LIS-4708

Programming Languages

Know about BOTH..

What you hear about

v.

What people actually use

“Closeness” to machine

Compiled (older method) – Before running, you have to “convert” it. Usually Makes for faster/more efficent code.

Interpreted - “Conversion” happens on the fly.

Scripted – No “conversion” necessary. Usually only good for shorter/smaller/ scripted things.

Compiled Languages

- Older
- “Normal”
- “Close to machine”

“COMPILED” - Source Code

Compiled Code (binaries)

(Somewhat) Important "real life" paradigms. (these are fuzzy)

- Imperative/Procedural (default) - Do things step by step
- Functional – Turn it into straight up math. No variables, no "procedures"
- Object Oriented – Everything uses an "object" metaphor. ("Black Box Approach")
- Declarative – (kind of the grail, you just SAY it.)

Remember this?

(Frankly, the stuff in yellow gets jumbly)

Changing your screensaver

Writer/Word

Calc/Excel

Macros

HTML/PHP/MYSQL

Bash

Python/Perl/Ruby/Java/Javascript/C# *interpreted/scripted*

C *compiled*

Assembly (00 4E A3 77 8C 0A etc)

Machine (00101010011010101100)

Bash/Shell languages

For the purpose of interacting with your computer on a day-to-day basis..

..before guis even existed.

Bash/Shell languages

If you are looking to learn something new, I HEARTILY recommend Bash.

Bash is the “language” most strongly associated with the highest developer salaries.

You’ll learn about Linux by default. And thus Unix by default, which is ALL operating systems that aren’t Windows.

(and another bold prediction, this will soon apply to ALL OPERATING SYSTEMS, period.)

Categories I’m completely making up to tell the story:

REGULAR (programming and operating on the machine in front of your face)

WEB

WEIRD

Categories I’m completely making up to tell the story:

REGULAR =

The normal way to go is:

- Command line interaction
- an optional REPL
- a way to execute/mess with in a terminal
- No other languages/interfaces required

REGULAR

C (and C-alikes)

Java

Visual Basic

Python

Ruby

C (C++, C#)

(low-level machines, everywhere)
(Apple apps are Objective-C)

- Practical grandaddy of (modern) everything, the first “accessible” language
- Still a good choice for speed and control, at the cost of ease
- Later C-alikes tend towards “object oriented” and just “easier”
- "Why C sucks" -- TOO MUCH control, too much “hardcore stuff” to worry about (eg malloc)

Java

(cross-compatible, Android)

- First attempt at dethroning C, looks a lot like it
- Originally company driven, as a result got popular/useful quickly
- VERY PORTABLE – write once, works in all OSes
- interpreted, not compiled (Virtual Machine required)
- "Why Java Sucks" - slow, verbose, too many options/fragmented

REGULAR (but much nicer)

Python

Ruby

Python

- Forced whitespace
- Usually ''one way to do a thing''
- In the ''middle'' on just about all parameters.
- In a perfect world, we'd all be using Python..or better yet...

Ruby

- Pure Object-Oriented
- Really really nice syntax (3.times print “hello”)
- Unfortunately, a bit slow
- Built somewhat for the web, and yet...

THE WEB

(screwed everything up, language wise)

HTML made the web easy to read – but unfortunately, not easy to write.

HTML/CSS

- Not really languages. (Can't really do anything besides dress up and move around text and pictures and other things)

- Traditionally, for dressing up text because fonts and colors (and HYPERLINKS WHOA) used to be a pretty big deal.
- Today, not even a great choice if you need a website, BUT you should know it because it's now the FRAMEWORK for the web.

All the web terribleness can be explained by...

We want interactivity on the web

NOOOOOOOOW!!!!

No matter how unsafe or ugly or hacky!

Fail early! Fail Often!

Flash

Designed to make gaudy stupid animations and games, and was just hacked to death to do everything else.

PHP
(not bad actually. Server-side)

- Designed to handle the web and HTML
- Features were added as needed, not from ground up
- Not flashy, but solid for databasey type things.
- "Why PHP Sucks" Purists HATE IT; Arguably duct-tapey.

(still, facebook and wordpress)

(ASP is Microsoft's slightly different “PHP”)

All the web terribleness can be explained by...

I'll be a bit more charitable.

What language runs on pretty much anything you can buy or install, out of the "box?"

Javascript

- Though looks like c, VERY DIFFERENT FROM JAVA. Confusing, huh?
- CLIENT SIDE (mostly)….but
- JAVASCRIPT IS EATING THE WEB RIGHT NOW. LOOKING VERY DOMINANT.
- As a result, people are rewriting EVERYTHING in Javascript. Frequently poorly.
- Also, people are trying to do a half-hearted version of "open-source" (e.g., yes you can look at the code, but we control the repository/framework/direction/dependencies/company)

Javascript, wtf

So, this thing doesn't look like its stopping. Things to watch out for:

- Millions and millions of "frameworks on top of Javascript to make it suck less."
- And then, the reverse
- And also "server-side Javascript"

Weird and perhaps cool

SQL

Lisp and the Functionals

whitespace

Brainf**k

scratch

and a zillion others

Lisp (and other functionals)

(Haskell, Clojure)

- MATHY

- VERY, perhaps TOO “versatile”

- Stallman and Emacs

- Functional is seeing a resurgence, because of its mathematical “purity.” There are NO VARIABLES, WHOA.

SQL
(structured query language)

- Declarative!
- nice syntax
- yet, still another layer of complexity