

Offset	Topic
00:17	<ul style="list-style-type: none"> <li>• <b>Intro</b></li> </ul>
	<ul style="list-style-type: none"> <li>• Latest CopyNight <ul style="list-style-type: none"> <li>• The easy issues have been exhausted</li> <li>• I have to admit to a little fatigue</li> <li>• Often, the discussion also runs afoul of not strictly IP related issues</li> <li>• Trying to organize more of a speakers series</li> </ul> </li> </ul>
04:20	<ul style="list-style-type: none"> <li>• <b>Word of the Week: creeping</b></li> </ul>
	<ul style="list-style-type: none"> <li>• <a href="http://catb.org/jargon/html/C/creeping-elegance.html">http://catb.org/jargon/html/C/creeping-elegance.html</a></li> <li>• <a href="http://catb.org/jargon/html/C/creeping-featurism.html">http://catb.org/jargon/html/C/creeping-featurism.html</a></li> <li>• <a href="http://catb.org/jargon/html/C/creeping-featuritis.html">http://catb.org/jargon/html/C/creeping-featuritis.html</a></li> </ul>
06:47	<ul style="list-style-type: none"> <li>• <b>Hacking 101: Static vs. Dynamic</b></li> </ul>
	<ul style="list-style-type: none"> <li>• Static vs. dynamic <ul style="list-style-type: none"> <li>• Currently usually refers to newer scripting languages vs. older languages</li> <li>• Ruby, Python are dynamic</li> <li>• C is static</li> </ul> </li> <li>• Definition goes back further <ul style="list-style-type: none"> <li>• <a href="http://en.wikipedia.org/wiki/Name_binding">http://en.wikipedia.org/wiki/Name_binding</a></li> <li>• Simply refers to when a name is bound to a value</li> <li>• The name usually implies type, as well</li> <li>• In early binding, or static, languages type is part of reference declaration</li> <li>• Type is known at compile time, can be checked, enforced</li> <li>• Analysis tools can better determine what a program will do at runtime</li> <li>• In late binding, or dynamic, type of value not known until runtime</li> <li>• Often cannot be known</li> </ul> </li> <li>• Some "static" languages are actually a hybrid <ul style="list-style-type: none"> <li>• Polymorphism in OO relies on dynamic references</li> <li>• Reference type is usual more general</li> <li>• Exact, or virtual, type cannot be known until runtime as may be conditional</li> <li>• Still, general types are more static</li> <li>• Can even implement duck typing, see below, in static languages</li> <li>• Objective-C is implemented in C but has duck typing like Smalltalk</li> </ul> </li> <li>• Contrast to Ruby, Python <ul style="list-style-type: none"> <li>• Python often talks about duck typing</li> <li>• <a href="http://en.wikipedia.org/wiki/Duck_typing">http://en.wikipedia.org/wiki/Duck_typing</a></li> <li>• Type is interpreted by the properties accessed, methods invoked</li> </ul> </li> </ul>

## Offset

## Topic

- In reality, it is about method, messages
- Properties or data are incidental
- Actually also goes back a ways, to Smalltalk
- Any message can be sent to any object in Smalltalk
- Categories can be attached to objects at runtime
- In Objective-C if an object cannot receive a message, responds with an error
- Runtime attachment helps solve these gaps
- Takes dynamic typing to its logical conclusion
- Controversy
  - Dynamic advocates think you'll only provide objects that respond to known messages
  - Why worry about static type declarations?
  - Emphasis is on speed, ease of development
  - Static typing proponents see it as a safety check
  - Compiler can check, enforce typing earlier
  - Does require anticipation, prediction of needed types
  - Detractors think this is superfluous information why waste time?
  - Also with the rise of unit testing, dynamic fans think this can do what the compilers checking can do
  - Static fans wonder why do that when the compiler does it for free?
  - In the case of components, a library author needs to check, enforce type intentions
  - Why make the caller unit test this contract?
  - Why not use a hybrid solution, at least in this case, where the static binding express intent, prevents stupid errors?
  - As in most cases, I believe the answer does lie in between, depends on cases

28:30

## • Outro

- Contact me
  - Email to [feedback@thecommandline.net](mailto:feedback@thecommandline.net)
  - Web site at <http://thecommandline.net/>
  - IM to [command.line@skype](mailto:command.line@skype)
  - Listener comment line is 240-949-2638
  - del.icio.us tag is "for:cmdln"
  - <http://twitter.com/cmdln>
- I'd like to thank [libsyn.com](http://libsyn.com) for AAC hosting and Wouter de Bie for MP3 hosting
- These notes and the show audio and music are covered by a Creative Commons license
  - <http://creativecommons.org/licenses/by-nc-sa/3.0/us/>
  - Attribution, non-commercial, share alike