TECHNOLOGIES AND LANGUAGES

WHAT IS EVERYTHING?

Terminal is in Linux/Mac, it is a Command Line Interface. It's kind of like Windows command prompt but more powerful. It gives you powerful control over the computer to access files, run all sorts of commands and more. You can write scripts in here, edit any of your computers files, or view processes of the computer. The terminal is a program, but it's more designed to interface with the computer itself.

Text Editor - Notepad++/Sublime are popular ones. These are often used for making code changes, or programming on those web/interpreted languages. They often have alot of features, but not as much as an IDE. Not usually language specific, just edit a file and close it out. Maybe you just need to edit 1 file in your project and don't want to open up the IDE. Maybe you are just editing some code that you will FTP (file transfer on the web) back up to a website quickly. There are also Text Editors you can run inside of your Terminal like VIM/Emacs that allow you to do more complex coding tasks on the command line.

Compiler- These compile your code from whatever language you're programming in. Once it is compiled, you can't edit it. These are most programs you see on your computer. This will be for stuff like C/C++. You can write a C++ program in a mac/linux terminal, and compile it in the terminal then run it. But there are also big programs that help coding and compiling, these are called IDE's. Only some languages require compiling, others languages (like web languages) are interpreted.

IDE (Integrated Development Environment)- This is like a program interface for developing. It's got all sorts of functions for your project. Netbeans/Eclipse are often used for Java. Visual Studio is a popular one for C++. (The IDE is also a compiler.) IOS is programmed in Swift/ObjectiveC in the Xcode IDE, Android is programmed in Java and uses Android Studio IDE.

Functional Vs Object Oriented -
Functional - A specific use case of a program, do this thing or things. The program can be large, but often does not change "states". The code is often much shorter.
Object Oriented - Maybe your program is massive with many different changing parts. Or you are using many prebuilt functions. Java is designed as an OOP

language. Cell phone languages are often very OOP. Instead of programming the camera from scratch, you would just call the camera function within your program.

Compiled vs Interpreted -
Compiled - C, C++, C#, Obj C. They run faster but must be compiled first. The program is standalone from its source code.
Interpreted - Java, Ruby, Python, Lots of web stuff, often slower. But you are actually running the source code live. The line occasionally get blurred here because some languages can be interpreted then cached.

LANGUAGES

Fundamental Languages, Machines and Hardware 1950-1980

Binary - 1's and 0's. What the metal see's. Whatever you do on a computer, eventually becomes this.

Assembly - Some basic logical actions, that start to be human readable'ish. This Low-Level language is just a step away from 1's and 0's, so it compiles to binary. But you can still feasibly write a program in it. Roller Coaster Tycoon was famously written all in assembly.

Cobol – Lots of legacy Banking/Retail/Commerce still runs COBOL.

SQL –SQL still drives most relational databases. (MySQL, Maria, PostgreSQL)

High Level- Builds Operating Systems and Software 1970-1990

C - This compiles into Assembly. It's light, basic, and powerful So many embedded systems still run C. Also almost everything is built on top of C. Windows, Linux, Mac, MySQL and many more.

C++ - "Classes" are the major addition to C that allowed for a new way of programming. Object Oriented. This is a more robust programming style.

Objective C - It also added Object Oriented features, but was mainly adopted by Apple. So Apples OS is built on FreeBSD Unix fork, and there programs are written in Objective C, or for IOS there own Swift language.

PERL – A powerful language built on C, still very common but rarely seen. It runs on the backend of many servers. It's like server duct tape.

Web Servers

Web Servers actually serve the files on the computer to then screen when you connect to that computer. When you visit a site, you're just accessing another computer's files.

Apache – The most popular web server on the internet. Flexible and powerful. Not fun to configure your first time.

NGINX – Newer, faster and lighter than Apache. But requires more optimization. Sometimes used on top of apache for cacheing and other high traffic sites.

IIS – Microsoft's web server. You use it when you are forced to run .aspx webpages or you just windows servers…. Why??

NodeJS – A web app server in javascript, customized for real time apps. Built on googles V8 JS engine it utilizes sockets to create real time applications with

Web and New Languages

Java – Java is a language and runs in a virtual container or JVM. Really designed from the core around Object Oriented principles. We almost all have the Java Updater installed and apps run on it. (Java applets were recently dropped from the web because of exploitability). Lots of corporate software stuff runs Java. This is Not Javascript!

HTML/CSS – HTML handles the basic structure website. Code runs inside of the HTML. HTML5 allows for lot of advanced features now but HTML has always been fairly basic. CSS specifies the styling, colors, fonts, positioning and more. CSS is often a separate script called inside the head.

Javascript - Not related to Java! Lots of internet runs javascript. It added things like movement of things on the web. Languages like Jquery, were built out of javascript, they make things move pretty, and do things. The syntax is so common in web work, that the language was used to build other things. Frameworks like Ember, backends can run Node.js, or full stack like meteor.js

AJAX -(Asyncronous Javascript and XML) allowed for the web to do real time updates without refreshing the page. This was a HUGE deal.

C# - Don't confuse it with Objective C! It' a Microsoft language for programming .net

PHP – Arguably the most popular server side language, many servers run PHP in the form of the LAMP (Linux, Apache, MySQL, PHP) handles a lot of backend web functionality and can interface between the website and database.

Ruby – Ruby is the language, and it usually runs on a framework called "Rails". Hence, Ruby On Rails. Ruby supports "gems" that are packages with lots of prebuilt functionality for easily making web applications. It's syntax is consider elegant, it's still a favorite language for prototyping apps.

Python – One of the most human readable languages made it very popular. Now it has support for many scientific and mathematical functions and AI.

Scala – Built to address Java failures, it runs on the JVM. Noted for being both Functional and Object Oriented.

Swift – Built as an improvement for ObjectiveC by Apple to run in Xcode.