bpy: Blender's Sweet New Python API

Christopher Allan Webber

2011-03-12 Sat

Outline

Blender features

Blender is a fully featured 3d suite.
Usable for:

- Modeling
- Texturing
- Rendering
- Animating
- Compositing
- Video editing
- Most 3d things
- Game engine??? (separate python API!)

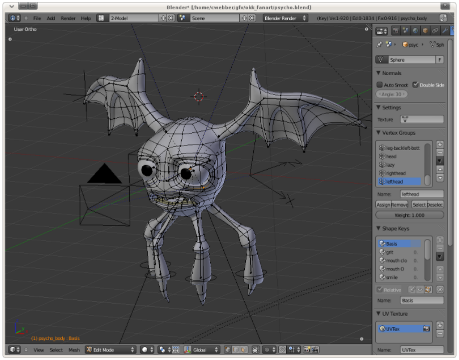
All free software, under the GPL

A few brief clips

Show the clips!

Let's look at psycho.blend!

Super briefly!
Look at a file so we can grasp Blender's basics. . .


Blender* [/home/cwebber/gfx/okk_fanart/psycho.blend]

(Key) Ve:1-920 | Ed:0-1834 | Fa:0-916 | psycho body

Auto Smoot ✓ Double Side
Angle: 30
Name: lefthead
Weight: 1.000
Name: Basis
Name: UVTex

(1) psycho_body : Basis

So what is bpy?

- **bpy** is Blender's new Python API
- Pretty much a complete overhaul of the Python API in 2.5X from the 2.4X and previous API
- A lot of it is "automatic" thanks to the **data api** and **tools api** of Blender 2.5X
- Huge addon community (think, similar to firefox's)
- **Python 3.X** and 3.X only!
- *Contrast to Blender 2.4x*
    - over 100,000 lines for Python/C API code
    - hard to maintain
    - often duplicated functionality of UI

DNA and RNA: the guts of a .blend

We can explore blender's datastructure / data api. It's easy, and all here!
Let's knock down some terms:

- **DNA**
  - Blender's internal datastructure
  - Backwards and forwards compatible! (mostly)
- **RNA**
  - A wrapper around Blender's DNA
  - Automatic free access to blender's datastructure! Wowee!
  - New in 2.5X!
- **bpy.data**
  - The portion of bpy that lets you access the datastructure from Python

Time to dive in

Let's try finding and changing some data.
This is easy thanks to our friend the **datablock outliner**.

What are operators?

Blender's tools api!
Simultaneously:

- Executable tools
    - from UI
    - from other python scripts
- UI dialogs (and, as buttons, elements)
- Almost any action you do in blender is some operator

Blender 2.5 is self-documenting, operators included!

- Operators are kept inside of **bpy.ops**
- Your actions are logged! Finding operators is easy!
- Hovering over UI elements helps you find the python equivalents!
- Let's test this inside of blender :D

Overview of an operator

```
import bpy

class ExampleOperator(bpy.types.Operator):
  bl_idname = "wm.example_operator"
  bl_label = "Example Operator"

  mouse_x = bpy.props.IntProperty()

  def execute(self, context):
    # The 'action' of the operator, what happens when called
    print("hello world!")
    return {'FINISHED'}

  def invoke(self, context, event):
    # Called first when invoked from UI (button/keypress),
    # has extra info like mouse data, etc
    self.mouse_x = event.mouse_x
    return self.execute(context)

  def draw(self, context):
    # Custom drawing interface.
    # If not used, we get an auto-UI from our properties
    pass
```

Panels in the UI

- Pretty much the same as scripting the operator. . .
- But for making UI panels.
- There's only a draw() method though.

Panel operator

```
class ReferenceDeskPanel(bpy.types.Panel):
    bl_label = 'Reference Desk'
    bl_space_type = 'VIEW_3D'
    bl_region_type = 'TOOLS'

    def draw(self, context):
        layout = self.layout
        row = layout.row()
        row.prop(context.scene, 'refdesk_search',
                 text="", icon='VIEWZOOM')
        # etc...
```

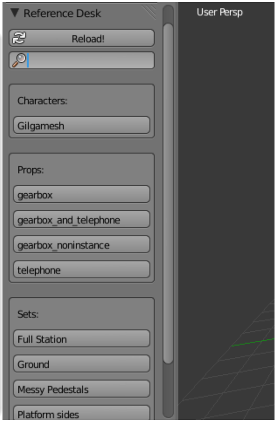
Reference Desk example

Later in the ReferenceDeskPanel.draw() method. . .
Creating buttons iteratively
Each button is an operator!

Source code

```
for item_name in item_names:
    item_data = section_data[item_name]

    row = box.row()
    menuitem = row.operator(
        'refdesk_menuitem',
        text=item_name)
    menuitem.section = section_name
    menuitem.item_name = item_name
```



Packaging?

- No time to discuss
- But not like python's packaging
- Search for Addons on http://wiki.blender.org

Some real-world examples

If we have time, let's look at these!

- Gilga rig
- Patent absurdity monstrosity
- ???

The future?

- **import bpy** from python, without blender open! (experimental)
- Full access to the event system
- a separated game engine (not part of this talk, but anyway)

Thanks

- Ton Roosendaal & the Blender Foundation
- Campbell Barton, leading awesome new python api
- Bassam Kurdali, answering stupid questions
- Blender's incredible community
- Creative Commons, for being a great place to work and encouraging, also awesome in general
- Emacs' org-mode and latex-beamer export, for making slide-making enjoyable

In conclusion / Where from here

- Check out http://blender.org
- The UI is great, but different! Practice, and it'll feel like home
- Check out http://wiki.blender.org and check out the 2.5 python api examples
- Check out Blender Foundation Films: *Sintel*, *Big Buck Bunny*, *Elephants Dream*

Contact me!

- **email / XMPP:** cwebber@dustycloud.org
- **other:** http://dustycloud.org/contact/