The Joy of Scripting on OS X

Simon Cozens
Kasei
Scripting

- Do you just mean ‘programming’?

- Interpreted programming?

- Interacting with applications?
The Joy of OS X

• Diversity
• Power
• Scriptability
Scripting on OS X

- AppleScript
- Tcl/Tk
- PHP
- Python
- Ruby
- Perl

The Apple
by Jimmy Brunelle

(LIGHTS UP ON A ROOM IN AN ART MUSEUM. THERE ARE VARIOUS PAINTINGS ON THE WALLS. UPSTAGE, THERE ARE A FEW BLACK PEDESTALS WITH FUNKY SCULPTURES ON THEM. AN EMPTY, WHITE PEDESTAL LIES CENTER STAGE. IT IS BRIGHTLY LIT FROM ABOVE. THERE ARE A FEW BENCHES ON EITHER SIDE OF THE WHITE PEDESTAL. A JANITOR SWEEPS THE FLOOR. HE LOOKS AROUND AT THE ART. HE CHECKS TO SEE IF ANYONE IS LOOKING AND THEN SPEAKS TO THE AUDIENCE)

JANITOR: Hello, invisible spectators of my life. I know you're there, mocking my simple existence. All you see is a menial servant, but that is an illusion. For within this—shell—is an infinity of creative potential. This—art—that you see surrounding me will someday bow down in fear and respect of the works I will someday manifest. My creations will move people to take action, not only in their lives, but to change the world for the better. I just need to find my medium. Is it painting? Comedy? People made of macaroni? Nah, no macaroni—my mother spoiled that one. I made an elbow made of elbows. She didn't get it. Gave me a triple time-out for wasting food. No matter, I will be a great artist someday. Someday. I will. Ah, who am I fooling? (TAKES AN APPLE FROM HIS POCKET AND RUBS IT ON HIS SHIRT. MOOCHER ENTERS AND INTERRUPTS HIM JUST AS HE IS ABOUT TO TAKE A BITE)


JANITOR: Why do you need a pen?

MOOCHER: Gotta have a pen needa pen.

JANITOR: What for?

MOOCHER: Don't gotta pen needa pen.

JANITOR: You know, sir, there's no PEN-handling allowed in the museum.

MOOCHER: Thatta joke thatta joke? Don't needa joke.

JANITOR: Panhandling for pens. PEN-Handling. Pen--
AppleScript

- Isn’t that some Mac thing?
- Scripting ‘for the rest of us’
- Exposes AppleEvent hooks to applications
AppleScript

- `osascript -e '...'`
- Used to be slow as hell
- Panther makes it all better
tell application "iTunes"
    if player state is playing then
        set thisTrack to current track
        set currentRating to thisTrack's rating
        if currentRating is less than 81 then
            set thisTrack's rating to currentRating + 20
        else -- can't set any HIGHER
            beep
        end if
    end if
end tell
Tcl / Tk

• I know nothing about this

• Sorry

• Aqua-native versions are available

• This does not mean all Tks are Aqua-native

• Sorry again
Native Tcl/Tk

#!/usr/local/bin/wish4.1
proc create_message { textvar } {
    if {[winfo exists .top]} { destroy .top }
    toplevel .top
    wm geometry .top +90+25
    message .top.msg -relief raised -bd 2 -text $textvar -justify center
    pack .top.msg
}

toplevel .but
wm withdraw .
wm geometry .but +5+25
frame .but.buttons -relief ridge -bd 2
button .but.buttons.ok -text "Exit" -command { exit }
button .but.buttons.help -text "Help" -command {
    create_message "Help, I'm being eaten by my socks"
}
button .but.buttons.fine -text "Fine" -command {
    create_message "I'm Just Fine, now get away from me"
}
button .but.buttons.shoo -text "Shoo" -command {
    create_message "SHOO, GET AWAY FROM THE COMPUTER..."
}
pack .but.buttons.help .but.buttons.fine .but.buttons.shoo .but.buttons.ok
pack .but.buttons
tkwait window .but
PHP

• I know nothing about this either

• Sorry

• But it’s apparently some web thing

• Which isn’t that exciting, really
Python

- OO, general-purpose language
- Useful for teaching
- They have an Aquaified Tk too
from xml.sax.handler import ContentHandler
import xml.sax
import sys

class textHandler(ContentHandler):
    def characters(self, ch):
        sys.stdout.write(ch.encode("Latin-1"))

parser = xml.sax.make_parser()
handler = textHandler()
parser.setContentHandler(handler)
parser.parse("test.xml")
Ruby

- I Like Ruby
- Completely OO language
- Elements of Perl, Python, CLS, ...
def get_appropriate_paras
    @app_paras = []
    my_keywords = @keywords.clone
    @text_a.sort{|a, b| b.final_score <=> a.final_score}.each do |x|
        words = x.best_keywords & my_keywords
        if (words.length > 0)
            my_keywords -= words
            x.marked_words <<= words
            @app_paras <<= x
            break if my_keywords.empty?
        end
    end
end

@app_paras.sort!{|a, b| a.order_of_appearance <=> b.order_of_appearance}
Perl

- Rather well-known programming language
- Excellent OS X integration
- And I have somewhat of a bias
use Mac::Glue;

my $itunes = Mac::Glue->new("iTunes");
if ($itunes->prop('player state')->{get eq "playing"}) {
    my $rating = $itunes->prop('current track')->prop('rating');
    $itunes->set($rating => to => ($rating->get + 20))
    if $rating->get < 81;
}
use Mac::Glue;

$f = new Mac::Glue 'Finder';
$volume = $f->prop('startup disk');
$label = $f->prop('label_index', $volume);

$f->activate;
for (0..8) {
  $f->set($label, to => $_);
  sleep 1;
}
$f->set($label, to => 0);
Mac::Glue

- So far we’ve dealt with getting and setting properties
- But Mac::Glue also gives us verbs
- ‘Mac::Glue->new("iTunes")->fast_forward’
Remote Events

perl -MMac::Glue -e 'Mac::Glue->new("iTunes")->fast_forward

perl -MMac::Glue -e 'Mac::Glue->new("iTunes",
eppc => iTunes => "desktop.example.com")->fast_forward'
Other Mac::Modules

- Mac::iTunes
- Mac::iPod::DB
- Mac::Speech
- (Mac::SpeechRecognition)
- ...
- ...
What about GUIs?

- CamelBones embeds Perl in Cocoa
- Which is generally what you want
- Unless you’re running Panther
- I’m running Panther
Perl in Cocoa

sub showPodFile {
    my ($self, $podFile) = @_; 
    my $document = $self->document();

    $self->{'StatusText'}->setStringValue('Reading file...');
    $self->{'StatusText'}->display();

    if ($document->readFromFile_ofType($podFile, '')) {
        my $tv = $self->{'TextView'};
        $tv->textStorage()->setAttributedString($document->{'_string'});
        $tv->scrollRangeToVisible(NSMakeRange(0, 0));
    }
}
RubyCocoa

• Same idea, different language
• No Panther support yet either, sadly
RubyCocoa

def initialize_drawer
    rect = @drawercontent.frame
    drawer = OSX::NSDrawer.alloc.initWithContentSize(preferredEdge:
        rect.size, drawer_edge)
    drawer.setContentView(@drawercontent)
    drawer.setParentWindow(self.window)
    drawer.open

    @drawer = drawer
end
PyObjC

- Embeds Python in a Cocoa framework
- Requires good Cocoa knowledge
- I could even get this one to work
Resources

• AppleScript: http://www.apple.com/applescript

• Aqua Tk: http://www.maths.mq.edu.au/~steffen/tcltk/TclTkAqua/

• Python: http://www.python.org/

• PyObjC: http://pyobjc.sourceforge.net/
Resources

• Ruby: http://www.ruby-lang.org/

• RubyCocoa: http://www.imasy.or.jp/~hisa/mac/rubycocoa/

• Perl: http://www.perl.com/

• Mac::Glue: http://dev.macperl.org/

• CamelBones: http://www.dot-app.org/
Metaresource

Thank You