



# Intelligent Picture Viewer

The Scripting Your World Team

1 November 2008

[authors@syw.fabulo.us](mailto:authors@syw.fabulo.us)

<http://syw.fabulo.us>

# The Challenge

- Create controllable textured surfaces
- Example Uses:
  - Picture frame for your walls
  - Building tools
  - Advertising
  - Publishing
  - Vendors

# Let's Build

- First, create a box.
  - Right click on the background
  - Select Create from the pie menu
  - Then click on the ground near you
- If you get an error of some sort, you probably don't have permission to build in the area.
  - SYW HQ has a public sandbox at  
<http://slurl.com/secondlife/Hennepin/38/138/108/>

# Prim Preparation

- Rename your object
  - On the *General* tab, change the *Name* to be something like “Picture Frame”
- Adjust your prim’s size to be somewhat more like a picture.
  - On the *Object* tab, adjust the  $X = 1.0$ ,  $Y = 0.01$ , and  $Z = 1.0$
- Place some textures into its inventory.
  - Click on the Frame’s *Content* folder
  - Open your Inventory, select some textures (there are lots in the Library)
  - Drag-and-drop them into the *Content* folder

# Script Creation

- Create its script.
  - In the *Content* tab, click the *New Script* button.
  - Then double click on the “New Script” to open an editor

# Script Tinkering (1)

- Delete the line

```
llSay(0, "Hello, Avatar!");
```

from the `state_entry()` event handler.

- `llSay()` on the public channel 0 puts a heavy load on the sim. Save `llSay()` for conscious use when you really want your script to talk to everyone around you, not as an excuse because you haven't bothered to fix it.
- Alternatively, `llOwnerSay("whatever")` is generally a better choice for debugging than `llSay(0, "whatever")` because it speaks only to the owner of the prim.

## Script Tinkering (2)

- Make the picture responsive to anyone's touch
- In the `touch_start()` event handler, you need to get the texture's name:

```
string name = llGetInventoryName(  
    INVENTORY_TEXTURE, 0);
```

- And then display it:

```
llSetTexture(name, ALL_SIDES);
```

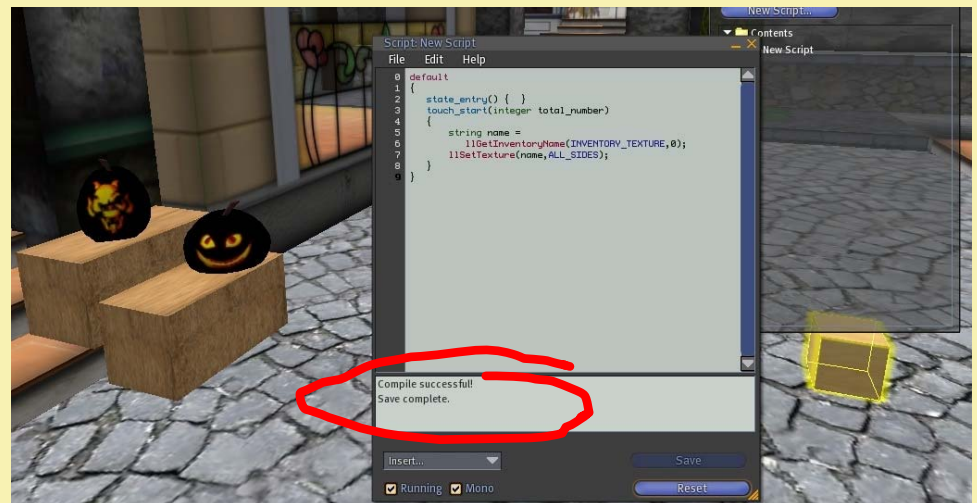
# Script Done!

- The completed script (#1) should look like

```
default
{
    state_entry() {
    }
    touch_start(integer total_number)
    {
        string name =
            llGetInventoryName( INVENTORY_TEXTURE, 0 );
        llSetTexture( name, ALL_SIDES );
    }
}
```

# Save and Run

- Click Save in the edit window.
- Assuming that you haven't made typos, (and your code looks just like the previous screen), then it should say "Compile successful, saving... Save complete."



# Script Testing

- Touch your Picture Viewer and it will display the first texture in inventory
- Touch it again, what happens? No change? That's because we're always showing the *first* texture (0)
  - LSL uses 0-based indexes (almost) exclusively
- Next, Let's enhance the viewer!

# Extended Script Tinkering (1)

- Now you're going to extend the viewer to select a different random texture at each click.
- First you need to know how many textures are in inventory.
- Add this line at the beginning of the `touch_start()` event handler:

```
integer n =  
  llGetInventoryNumber( INVENTORY_TEXTURE );
```

# Extended Script Tinkering (2)

- You'll want to select a random number between zero and  $n$ , where  $n$  is number of textures in inventory (from the previous slide)
- In contrast, `llGetInventoryName()` wants an *index*, so 0,1,2,3,4,5 are the possible legal values of  $y$  if there are 6 textures
- `llFrاند(n)` returns a random number that can be as low as 0 and almost but not quite  $n$ .

```
integer y = (integer) llFrاند(n);
```

If `llFrاند(6)` returns 5.999999,  $y$  will be truncated to 5

- Now that  $y$  is a legitimate index,  

```
string name = llGetInventoryName  
(INVENTORY_TEXTURE, y);
```

SW

# Script (re)Done!

- Script #2 should look like

```
default
{
  state_entry() {
  }
  touch_start(integer total_number)
  {
    integer n = llGetInventoryNumber(INVENTORY_TEXTURE);
    integer y = (integer) llFrand(n);
    string name =
      llGetInventoryName(INVENTORY_TEXTURE, y);
    llSetTexture(name, ALL_SIDES);
  }
}
```

# Save and Run (again)

- Click Save in the edit window. Assuming that you haven't made typos, and your script look like script 2, then the script console should say "Compile successful, saving... Save complete."
- You can touch your object & it will page randomly between images. (Note that it might sometimes choose the same image in a row)

# Advanced Topics—Preloading

- Under normal circumstances, you'll notice pretty quickly that texture loading is very noticeable (fuzzy images)
- **However**, the busier the sim, and the more textures in your viewer, (or the bigger the textures) the sloooooooooower it gets.

# Advanced Topics—Preloading

- The remedy: trick the viewer into downloading the textures before it really needs it
- Display the next texture on an object in view but that will not be noticed
- For instance, load the new image on the back, and when it's ready, move it to the front and put the next one on the back!

# Preloading Discussion

- Create a variable to cache the name of the texture you've preloaded. This should be the very first line in the script:

```
string gCachedTexture;
```

- In `state_entry()`, initialize its value with the first texture in inventory:

```
gCachedTexture =
```

```
    llGetInventoryName( INVENTORY_TEXTURE, 0 );
```

- Next, place the following line at the end of the `touch_start()` handler: `gCachedTexture = name;`
- (You might want to Save at this point to make sure that you have no typos.)

## Preloading Discussion (2)

- Now, to make the precaching work, you'll need to tweak the calls to `llSetTexture()`
- First, change the value for `ALL_SIDES` in the call to `llSetTexture()` at the end of the event handler to be the number 3

Face 3 is the back of the box.

- Second, add the line:  
`llSetTexture(gCachedTexture, 1);`  
at the beginning of the `touch_start()` handler

# A Preloading slideshow

- Script #3 should look like

```
string gCachedTexture;

default
{
    state_entry() {
        llSetColor(<1,0,0>,3); // usually make this <0,0,0>
        gCachedTexture=llGetInventoryName(INVENTORY_TEXTURE,0);
    }
    touch_start(integer total_number) {
        llSetTexture(gCachedTexture,1); // Front
        integer n = llGetInventoryNumber(INVENTORY_TEXTURE);
        integer y = (integer) llFrاند(n);
        string name = llGetInventoryName(INVENTORY_TEXTURE,y);
        llSetTexture(name, 3); // Back
        gCachedTexture = name;
    }
}
```

# Advanced Topics:

## Unidirectional Slideshow

- Keep a counter with a variable outside the default state
- On each touch, display the current counter, and then add one to advance
- Wrap back to 0 when you reach the number of textures
- Maybe use the new LSL touch functions to go forward and backward

# Advanced Topics: Transition Effects

- Display the old and new texture on separate prims layered closely together
- Expose the new one or hide the old one
- For a fade effect, change the alpha of the front face to make the back visible
- For a wipe effect, move the back forward with a little rotation around the z axis