

# programming Nu

John Labovitz

# an example

```
;; define the application delegate class
(class ApplicationDelegate is NSObject
  (imethod (void) applicationDidFinishLaunching: (id) sender is
    (build-menu default-application-menu "Console")
    (set $console ((NuConsoleWindowController alloc) init))
    ($console toggleConsole:self)))
```

# Nu is

- created by Tim Burks of RubyCocoa, RubyObjC
- interpreted & dynamic
- hybridized (Lisp+Smalltalk+Ruby+ObjC)
- directly coupled (to ObjC runtime)
- experimental & changing

# Nu ain't

- speedy
- all-purpose
- LISP, Scheme, etc.
- alternative to ObjC or Cocoa
- fixed & specified
- Xcode/IB-integrated (as shipped)

# it's a functional language

- atoms & lists
- expressions & evaluations (code as data)
- lambdas & functions
- contexts & closures
- macros

# in a world of objects

- classes
- instances
- methods
- ivars
- messages

# with the grit of C

- scalars (int, char, float)
- pointers
- #defines & extern symbols
- functions
- *“hell is other people”* (or their APIs)

# usual app structure

- Nukefile
- nu/
- objc/
- resources/

# usual app dev build

- in Terminal:
  - `nuke && open appname.app`
- doesn't build distributable app!
  - need `Nu.framework`

# playing with Nu

- nush

# Nu-Cocoa apps

- simple app: **RandomAppWithNibFile**
- Nu modules: **Console**
- bindings: **MailDemo**

# Nu-ObjC hybrids

- Nu app with ObjC extension: **Benwanu**
- ObjC bundle extended with Nu:  
**ScreenSaver**

# beyond Nu apps

- bundles
- NuAnywhere
- obfuscation

# beyond Cocoa apps

- web apps
- other systems

# installing Nu

- MacPorts: `sudo install nu`
  - edit `/opt/local/bin/nuke`
  - change `/usr/local` to `/opt/local`
- latest *git* tree: <http://code.neontology.com>

# resources

- <http://programming.nu>
- <http://groups.google.com/group/programming-nu>

???