/dev/world/2010

Your App Universe Down Under
28-29 September
Rydges Melbourne
AppleScriptObjC: Good News for Objective-C Coders, too

Shane Stanley
Myriad Communications Pty Ltd

<sstanley@myriad-com.com.com.au>
Inter Application Communications

• Getting information from other applications
Inter Application Communications

• Getting information from other applications

• Allows treating other applications as “super frameworks”
Inter Application Communications

- Getting information from other applications
- Allows treating other applications as "super frameworks"
- Built on Apple events
Inter Application Communications

- Getting information from other applications
- Allows treating other applications as “super frameworks”
- Built on Apple events
- AppleScript and Apple events were designed around each other
Apple Events from Objective-C

- **NSAppleEventDescriptor & AESendMessage() (10.0)**
Apple Events from Objective-C

- NSAppleEventDescriptor & AESendMessage() (10.0)
- NSAppleScript (10.2)
Apple Events from Objective-C

- **NSAppleEventDescriptor** & **AESendMessage()** (10.0)
- **NSAppleScript** (10.2)
- **Scripting Bridge** (10.5)
Apple Events from Objective-C

- NSAppleEventDescriptor & AESendMessage() (10.0)
- NSAppleScript (10.2)
- Scripting Bridge (10.5)
- objc-appscript (third-party)
Apple Events from Objective-C

- NSAppleEventDescriptor & AESendMessage() (10.0)
- NSAppleScript (10.2)
- Scripting Bridge (10.5)
- objc-appscript (third-party)
- AppleScriptObjC (10.6)
AESendMessage()

• Fast and efficient
AESendMessage()

- Fast and efficient, but complicated

```c
const char *signature = "com.apple.itunes";
OSErr err;
AEDesc replyDesc;
address = [NSAppleEventDescriptor descriptorWithDescriptorType: typeApplicationBundleID
            bytes: signature
            length: strlen(signature)];

event = [[[NSAppleEventDescriptor alloc] initWithClass: kAECoreSuite
            eventID: kAEGetData
            targetDescriptor: address
            returnID: kAutoGenerateReturnID
            transactionID: kAnyTransactionID];

ct = [NSAppleEventDescriptor recordDescriptor];
[ct setDescriptor: [NSAppleEventDescriptor descriptorWithTypeCode: 'prop' forKeyword: 'form'];
[ct setDescriptor: [NSAppleEventDescriptor descriptorWithTypeCode: 'prop' forKeyword: 'want'];
[ct setDescriptor: [NSAppleEventDescriptor descriptorWithTypeCode: 'pTrk' forKeyword: 'seld'];

ct = [ct coerceToDescriptorType: typeObjectSpecifier];
obj = [NSAppleEventDescriptor recordDescriptor];
[obj setDescriptor: [NSAppleEventDescriptor descriptorWithTypeCode: 'prop' forKeyword: 'form'];
[obj setDescriptor: [NSAppleEventDescriptor descriptorWithTypeCode: 'prop' forKeyword: 'want'];
[obj setDescriptor: [NSAppleEventDescriptor descriptorWithTypeCode: 'pnam' forKeyword: 'seld'];

[obj setDescriptor: ct forKeyword: 'from'];
 obj = [obj coerceToDescriptorType: typeObjectSpecifier];

[event setParamDescriptor: obj forKeyword: keyDirectObject];

err = AESendMessage([event aeDesc], &replyDesc, kAEWaitReply, kAEDefaultTimeout);
reply = [[[NSAppleEventDescriptor alloc] initWithAEDescNoCopy: &replyDesc] autorelease];
NSString *result = [[reply descriptorForKeyword: keyDirectObject] stringValue];
```
NSAppleScript

• Simple, but slow and limited

```objective-c
NSString *s = @"tell application \"iTunes\" to get name of current track";
NSAppleScript *script = [[NSAppleScript alloc] initWithSource:s];
NSString *result = [[script executeAndReturnError:nil] stringValue];
```
NSAppleScript

• Simple, but slow and limited
• Simple use is much slower
• Doesn’t integrate well with Objective-C
• Doesn’t scale well
Scripting Bridge

• Fast, efficient, simple
Scripting Bridge

- Fast, efficient, simple
- Generate header file and add to project
- Link to framework
- Write code

```objective-c
iTunesApplication *iTunes = [SBApplication applicationWithBundleIdentifier:@"com.apple.iTunes"];
NSString *result = [[iTunes currentTrack] name];
```
Scripting Bridge

- Fast, efficient, simple
- Generate header file and add to project
- Link to framework
- Write code

- Lots of shortcomings
- Incompatible with many applications

```
NSString *result = [[iTunes currentTrack] name];
```
objc-appscript

- Third-party framework
- Work-in-progress
- Need to generate objc-appscript dictionary documentation and glue code
AppleScriptObjC

- Bridges AppleScript and Cocoa
AppleScriptObjC

- Bridges AppleScript and Cocoa
- **Script objects become Cocoa classes**
AppleScriptObjC

• Bridges AppleScript and Cocoa
• Script objects become Cocoa classes
• **Scripts call methods as handlers**
AppleScriptObjC

- Bridges AppleScript and Cocoa
- Script objects become Cocoa classes
- Scripts call methods as handlers
- **Classes call handlers as methods**
AppleScriptObjC

- Bridges AppleScript and Cocoa
- Script objects become Cocoa classes
- Scripts call methods as handlers
- Classes call handlers as methods
- Fast, easy to use
AppleScriptObjC

• Bridges AppleScript and Cocoa
• Script objects become Cocoa classes
• Scripts call methods as handlers
• Classes call handlers as methods
• Fast, easy to use

• Lets languages play to their strengths
Implementing AppleScriptObjC

- Runs in 10.6 only
Implementing AppleScriptObjC

- Runs in 10.6 only
- Link to framework
Implementing AppleScriptObjC

- Runs in 10.6 only
- Link to framework
- Insert in main.c:

```c
#import <AppleScriptObjC/AppleScriptObjC.h>
[[NSBundle mainBundle] loadAppleScriptObjectiveCScripts];
```
Implementing AppleScriptObjC

• Add AS class, instantiate in .xib
Implementing AppleScriptObjC

- Add AS class, instantiate in .xib

- Use @class declaration, call handlers as methods (colons > underscores)
Implementing AppleScriptObjC

• Add AS class, instantiate in .xib

• Use @class declaration, call handlers as methods (colons > underscores)

• Use protocol to suppress compiler warnings
AppleScriptObjC in Action

AttachAScript.app (Apple sample code) and FrankenApp.app