Presentation Background Material (Silas Henderson)

Top languages to learn (for production)

Javascript: ← The Website Programming Language

Examples:	Any web-page written in the last 15 years.
Good:	Fast, Universal Support on all machines
Bad:	Not a 'real' programming language

Python: ← The Get-Stuff-Done Language

Examples:	Instagram, Google, Spotify, Netflix
Good:	Works with everything.
Bad:	Use libraries for everything.

C++ ← The Industry Standard Language

Examples:	Autodesk Maya, Adobe Photoshop
Good:	Fast, can do anything.
Bad:	Very time-consuming to write.

Top languages to learn (for cutting-edge)

WASM: ← Assembly Language for the Web

Examples:	AutoCad WebApp, Experimental
Good:	Run 'C/Fortran' in the browser
Bad:	Still in development.

Rust: ← A modern re-write of C++

Good:	Faster than C++, way less baggage
Bad:	More practical to just learn C++

Top languages to learn (for research, besides Matlab)

Python Stack ← C/C++ Numerical Computation called through Python

Good:	Reputable enough to publish research with. Faster than Matlab.
Bad:	Time-consuming to get everything set up.

Julia: ← A compiled language written specifically for scientific computation.

Good:	Fast, works with LLVM compiler (can directly see machine code)
Bad:	Not fully cooked yet. Advertises misleading 'speed tests'(concerning).

Ingredients of a Gui Application

Name		*What it does*		*What it needs to work well*
Window	←	Open a window	←	Communication with operating system
Graphics	←	Display graphics	←	Direct access to the graphics processor
Computation	←	Do calculations	←	Compile to machine code to directly run on cpu

Ingredients of a Bad Gui Application

Name		*What it does (badly)*
Window	←	Open program that opens another program than opens a window
Graphics	←	Graphics math on cpu, Send data to intermediate program, Send data to the graphics processor.
Computation	←	Do calculations within ‘scripted language’

Example Setup, Good (Matlab)

Window ← Java
Graphics ← OpenGL
Computation ← C

Example Setup, Bad (Any webpage during the mid 2000’s)

Window ← Flash-Plugin
Graphics ← 3rd-party add-on
Computation ← Javascript (before v8)

Options for writing a complete GUI computer program:

The Hard way: C++ with OpenGL *(Write everything from scratch)*

Example: Sublime Text

https://www.sublimetext.com/

Window ← Figure it out yourself
Graphics ← Write your own OpenGL
Computation ← Set up your own compiler specs

The Middle way: Java / Swift / C#, etc *(Write the important parts from scratch)*

Example: Minecraft

https://www.minecraft.net/en-us/

Window ← Faster Setup, Runs on different systems easier
Graphics ← Similar to C++
Computation ← Similar to C++

The Easy Way: Javascript! *(Just write the actual program)*

Example: Microsoft VS Code

https://code.visualstudio.com/

Window ← Internet Browser
Graphics ← WebGL
Computation ← Javascript

The Epic Way: WASM *(Write everything with new languages)*

Example: AutoCAD Web App

https://www.autodesk.com/products/autocad-web-app/overview

Window ← Anything
Graphics ← Vulkan
Computation ← Rust/WASM

Why Javascript?

Runs on any computer with firefox, chrome, safari, or internet explorer.

Fast computation (V8 JIT compilation)

Fast Graphics (direct 3d pipeline: webGL)

Has the most open source resources of any language

Javascript was bad. Now it is good

JS Performance, Past 20 years	*JS Performance Now*
Window ← Internet Explorer (bad)	Window ← Chrome, Firefox (excellent)
Graphics ← 3rd party-plugins (bad)	Graphics ← WebGL (excellent)
Computation ← Interpreted JS (bad)	Computation ← Compiled JS (½ speed of C)
	WASM (faster than C)

History of the Internet

1989: HTML ← make 'word documents'

1994: CSS ← add 'style' to documents

1995: Javascript ← add fancy style to documents

2008: V8 compiler ← Javascript is now 'real programming language'

2011: WebGL ← 3d graphics support in all browsers.

2015: WebAssembly ← Universal machine-code support in all browsers

How a webpage works

HTML	← the 'word document'
CSS	← the 'style sheet' (not needed)
Javascript	← the actual **programming language**

Generic Webpage

```
<html>                    ←  Say that this is a webpage (HTML opening tag)
   <head>  </head>        ←  Do some setup
   <body>  </body>        ←  Make the page
</html>                   ←  Close the document
```

Complete Webpage Example 1: Write some text (HTML)

```
<html> <body> This is some text </body> </html>
```

Complete Webpage Example 2: 3d Rotating Cube

```
<html> <body> <script src = "three.min.js"> </script> <script>

var renderer = new THREE.WebGLRenderer();
    renderer.setSize(window.innerWidth, window.innerHeight);
    renderer.setClearColor(0x151515);
document.body.appendChild(renderer.domElement);

var scene  = new THREE.Scene();
var camera = new THREE.PerspectiveCamera(
    45, window.innerWidth/window.innerHeight, 1, 1000);
camera.position.z = 100;

var light  = new THREE.PointLight();
    light.position.set(50, 50, 500);
var box = new THREE.Mesh(
    new THREE.BoxGeometry(20, 20, 20),
    new THREE.MeshPhongMaterial({color:0x446644}));

scene.add(light, box);

var loop = function() {
    requestAnimationFrame(loop);
    renderer.render(scene, camera);
    box.rotation.x += .01;
    box.rotation.y += .01;}
loop();

</script> </body> </html>
```

Python: The only *industry-proven* viable alternative to Matlab

Python is not a 'new language'. It is a 25-year-old, multi-purpose language that has already proven its worth in industry. It is automatically installed (along with c) on all Linux and OS devices. Companies like Google, Netflix, and Spotify already use python as their go-to programming language.

Python is popular because it is easy to read and reliable. For the applications, that **cannot crash**, python is the go-to language to use. In industry, python is used to be the stitch together large-scale software operations, *wrapping around the other softwares, like a python*.

Because python is good at 'wrapping', it is also a good interface for scientific computation. The user can call low-level C/Fortan and do 'real' numerical computation, without having to spend 100's of hours getting everything set up.

Professional Certification Programs

Universities aren't teaching scientific computing yet, but there is a big need for it in industry. A lot of corporations have created their own courses to get people up to speed.

A link for IBM's main python course is shown below. It teaches how to do everything Matlab can do (and more) with Python. Also, the course is taught with 'virtual-servers', which means you *immediately start coding in python without having to install anything*.

https://www.coursera.org/specializations/ibm-data-science-professional-certificate

The best introduction to python I have seen is from Course 4: of the IBM program.

Course 4: Python for Data Science and AI

- How to do functions, variables, and objects with python.
- How to use a 'jupyter notebook'
- How to do high-speed computation with NumPy.

Talk Python Podcast

Interviews with lead software engineers and project managers from all major companies. Listen to the highest ranking software engineers from Microsoft, Google, etc, talk about what they are doing, how they are doing it, and what tools they are using.

This is a big deal because, ten years ago, it would have been absurd to think that industry leaders would openly discuss the exact builds of their software. However, almost all major software companies (except mathworks) have been forced to open-source large portions of their products in order to stay competitive.

https://talkpython.fm/

Links

Anaconda: Install the python stack all at once!

https://www.anaconda.com/distribution/

Web Assembly: Assembly Language for the web

https://webassembly.studio/

Sushi.js: A lightweight linear algebra for Javascript

http://mil-tokyo.github.io/miljs.html

Three.js: Short-cut functions for WebGL (3d graphics online)

https://threejs.org/

Sublime Text: The 'old-school' text editor

https://www.sublimetext.com/

Iodide: Scientific Computation with Javascript

https://alpha.iodide.io/

History of the web: Presentation by Steve Klabik

https://www.youtube.com/watch?v=EHzrk1j7z7Q

Make a free website: Free direct web-hosting service: no ads.

https://www.000webhost.com/

Matlab File Exchange: Get sample codes for math stuff

https://www.mathworks.com/matlabcentral/fileexchange/

IBM data science professional certificate: Good way to learn python

https://www.coursera.org/specializations/ibm-data-science-professional-certificate