mouse moves over a button and the side "handles" glow
- Flip - where an image rotates or moves, for example, the victory cup notification or the promotion chevrons in the unit panel

(Aside: It wouldn't surprise me if there are more than these. Writing these notes was a bit like the "What have the Romans ever done for us?" sketch ... "The three animation types are 'Alpha', 'Scroll' and 'Flip'." "Don't forget 'Slide'." "The FOUR ...")

Displaying the Sample Dialog

To display the popup panel to view the various animations

1) Start CivV
2) Enable this mod
3) Start a new modded game
4) Start FireTuner (from the CivV SDK)
5) Click on the Lua Console tab
6) In the drop-down, select the Animations context
7) In the command line box, enter "Show()" (without the double quotes) and press Enter



Slide Animations

These are where a component (or more likely a group of components) move across or up/down the screen. The <SlideAnim> tag groups the UI components and controls the direction, speed, etc of the "slide". Examples can be found in the LoadScreen (right to left) and NotificationPanel (top to bottom) UI contexts.

The following draws a white border in the middle of the screen and then sets up two slide animations, the SlideYAnim (inner) slides a graphic (the apple) up and down the screen within the frame, the SlideXAnim (outer) slides the inner slide (and hence the apple) from side to side within the frame ... apple ping-pong!

```
<Box Anchor="C,C" Size="420,420" Color="0,0,0,255" ID="SlideFrame">
  <Box Anchor="C,C" Size="420,420" Color="White,255"/>
  <Box Anchor="C,C" Size="416,416" Color="Black,255"/>

  <SlideAnim Anchor="C,C" Size="400,400" Start="200,0" End="-200,0"
Cycle="Bounce" Speed="1" Stopped="1" ID="SlideXAnim">
    <SlideAnim Anchor="C,C" Size="400,400" Start="0,200" End="0,-200"
Cycle="Bounce" Speed="2" Stopped="1" ID="SlideYAnim">
      <Image Anchor="C,C" Size="32,32" Texture="32x32_Apple.dds"/>
    </SlideAnim>
  </SlideAnim>
</Box>
```

Use Controls.SlideXAnim:Play() to start the side to side motion, and Controls.SlideYAnim:Play() to start the up and down motion (use the Stop() method to cease the motion of either control)

A Speed of 0 is stopped, and a speed of 20 is manic! The slowest speed used in game is 0.003 for the scrolling credits.

If you only want something to slide into position and then stop, set Cycle="Once". If you want the animation to play when shown (without the need for the Play() method), either omit the Stopped attribute or set it to 0. To reset the animation to its starting position (usually just before un-hiding it) use the SetToBeginning() method.

Zoom Animations

These are where an image is zoomed within a frame; the only example is WonderPopup. This animation doesn't have a specific tag but relies on properties of the <ScrollPanel> and dynamic scaling of the image via Lua.

The following defines a scroll panel (that doesn't actually scroll!) to clip the image ...

```
<ScrollPanel ID="WonderFrame" Size="972,568" Anchor="C,C" FullClip="1"
Disabled="1" Hidden="1">
  <Image ID="WonderSplash" Anchor="C,T" Size="972,568"/>
</ScrollPanel>
```

... as the Lua performs the dynamic scaling to produce the zoom effect.

```
local m_fAnimSeconds = 5     -- Total time to zoom
local m_fScaleFactor = 0.5   -- Starting zoom, add 1 and multiply by
100 to get a percent, so this is 150% of original size

local m_fPct = 0
local m_fOriginalSizeX = 0
local m_fOriginalSizeY = 0
local m_image = nil

function OnZoom(fDTime)
  -- Calculate the percentage to zoom this time around
  m_fPct = m_fPct + (fDTime / m_fAnimSeconds)

  if (m_fPct > 1) then
    -- If at 100%, stop the timing call-back
    ContextPtr:ClearUpdate()
    -- and make sure the image is at normal size
    m_image:SetSizeVal(m_fOriginalSizeX, m_fOriginalSizeY)
  else
    -- Otherwise zoom out a little bit
    local fScale = 1 + ((1 - m_fPct) * (1 - m_fPct) * m_fScaleFactor)
    m_image:SetSizeVal(m_fOriginalSizeX * fScale, m_fOriginalSizeY *
fScale )
```

```
  end
end

function InitZoom(control, sTexture)
  m_image = control
  m_fPct = 0

  -- Load the image into the control
  m_image:SetTextureAndResize(sTexture)
  -- Get the original size of the image
  m_fOriginalSizeX, m_fOriginalSizeY =
Controls.WonderSplash:GetSizeVal()
  -- Zoom in by magnifying the image (the ScrollPanel will crop the
image)
  m_image:SetSizeVal(m_fOriginalSizeX * (1 + m_fScaleFactor),
m_fOriginalSizeY * (1 + m_fScaleFactor))

  -- Start the timing call-back
  ContextPtr:SetUpdate(OnZoom)
  OnZoom(0)
end

-- Start the zoom
InitZoom(Controls.WonderSplash, "MtRushmore.dds")
```

By anchoring the image at different points, eg "R,T", "L,B" etc, the "camera" appears to zoom out from different directions.

Scroll Animations

These are where an image is "played" over another component, usually a button, to produce apparent motion on the control. The <ScrollAnim> tag controls the image, speed, etc of the "scroll". The animation is always bottom-to-top (or vice-versa), side-to-side scrolling doesn't seem to be possible. The most common scroll animation is the bubbles on tech buttons and the "shine" on advisor buttons. Examples can be found in the TechTree (bubbles) and AdvisorInfoPopup (shine) UI contexts.

The following plays the bubbles over a button

```
<GridButton Style="SmallButton" Size="130,32"
StateOffsetIncrement="0,0" ID="IconButton">
  <Label Anchor="C,C" String="Icon Button" Font="TwCenMT16"
ColorSet="Beige_Black_Alpha" FontStyle="Shadow" />
  <ScrollAnim Size="130,32" Speed="0.25" Texture="Bubbles256.dds"
MaskTexture="BubblesMask336x48.dds" />
</GridButton>
```

A Speed of 0 is stopped, and a speed of 20 is very, very fast! In-game speeds are all less than one. A positive speed produces motion from bottom to top, a negative speed from top to bottom.

Alpha Animations

These are where the alpha channel of an image is varied, either to make the image appear to grow brighter/dimmer or appear to fade in/out (more or less transparent). The <AlphaAnim> tag groups the UI components that are to be varied and controls the speed, etc of the "fade". There are examples of alpha animations in most UI contexts.

The following alpha animation makes the handles at the side of a button appear to glow

```
<AlphaAnim Anchor="C,C" Size="130,32" Cycle="Bounce" Speed="1"
AlphaStart=".99" AlphaEnd=".25">
  <Image Anchor="R,C" Offset="-2,0" TextureOffset="16,0"
AnchorSide="O,O" Texture="buttonsidesglow.dds" Size="16,32" />
  <Image Anchor="L,C" Offset="-2,0" TextureOffset="0,0"
AnchorSide="O,O" Texture="buttonsidesglow.dds" Size="16,32" />
</AlphaAnim>
```

If there is only one image within the <AlphaAnim> element, the image texture can be specified as an attribute of the <AlphaAnim> tag, for example,

```
<AlphaAnim ID="Icon" Anchor="C,C" Size="32,32"
Texture="BlueExclaim.dds" Cycle="Bounce" Speed="1" AlphaStart=".99"
AlphaEnd=".4"/>
```

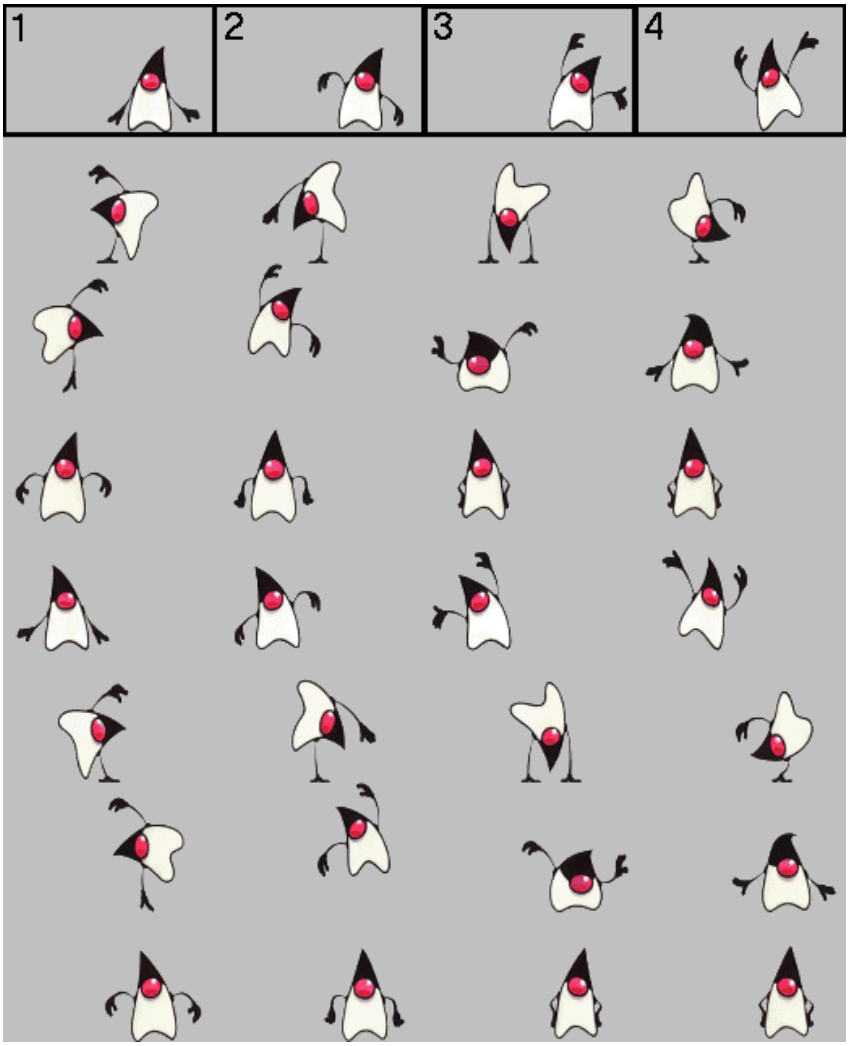
Use Controls.Icon:Stop() to stop the alpha animation and Play() to start it. To reset the animation to its starting position (usually just before un-hiding it) use the SetToBeginning() method.

A Speed of 0 is stopped, and a speed of 20 is manic - alpha animations are typically speed one. If you only want something to fade in or out and then stop, set Cycle="Once". Typically the Stopped="1" attribute is specified with a Cycle="Once" animation and then the animation started on some event from Lua with the Play() method.

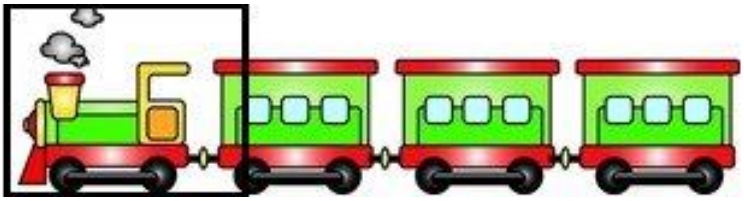
AlphaStart and AlphaEnd typically take a decimal number (to 2 places of accuracy) between 0 (off/transparent) and 1 (on/opaque), although some in-game buttons in lists (eg the select saved game menu) take values up to 2

Flip Animations

These are where parts of a larger image are player over each other to produce motion - a bit like drawing a stick man in sequential poses in the corner of a pad and then flicking the pages to make it dance. The sub-images can either be single frames of the overall motion,



or the sub-image can be taken by offsetting within a larger image



The following animates Duke

```
<FlipAnim ID="DukeAnim" Size="130,80" Anchor="C,C" Columns="4"
Speed="10" StepSize="130,80" FrameCount="32" EndPause="3"
Texture="DukeAnim.dds" />
```

The StepSize attribute controls the offset of the sub-image taken from the texture, Columns determines how many steps there are in each row, and FrameCount controls the total number of steps.

If FrameCount is not a multiple of Columns, the last frames in the last row are not displayed. This enables a related static image (for a button to start the animation say) to be placed in the same texture file and not displayed as part of the sequence.

In general, for frame based animation,

```
StepSize.X = Texture.X / Columns
StepSize.Y = Texture.Y / (FrameCount/Columns)
Size = StepSize
```

The following animates the train

```
<FlipAnim ID="TrainAnim" Size="100,80" Anchor="C,C" Columns="40"
Speed="10" Pause="0" EndPause="0" StepSize="10,0" FrameCount="40"
Texture="TrainAnim.dds" />
```

The StepSize attribute controls the offset of the "slice" taken from the main image. Columns and FrameCount control the total number of slices.

In general, for slice based animation

```
FrameCount = Texture.X / StepSize.X
StepSize.Y = 0
Columns = FrameCount
```

For vertical motion switch the X and Y dimensions.

A Speed of 0 is stopped, and a speed of 20 is fast - in-game flip animations use speeds between 10 and 20 depending on the number of frames in the sequence. If you only want something to play once and then stop, set Cycle="Once". Typically the Stopped="1" attribute is specified with a Cycle="Once" animation and then the animation started on some event from Lua with the Play() method. To reset the animation to its starting position (usually just before un-hiding it) use the SetToBeginning() method.

To specify a delay at the end of the sequence before it loops (only relevant for Cycle="Bounce") use the EndPause attribute.

Note on the Pause attribute

All the <XyzAnim> tags also take the Pause attribute, but its exact usage is currently unknown. It is mainly used on the shine alpha animations on the advisor buttons so probably introduces a delay at the end of the sequence before cycling. For flip animations it appears to behave the same as the EndPause attribute.

Summary

This tutorial has covered the Civ 5 User Interface (UI) elements and techniques for creating animations, and they are

- Animation
    - SlideAnim
    - ScrollAnim
    - FlipAnim
    - AlphaAnim
    - :Play() & :Stop()
    - :SetToBeginning()
    - FullClip="1" (attribute of ScrollPanel)

All of the XML and Lua code for the examples in this tutorial can be downloaded from the Mod Hub as "Test - UI Tutorial - 7 Animations" (in the Other category).

Part 8 of this series of UI Tutorials wraps up by covering the infrequently used elements that haven't been discussed elsewhere.