Java script

Day-2

Agenda

- what is language
- Different type of language

language

- A language is a structured system of communication used by humans, based on speech and gesture (spoken language), sign, or often writing. The structure of language is its grammar and the free components are its vocabulary. Many languages, including the most widely-spoken ones, have writing systems that enable sounds or signs to be recorded for later reactivation. Human language is unique among known systems of animal communication in that it is not dependent on a single mode of transmission (sight, sound etc.), it is highly variable between cultures and across time, and affords a much wider range of expression than other systems.[1] It has the properties of productivity and displacement, and relies on social convention and learning.

Computer language

- The computer language is defined as code or syntax which is used to write programs or any specific applications. The computer language is used to communicate with computers

Different Types of Computer Language

- Machine Language
- Assembly Language
- High-Level Language

Machine level language

- The machine language is sometimes referred to as machine code or object code which is set of binary digits 0 and 1. These binary digits are understood and read by a computer system and interpret it easily. It is considered a native language as it can be directly understood by a central processing unit (CPU). The machine language is not so easy to understand, as the language uses the binary system in which the commands are written in 1 and 0 form which is not easy to interpret. There is only one language which is understood by computer language which is machine language. The operating system of the computer system is used to identify the exact machine language used for that particular system.
- The operating system defines how the program should write so that it can be converted to machine language and the system takes appropriate action. The computer programs and scripts can also be written in other programming languages like C, C++, and JAVA. However, these languages cannot be directly understood by a computer system so there is a need for a program that can convert these computer programs to machine language. The compiler is used to convert the programs to machine language which can be easily understood by computer systems. The compiler generates the binary file and executable file.
- Example of machine language for the text “Hello World”.
- 01001000 0110101 01101100 01101100 01101111 00100000 01010111 01101111 01110010 01101100 01100100.

Assembly Language

- The assembly language is considered a low-level language for microprocessors and many other programmable devices. The assembly language is also considered as second-generation language. The first generation language is machine language. The assembly language is mostly famous for writing an operating system and also in writing different desktop applications. The operations carried out by programmers using assembly language are memory management, registry access, and clock cycle operations. The drawback of assembly language is the code cannot be reused and the language is not so easy to understand. The assembly language is considered a group of other languages. It is used to implements the symbolic representation of machine code which is used to program CPU architecture. The other name of assembly language is assembly code. For any processor, the most used programming language is assembly language.
- In assembly language, the programmer does the operation which can be directly executed on a central processing unit (CPU). The language has certain drawbacks as it does not contain any variables or functions in programs and also the program is not portable on different processors. The assembly language uses the same structure and commands which machine language does use but it uses names in place of numbers. The operations performed using the assembly language is very fast. The operations are much faster when it is compared to high-level language.

High-Level Language

- The development of high-level language was done when the programmers face the issue in writing programs as the older language has portability issues which mean the code written in one machine cannot be transferred to other machines. Thus lead to the development of high-level language. The high-level language is easy to understand and the code can be written easily as the programs written are user-friendly in a high-level language. The other advantage of code written in a high-level language is the code is independent of a computer system which means the code can be transferred to other machines. The high-level of language uses the concept of abstraction and also focus on programming language rather than focusing on computer hardware components like register utilization or memory utilization.
- The development of higher-level language is done for a programmer to write a human-readable program that can be easily understood by any user. The syntax used and the programming style can be easily understood by humans if it is compared to low-level language. The only requirement in a high-level language is the need of compiler. As the program written in a high-level language is not directly understood by the computer system. Before the execution of high-level programs, it needs to be converted to machine level language. The examples of high-level language are C++, C, JAVA, FORTRAN, Pascal, Perl, Ruby, and Visual Basic.

Some popular language

- C
- Java
- Python
- C++
- C#
- Visual Basic
- JavaScript
- PHP
- SQL
- Assembly language
- R
- Groovy

C

- **What is C?**
- The C programming language was first released back in 1972. It's a high-level procedural language that has become one of the most widely-used ones out there. Despite its age, it's still a relatively complex language, yet its influence can be seen in many others. C#, C++, Java, Python, and more all take elements of inspiration from C.
- **What is C used for?**
- C is a versatile language that has uses in many different areas. It's mainly used for creating system applications, meaning that operating systems such as Windows and Linux use a lot of C programming. You can also use C to create games, graphics, and apps that use lots of calculations.
- **How to learn C**
- If you're a newcomer to the world of coding and programming, C is a fairly difficult place to start. There are other high-level languages such as Python that offer a better starting point. That being said, there are plenty of online resources for learning C.

- C is the predecessor to more complex programming languages like Java and C#. C is best when you want to work small and when dealing with low-level applications. It's widely used for embedded systems like the firmware of your television or the operating system of an airplane, as well as computer operating systems like Windows.

Python

- As far as easy-to-learn programming languages go, Python is up there with the best of them. If you're just starting out learning to code, there are few better places to start.
- **What is Python?**
- Python is an object-oriented, high-level programming language launched in 1992. It's built in such a way that it's relatively intuitive to write and understand, making it ideal for those who want rapid development. It's a very popular language right now, meaning there are plenty of Python jobs available.
- **What is Python used for?**
- Because of how versatile a language it is, Python has many applications. As well as being good for general use, such as for web applications, it also has many areas of specialisation. A good example of the latter is for artificial intelligence (AI) and machine learning.
- **How to learn Python**
- We have a range of free Python courses that can get you started with some of the basics. Programming for Everybody is the ideal place if you're totally new to programming. For more experienced learners, we have ExpertTracks on Python topics such as data analytics, deep learning, and data visualisation.

Java

- **Java**
- Another useful starting point for programming is to learn Java. It's a versatile and general-purpose language that is similar to JavaScript and Python.
- **What is Java?**
- Java is a hugely popular object-oriented programming language. Part of its popularity comes from the fact that once you write a piece of code in Java, it can run on just about any device with the Java platform.
- **What is Java used for?**
- The 'write once, run anywhere' concept at the heart of Java means it has many different uses. However, some of the main uses include for business software, web applications, and mobile apps. Google's Android OS, for example, uses Java as its native language.
- **How to learn Java**
- You can get started with the basics of Java programming with our free courses on building your first mobile game. This will introduce you to some of the essential concepts and constructs of Java. If you're looking for a more detailed exploration, our Software Development with Python and Java ExpertTrack is the ideal place to start.

PHP

- PHP is another easy-to-learn coding language that is both free and open source. Much like JavaScript, it's mainly used for coding on websites.
- **What is PHP?**
- PHP: Hypertext Preprocessor is a high-level, object-oriented programming language. Although similar to JavaScript in some ways, PHP is a server-side rather than client-side scripting language that is embedded in HTML. As such, it is often used together with JavaScript. As one analogy puts it, if PHP is the paintbrush, JavaScript is the paint.
- PHP is server sided scripting language used for building any website.
- **What is PHP used for?**
- There are many uses for PHP, although mostly for website development. You can use it to manage dynamic content and databases on a website, for example. The latter is particularly relevant, as it integrates well with database languages such as MySQL.
- **How to learn PHP**
- As with many coding languages, the best way to learn is to get as much practice as you can with the language. There are many resources available that will take you through the basics, such as how the language works and what the basic syntax looks like.

SQL

- **SQL**
- If you're interested in areas like database management, SQL is a language that you simply have to know. In fact, most developers need to have at least a basic knowledge of the language.
- **What is SQL?**
- Structured Query Language (SQL, sometimes called Sequel), is a domain-specific language designed for managing data held in databases. Unlike many of the different programming languages mentioned so far, this isn't a general-purpose one, meaning its use is far narrower.
- **What is SQL used for?**
- SQL is mainly used to communicate with databases. As such, it's used by server developers, database administrators, and software developers across a range of industries. However, more recently, it's also used in areas such as data analysis and big data mining.
- **How to learn SQL**
- If you're new to the subject, our Introduction to Databases and SQL is the ideal place to start. Here, you'll learn about how databases work and how SQL can be used to search and manipulate data.

Java Script

- Despite its name, JavaScript isn't directly related to Java. However, it does use a Java-like syntax, which is where the name comes from. If you're interested in client-side web browser coding, this is a language you want to learn.
- **What is JavaScript?**
- JavaScript is a high-level, object-oriented programming language (although that second point is up for some debate). The language was created in the early days of the internet, back in 1995. You'll find that all major web browsers have JavaScript support.
- **What is JavaScript used for?**
- As mentioned, JavaScript is a language used to write code that runs in web browsers. As such, it has a wide range of applications and is especially useful for making web pages interactive and responsive. It's often used alongside HTML and CSS to add things like animations, video players, and even browser-based games.
- **How to learn JavaScript ???**

What is JavaScript?

- *JavaScript* was initially created to “make web pages alive”.
- The programs in this language are called *scripts*. They can be written right in a web page’s HTML and run automatically as the page loads.
- Scripts are provided and executed as plain text. They don’t need special preparation or compilation to run.
- In this aspect, JavaScript is very different from another language called Java

Why is it called JavaScript?

- When JavaScript was created, it initially had another name: “LiveScript”. But Java was very popular at that time, so it was decided that positioning a new language as a “younger brother” of Java would help.
- But as it evolved, JavaScript became a fully independent language with its own specification called ECMAScript, and now it has no relation to Java at all.
- Today, JavaScript can execute not only in the browser, but also on the server, or actually on any device that has a special program called the JavaScript engine.
- The browser has an embedded engine sometimes called a “JavaScript virtual machine”.
- Different engines have different “codenames”. For example:

- V8 – in Chrome, Opera and Edge.
- SpiderMonkey – in Firefox.
- ...There are other codenames like “Chakra” for IE, “JavaScriptCore”, “Nitro” and “SquirrelFish” for Safari, etc.

The terms above are good to remember because they are used in developer articles on the internet. We’ll use them too. For instance, if “a feature X is supported by V8”, then it probably works in Chrome, Opera and Edge.

How do engines work?

- The engine (embedded if it's a browser) reads ("parses") the script.
- Then it converts ("compiles") the script to the machine language.
- And then the machine code runs, pretty fast.
- The engine applies optimizations at each step of the process. It even watches the compiled script as it runs, analyzes the data that flows through it, and further optimizes the machine code based on that knowledge.

What can in-browser JavaScript do?

- Modern JavaScript is a “safe” programming language. It does not provide low-level access to memory or CPU, because it was initially created for browsers which do not require it.
- JavaScript’s capabilities greatly depend on the environment it’s running in. For instance, Node.js supports functions that allow JavaScript to read/write arbitrary files, perform network requests, etc.
- In-browser JavaScript can do everything related to webpage manipulation, interaction with the user, and the webserver.
- For instance, in-browser JavaScript is able to:
- Add new HTML to the page, change the existing content, modify styles.
- React to user actions, run on mouse clicks, pointer movements, key presses.
- Send requests over the network to remote servers, download and upload files (so-called AJAX and COMET technologies).
- Get and set cookies, ask questions to the visitor, show messages.
- Remember the data on the client-side (“local storage”).

What CAN'T in-browser JavaScript do?

- JavaScript's abilities in the browser are limited for the sake of the user's safety. The aim is to prevent an evil webpage from accessing private information or harming the user's data.
- Examples of such restrictions include:
- JavaScript on a webpage may not read/write arbitrary files on the hard disk, copy them or execute programs. It has no direct access to OS functions.
- Modern browsers allow it to work with files, but the access is limited and only provided if the user does certain actions, like "dropping" a file into a browser window or selecting it via an <input> tag.
- There are ways to interact with camera/microphone and other devices, but they require a user's explicit permission. So a JavaScript-enabled page may not sneakily enable a web-camera, observe the surroundings and send the information to the NSA.
- Different tabs/windows generally do not know about each other. Sometimes they do, for example when one window uses JavaScript to open the other one. But even in this case, JavaScript from one page may not access the other if they come from different sites (from a different domain, protocol or port).
- This is called the "Same Origin Policy". To work around that, *both pages* must agree for data exchange and contain a special JavaScript code that handles it. We'll cover that in the tutorial.
- This limitation is, again, for the user's safety. A page from http://anysite.com which a user has opened must not be able to access another browser tab with the URL http://gmail.com and steal information from there.
- JavaScript can easily communicate over the net to the server where the current page came from. But its ability to receive data from other sites/domains is crippled. Though possible, it requires explicit agreement (expressed in HTTP headers) from the remote side. Once again, that's a safety limitation.
- Such limits do not exist if JavaScript is used outside of the browser, for example on a server. Modern browsers also allow plugin/extensions which may ask for extended permissions.

What makes JavaScript unique?

- There are at least *three* great things about JavaScript:
- Full integration with HTML/CSS.
- Simple things are done simply.
- Support by all major browsers and enabled by default.
- JavaScript is the only browser technology that combines these three things.
- That’s what makes JavaScript unique. That’s why it’s the most widespread tool for creating browser interfaces.
- That said, JavaScript also allows to create servers, mobile applications, etc.

Languages “over” JavaScript

- The syntax of JavaScript does not suit everyone’s needs. Different people want different features.
- That’s to be expected, because projects and requirements are different for everyone.
- So recently a plethora of new languages appeared, which are *transpiled* (converted) to JavaScript before they run in the browser.
- Modern tools make the transpilation very fast and transparent, actually allowing developers to code in another language and auto-converting it “under the hood”.
- Examples of such languages:
- CoffeeScript is a “syntactic sugar” for JavaScript. It introduces shorter syntax, allowing us to write clearer and more precise code. Usually, Ruby devs like it.
- TypeScript is concentrated on adding “strict data typing” to simplify the development and support of complex systems. It is developed by Microsoft.
- Flow also adds data typing, but in a different way. Developed by Facebook.
- Dart is a standalone language that has its own engine that runs in non-browser environments (like mobile apps), but also can be transpiled to JavaScript. Developed by Google.
- Brython is a Python transpiler to JavaScript that enables the writing of applications in pure Python without JavaScript.
- Kotlin is a modern, concise and safe programming language that can target the browser or Node.
- There are more. Of course, even if we use one of transpiled languages, we should also know JavaScript to really understand what we’re doing.

Summary

- JavaScript was initially created as a browser-only language, but it is now used in many other environments as well.
- Today, JavaScript has a unique position as the most widely-adopted browser language with full integration in HTML/CSS.
- There are many languages that get “transpiled” to JavaScript and provide certain features. It is recommended to take a look at them, at least briefly, after mastering JavaScript.

Javascript

- JavaScript is the **Programming Language** for the Web.
- JavaScript can update and change both **HTML** and **CSS.**
- JavaScript can **calculate**, **manipulate** and **validate** data.

Assignment

- https://www.indeed.com/career-advice/career-development/types-of-programming-languages
- https://www.watelectronics.com/types-of-programming-languages-with-differences/
- Difference between c and java
- Difference between java and java script
- https://www.peerbits.com/blog/reasons-to-choose-reactjs-for-your-web-development-project.html

Thank You