

# LIS-5362

## Programming



Everyone needs to learn to code!!

?



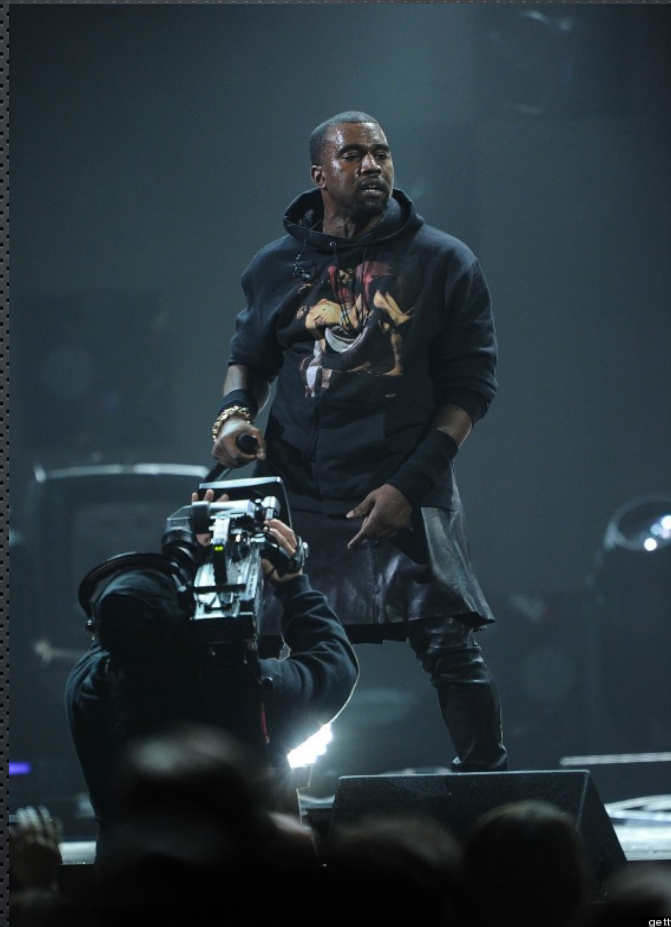
~~Everyone needs to learn to code!!~~

Everyone needs to learn  
about coding.



# “Popularity” can be misleading..

Languages are like fashion.. maybe the new hot thing will stick around, maybe not...(and vice versa)





# Know about BOTH..

What you hear about

v.

What people actually use



# WARNING:

Discussions about coding are very passionate  
and controversial..

Everyone thinks that their way is the best, and  
that anything not their way sucks.

(someone **might** be right...)



# Which is “primitive?”

```
Last login: Fri Aug 17 16:33:00 on ttys000
Valkyrie:~ whitsongordon$ sh top10.sh
Example Commands:
10. top
9. ifconfig /all
8. chmod +x newtop10.sh
7. ssh -l whitsongordon@192.168.0.12
6. wget http://lifehacker.com
5. vim todo.txt
4. grep top 10
3. ...
2.
1.

Valkyrie:~ whitsongordon$
```





# Why command line/text?

Because you can very quickly say/relate complex concepts in a concise way, by combining a series of simple symbols.

You know, like talking. Or writing.

Command line is the act of literally talking to the computer....unlike...



# What's so bad about the mouse again?

“Caveman interface.”

- Pre-linguistic/animal-like
- “Point and grunt”

(Tablets and even “Minority Report” are cool and fun...but why is Charades a game?)



It's so easy, even...





# Intelligence requires Language

Buttons and gestures are frequently convenient for repetitive tasks...

...but to do anything intelligent,  
you need LANGUAGE.

TEXT. Numbers and Letters.



So, “text”

- sane
- predictable
- simple
- infinitely useful and portable

...and not sexy at all.



# On UI (User Interface) and UX (User Experience)

- subjective
- frequently “anti-text”
- faddish
- as software; very changeable

...but – super sexy and sellable.



# Flat Design v. Skeuomorphism

## Which is better?

### Flat Design

August 2014						
SU	MO	TU	WE	TH	FR	SA
26	27	28	29	30	1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31	1	2	3	4	5	6

### Skeuomorphism Design

September 2013						
Sun	Mon	Tue	Wed	Thu	Fri	Sat
24	25	26	27	28	1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31	1	2	3	4	5	6



# Flat Design v. Skeuomorphism

## Which is better?

There are lots of professionals out there with very serious and thought out answers to this questions; and one can get quite far by being engaged with it...



# Flat Design v. Skeuomorphism

## Which is better?

There are lots of professionals out there with very serious and thought out answers to this questions; and one can get quite far by being engaged with it...

..and I'm not one of them.  
My super-cynical answer is

WHO CARES.



# Flat Design v. Skeuomorphism

## Which is better?

I do have a serious answer though:



Specifically: Who/what is preventing you from having a choice?



## Ps ..Why Linux?

- 1) Free and open
- 2) Language/Text Based

(These two are completely related; newer stuff tries to allow for more of 2 and/or 1; e.g. Mac OS X, Android)



Either way...

Yes – early text was ugly and unforgiving, and fixing that to various degrees propelled the big tech companies..

... but, along the way, restricting access to “back-end” text stuff helped create some of the messes we deal with today; including the oddness of programming languages.



# Human Readability ("Matrix" vs. "English")

Old school - Human readability is not important

- Short abstractions are concise and thus quicker
- Forced whitespace is limiting
- There should be *MANY* ways to do a thing.



# Human Readability

## ("Matrix" vs. "English")

New school – Hey, looks like human readability is at least a little important:

- Multiple people working on projects
- Older code needs to be understandable
- There is value in forcing people to do things only one way

*"Always code as if the guy who ends up maintaining your code will be a violent psychopath who knows where you live."*



# “Closeness” to machine

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

