Introduction to PyObjC

Author
Bob Ippolito

Conference
PyCon DC, March 2005
Intended Audience

- Python developers using Mac OS X 10.3 or later
- Spies from the Linux and Win32 camps
- Hopefully a GNUstep porter/maintainer
Topics

- Installing PyObjC
- Why Bother?
- Objective-C Primer
- Crossing the Bridge
- Interface Builder
- Your First Application
- Help!
- Who's Using This Stuff?
Installing PyObjC

Install Xcode:
  http://developer.apple.com/
Install PyObjC:
  http://pyobjc.sourceforge.net/
Why Bother?

- You paid for that Mac
- The tools kick ass
- Apple (often) writes good code
- The tools kick ass
- Objective-C and Python are friends
Objective-C

- True superset of C
- Everything is not an object
- Looks kinda like Smalltalk
Classes

- Flat Namespace
- Single Inheritance
- ... with Categories and Protocols
- Classes are objects
- Instance Variables
@interface MyClass : NSObject
{
    int myInt;
}
+(id)myClassWithInt:(int)anInt;
-(int)myInt;
@end
@implementation MyClass

+(id)myClassWithInt:(int)anInt;
{
    self = [[self alloc] init];
    intInstanceVariable = anInt;
    return self;
}

-(int)myInt
{
    return myInt;
}

@end
Objects

- Separate alloc/init
- Everything is an accessor
- ... except when using Key-Value Coding
- Reference counted
- ... but we take care of that
- ... except where Apple doesn't
Messages

- Target
- ... can be nil
- Selector
- Arguments
Exceptions

- Exceptions are exceptional
- Expect bad code to just crash
- ... even from Python
Crossing the Bridge

- unicode, int, long, float work magically
- ... str is not safely bridged!
- None is just like nil
- ... except you can't send messages to it!
Objective-C Messages

Objective-C Message:

```
[aMutableArray addObject:@"someObject"]
```

Target:

```
aMutableArray
```

Selector:

```
addObject:
```

Arguments:

```
@"someObject"
```
PyObjC Messages

Python Message:

```
import PyObjC

aMutableArray.addObject_(u'someObject')
```

Target:

```
aMutableArray
```

Selector:

```
addObject: (with colons replaced by underscores!)
```

Arguments:

```
u'someObject' (unicode is equivalent to @"string")
```
Key-Value Coding

- Kinda like `getattr` protocol
- ... but it calls accessors for you (like `property`)
- ... or it will fetch an ivar and convert to an object
- `valueForUndefinedKey:` (like `__getattr__`)
- `valueForKeyPath:` looks like a Python expression
- ... except it will also "map" over arrays
- ... and can do cool things like `sum`
Interface Builder

- Design your interface
- ... using a well designed interface
- Don't write so much code
- Plug objects together
- Manages an *object graph*
- ... think pickle
Making Money

- Currency Converter
- Using Cocoa Bindings
- Almost entirely in Interface Builder
New Application in IB
Create an NSTextField
Drag to the NSWindow
Create the input NSTextField
Almost finished UI Layout
Align the labels
Use currency NSNumber Formatters
Set up the Bindings
To point to your delegate
Dollars binding...
Other Currency Binding...
Subclass NSObject
To create your delegate class
Instantiate it in your nib
Create a connection
To the NSApplication
from Foundation import *
from AppKit import *
import objc

class ConverterAppDelegate(NSObject):
    def init(self):
        self = super(ConverterAppDelegate, self).init()
        self.exchangeRate = 3
        self.dollarsToConvert = 4
        return self

    def amountInOtherCurrency(self):
        return self.dollarsToConvert * self.exchangeRate

    def setAmountInOtherCurrency_(self, amt):
        self.dollarsToConvert = amt / self.exchangeRate

    # shamelessly preventing line wrapping
    cls = ConverterAppDelegate
    cls.setKeys_triggerChangeNotificationsForDependentKey_(
        [u'dollarsToConvert', u'exchangeRate',
         u'amountInOtherCurrency',
        ]
)
from PyObjCTools import AppHelper
import ConverterAppDelegate

if __name__ == '__main__':
    AppHelper.runEventLoop()
from distutils.core import setup
import py2app
setup(
    app = ['Converter.py'],
    data_files = ['MainMenu.nib'],
)
Build and Run

Build:

    % python setup.py py2app --alias

Run:

    % open dist/Converter.app

Done:

![Exchange Rate Window]

<table>
<thead>
<tr>
<th>Currency</th>
<th>Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>$</td>
<td>4.00</td>
</tr>
<tr>
<td>€</td>
<td>12.00</td>
</tr>
</tbody>
</table>
Hack the Gibson

- Views password file
- ... using nidump utility
- In a table view
New NSTableView
Name the columns

<table>
<thead>
<tr>
<th>user</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Cupertino</td>
</tr>
<tr>
<td>1</td>
<td>San Jose</td>
</tr>
<tr>
<td>2</td>
<td>Santa Clara</td>
</tr>
<tr>
<td>3</td>
<td>San Francisco</td>
</tr>
<tr>
<td>4</td>
<td>Palo Alto</td>
</tr>
<tr>
<td>5</td>
<td>San Carlos</td>
</tr>
<tr>
<td>6</td>
<td>Los Gatos</td>
</tr>
<tr>
<td>7</td>
<td>Sunnyvale</td>
</tr>
<tr>
<td>8</td>
<td>Mountain View</td>
</tr>
<tr>
<td>9</td>
<td>Redwood City</td>
</tr>
</tbody>
</table>
Change the resize behavior
To expand with the NSWindow
Create an NSArrayController
Create the ViewerAppDelegate
Bind the NSArrayController
Bind the user column
Bind the uid column
from PyObjCTools import AppHelper
from Foundation import *
from AppKit import *
import os

# another shameless anti-line-wrapping hack
FIELDS = ""
user password uid gid class change
directory gecos home_dir shell
"".split()

class ViewerAppDelegate(NSObject):
    def init(self):
        self = super(ViewerAppDelegate, self).init()
        self.passwords = [
            dict(zip(FIELDS, line.rstrip().split(':')))  # another shameless anti-line-wrapping hack
            for line in os.popen('/usr/bin/nidump passwd .')
            if line and not line.startswith('#')
        ]
        return self

if __name__ == '__main__':
    AppHelper.runEventLoop()
Build and Run Viewer

Build (redistributable!):

% py2applet Viewer.py MainMenu.nib

Run:

% open Viewer.app

Done:
Bindings give you sorting for free!
Help!

Documentation:
/Developer/Python/PyObjC/Documentation

Examples:
/Developer/Python/PyObjC/Examples

Wiki:
http://pythonmac.org/wiki

IRC:
#macpython (on freenode)

Mailing Lists:
- pyobjc-dev@lists.sourceforge.net
- pythonmac-sig@python.org
Help! (Objective-C)

Documentation:

http://developer.apple.com/

Examples:

/Developer/Examples/AppKit

Wiki:

http://cocoadev.com/

Mailing List:

cocoa-dev@lists.apple.com
Introduction to PyObjC

Author
Bob Ippolito

Conference
PyCon DC, March 2005

Intended Audience
- Python developers using Mac OS X 10.3 or later
- Spies from the Linux and Win32 camps
  - Hopefully a GNUstep porter/maintainer

Topics
- Installing PyObjC
- Why Bother?
- Interface Builder
- Objective-C Primer
- Crossing the Bridge
- Your First Application

Intended Audience
- Python developers using Mac OS X 10.3 or later
- Spies from the Linux and Win32 camps
  - Hopefully a GNUstep porter/maintainer

Topics
- Installing PyObjC
## Flame

<table>
<thead>
<tr>
<th>Host</th>
<th>Service</th>
</tr>
</thead>
<tbody>
<tr>
<td>Andrew Dalke's Computer [00:0a:95:68:26:c8]</td>
<td>(1)</td>
</tr>
<tr>
<td>Andrew Gross (10.0.43.224)</td>
<td></td>
</tr>
<tr>
<td>iChat 2 presence</td>
<td>Andrew Gross</td>
</tr>
<tr>
<td>Remote login</td>
<td>mitya</td>
</tr>
<tr>
<td>Personal file sharing</td>
<td>mitya</td>
</tr>
<tr>
<td>Workgroup Manager</td>
<td>mitya [00:11:24:73:74:88]</td>
</tr>
<tr>
<td>iTunes shared music</td>
<td>arg</td>
</tr>
<tr>
<td>iTunes remote control</td>
<td>iTunes_Ctrl_9DF57C44A0</td>
</tr>
<tr>
<td>Bob Ippolito (10.0.40.155)</td>
<td></td>
</tr>
<tr>
<td>Daniel Krech (10.0.43.214)</td>
<td></td>
</tr>
<tr>
<td>David Goodger's Computer (10.0.40.157)</td>
<td></td>
</tr>
<tr>
<td>Drifty's Computer [00:0d:93:c5:a0:b6] (10.0.40.1)</td>
<td></td>
</tr>
<tr>
<td>Ian Bicking's Computer (10.0.42.153)</td>
<td></td>
</tr>
<tr>
<td>Remote login</td>
<td>Ian Bicking's Computer</td>
</tr>
<tr>
<td>Personal file sharing</td>
<td>Ian Bicking's Computer</td>
</tr>
<tr>
<td>Workgroup Manager</td>
<td>Ian Bicking's Computer [0]</td>
</tr>
<tr>
<td>FTP server</td>
<td>Ian Bicking's Computer</td>
</tr>
<tr>
<td>Web server</td>
<td>Emily Murphy</td>
</tr>
<tr>
<td>_MacOSXDupSuppress._tcp.</td>
<td>-366817258; -36681725</td>
</tr>
<tr>
<td>James Knight's Computer [00:0a:95:a5:0f:b2]</td>
<td>(1C)</td>
</tr>
<tr>
<td>Linden Wright (10.0.43.159)</td>
<td></td>
</tr>
<tr>
<td>MailMaster [00:0a:95:ca:1e:cc] (10.0.41.184)</td>
<td></td>
</tr>
<tr>
<td>Nicholas Bastin's Computer [00:0d:93:29:27:fe]</td>
<td>(</td>
</tr>
</tbody>
</table>
NodeBox

```python
size(300, 300)
white = color(1,1,1,0.9)
red = color(1,0,0,0.9)
black = color(0,0,0,0.9)

for i in range(100):
    fill(choice((white, red, black)))
    font("Arial Bold")
    fontsize(random(300))
    text(u"TM", random(250), random(200))
    text(u"®", random(250), random(200))
    text(u"®", random(250), random(200))
```
Questions?

Go ahead, ask.