Goal of the Submodule

The goal of this submodule is to help the learners understand the following:

- programming and programming languages.
- Algorithm
- different types of developers
- JSON
- XML- markup language

Topics

- Programming languages types
- Algorithm and programming
- Developers profiles
- JSON
- XML- markup language

Glossary

Term	Definition
POP	Procedural Oriented programming
OOP	Object Oriented Programming
UI	User Interface
UX	User Experience
JSON	JavaScript Object Notation

What is Programming?

Programming: is a way to communicate with computers.

Language: is made up of letters to words that describes something or share opinions or ideas with each other.

Programming Language : is a computer language that used by programmers aka developers to communicate with computers.
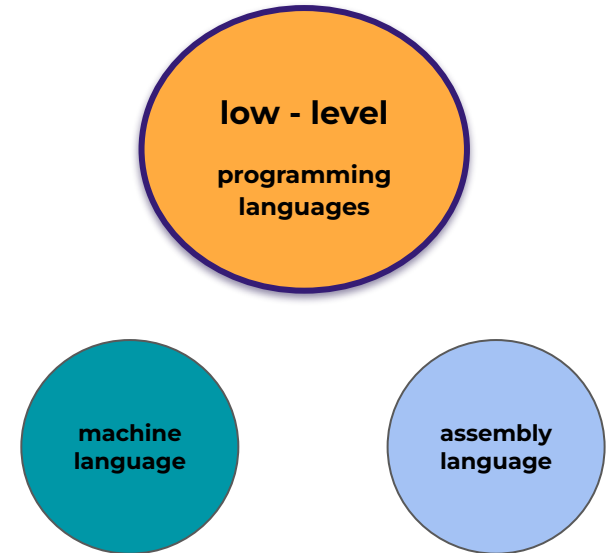
Types of Programming Languages

1- Low-level programming language

Is machine dependent (0s and 1s) programming language. Where the processor runs low-level directly with no need for a compiler or interpreter, so the program runs very fast.
Low-level is further division into:

- A- Machine language: Machine language is easier to read because it is normally displayed in binary or hexadecimal form (base 16) form. It does not require a translator to convert the programs because computers directly understand the machine language programs.

low - level

programming languages

machine language

assembly language

Types of Programming Languages

B- Assembly language

It represents the set of instructions in a symbolic and human-understandable form. It uses an assembler to convert the assembly language to machine language.

The advantage of assembly language is that it requires less memory and less execution time to execute a program.

Types of Programming Languages

Example assembly

0804892b jmp

0804892d nop

0804804c leave

0804804c ret

2- Middle-level programming languages

lie between the low-level programming languages and high-level programming languages. They are also known as intermediate programming languages and pseudo-languages.

A middle-level programming language has the advantage that it supports the features of high-level programming, it is a user-friendly language, and it is closely related to machine language and human language.

Types of Programming Languages

3- High-level programming languages are designed for developing user-friendly software programs and websites. These programming languages require a compiler or interpreter to translate the program into machine language (execute the program). The main advantage of high-level languages is that they are easy to read, write, and maintain.

High-level programming languages include Python, Java, JavaScript, PHP, C#, C++, Objective C, Cobol, Perl, Pascal, LISP, FORTRAN, and the Swift programming language.

high-level

programming languages

procedural oriented

object oriented

natural
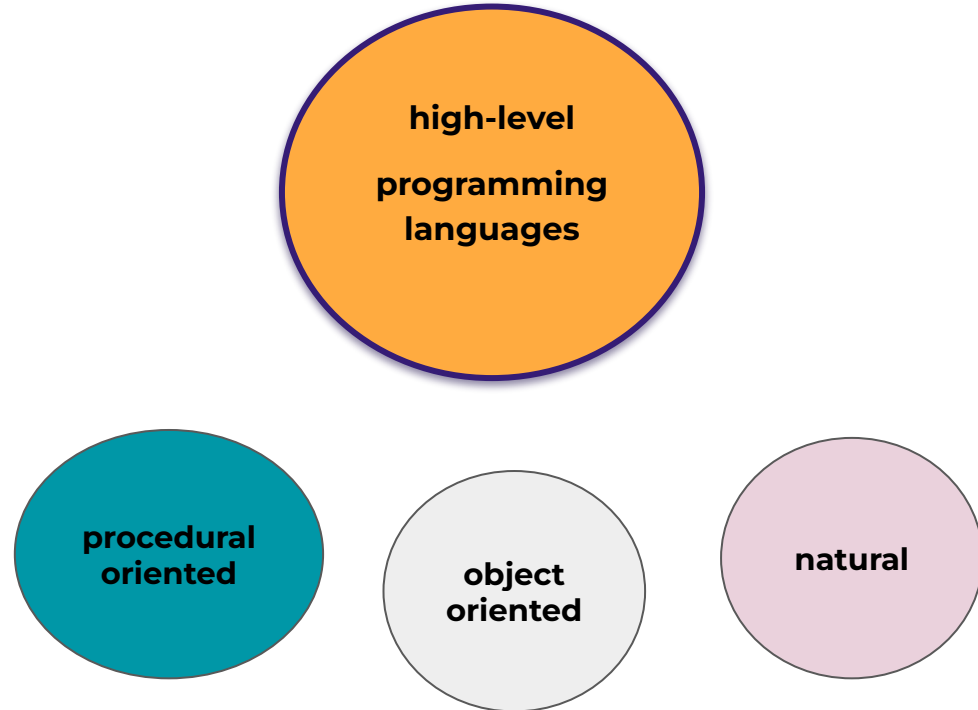
Types of Programming Languages

A high-level language is further divided into three parts:

- **A- Procedural Oriented programming language**
  POP is derived from structured programming and based upon the procedure call concept. It divides a program into small procedures called routines or functions.
 Example: C, Basic, Pascal, Fortran

ProTip: You need a programming editor like IDE, this app will help you write better and cleaner code.

Types of Programming Languages

- **B- Object-Oriented Programming language:** OOP is based upon the objects. In this programming language, programs are divided into small parts called objects.
Example: C++, Java, Python, C#

- **C- Natural language processing** which is a part of Computer Science, Human language, and Artificial Intelligence. It is the technology that is used by machines to understand, analyse, manipulate, and interpret human's languages.

Digital Career Institute
DCI
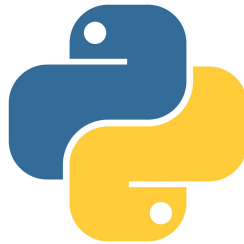
At the core of the lesson

Lessons Learned:

- An assembly language is a type of low-level programming language that is intended to communicate directly with a computer's hardware

Assembly Language

```
mov ecx, ebx
mov esp, edx
mov edx, r9d
mov rax, rdx
```

Assembler + Linker →

Machine Language

```
100101011001
010011111011
111010101101
01010101010
```

Programmer

Processor

At the core of the lesson

Lessons Learned:

- A programming language is a formal language comprising a set of strings that produce various kinds of machine code output. Programming languages are one kind of computer language

Src dealna.com

Self Study

Research on which programming languages are out there and their types

Algorithm

Algorithm

In computer programming terms, an algorithm is a **set of well-defined instructions to solve a particular problem**. It takes a set of **input** and produces a desired **output**.

It can be defined as simply **a set of steps used to complete a specific task**. They're the building blocks for programming, and they allow things like computers, smartphones, and websites to function and make decisions.

Algorithms are a great way of **automating computer decisions**.

Algorithm

In addition to being used by technology, a lot of things we do on a daily basis are similar to algorithms.

For example, algorithms resemble recipes. Recipes tell you how to **accomplish a task by performing a number of steps**.

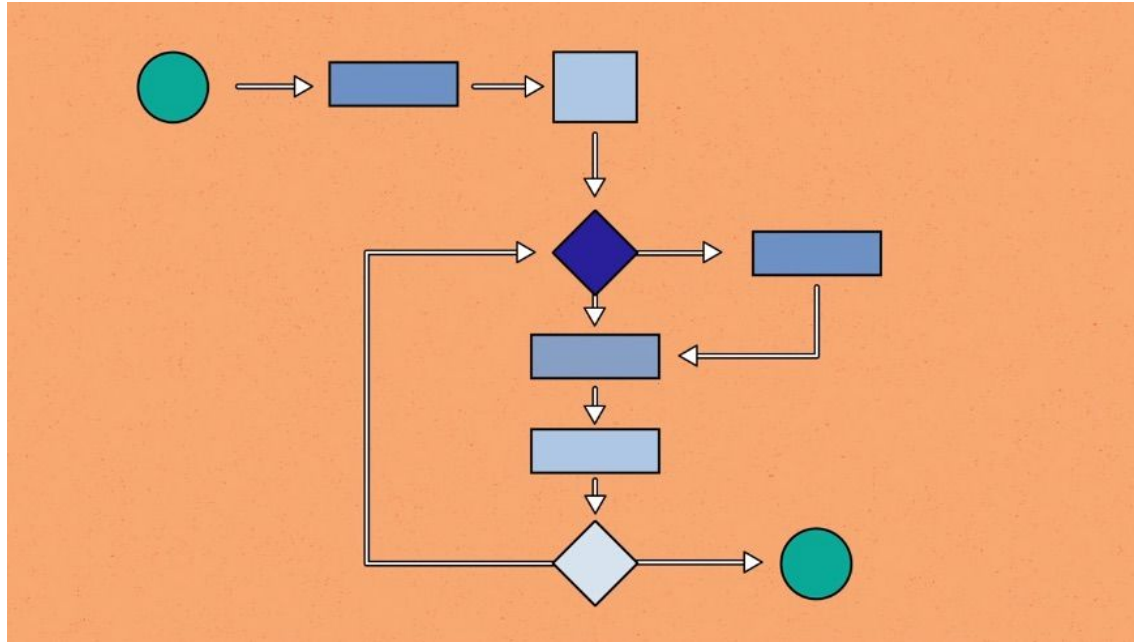| Bake a Cake	Make Spaghetti
Preheat the oven	Boil water in a pot
Mix the ingredients	Add the spaghetti
Pour into a baking pan	Wait for it to cook
Put it in the oven	Drain the water
…	…

Programs work in a similar way. Their code is made up of **algorithms telling them what to do**.

However, *algorithm* is a technical term with a more specific meaning than recipe, and calling something an algorithm means that the following properties are all true:

- An algorithm is an **unambiguous description** that makes clear **what has to be implemented.**
  - In a recipe, a step such as "Bake until done" is **ambiguous** because it **doesn't explain what "done" means**.
  - In a computational algorithm, a step such as "Choose a large number" is **vague**: **what is large?** 1 million, 1 billion, or 100?
- An algorithm **expects a defined set of inputs.**
  - For example, it might require two numbers where both numbers are greater than zero. Or it might require a word, or a list of zero or more numbers.
- An algorithm **produces a defined set of outputs.**
  - It might output the larger of the two numbers, an all-uppercase version of a word, or a sorted version of the list of numbers.

Algorithm

- An algorithm is **guaranteed to terminate and produce a result**, always stopping after a finite time.
  - If an algorithm could potentially run forever, it wouldn't be very useful because you might never get an answer.
- Most algorithms are **guaranteed to produce the correct result.**
  - It's rarely useful if an algorithm returns the largest number 99% of the time, but 1% of the time the algorithm fails and returns the smallest number instead.
- If an algorithm imposes a **requirement on its inputs** (called a *precondition*), that **requirement must be met**.
  - For example, a pre condition might be that an algorithm will only accept positive numbers as an input. If the pre conditions aren't met, then the algorithm is allowed to fail by producing the wrong answer or never terminating.

Algorithm

An Algorithm flow

Algorithm

Algorithm Flow Chart

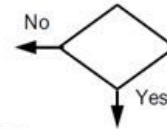### Symbols Used in Flowcharts

|	
Start / End	(oval)
Process Step	(rectangle)
Decision	(diamond) No / Yes
Connector	(circle)
Measurement	(triangle with M)
INPUT/OUTPUT	(parallelogram)
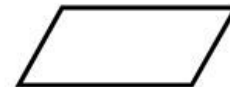
Algorithm

Print the sum of 5 numbers

Algorithm in simple English

1. **Initialize** *sum = 0* and *count = 0* **(PROCESS)**
2. **Enter** *n* **(I/O)**
3. **Find** *sum + n* and **assign** it to *sum* and **increment** *count by 1* **(PROCESS)**
4. Is *count < 5*? **(DECISION)**
   a. If YES go to step 2
   b. Else **print** *sum* **(I/O)**
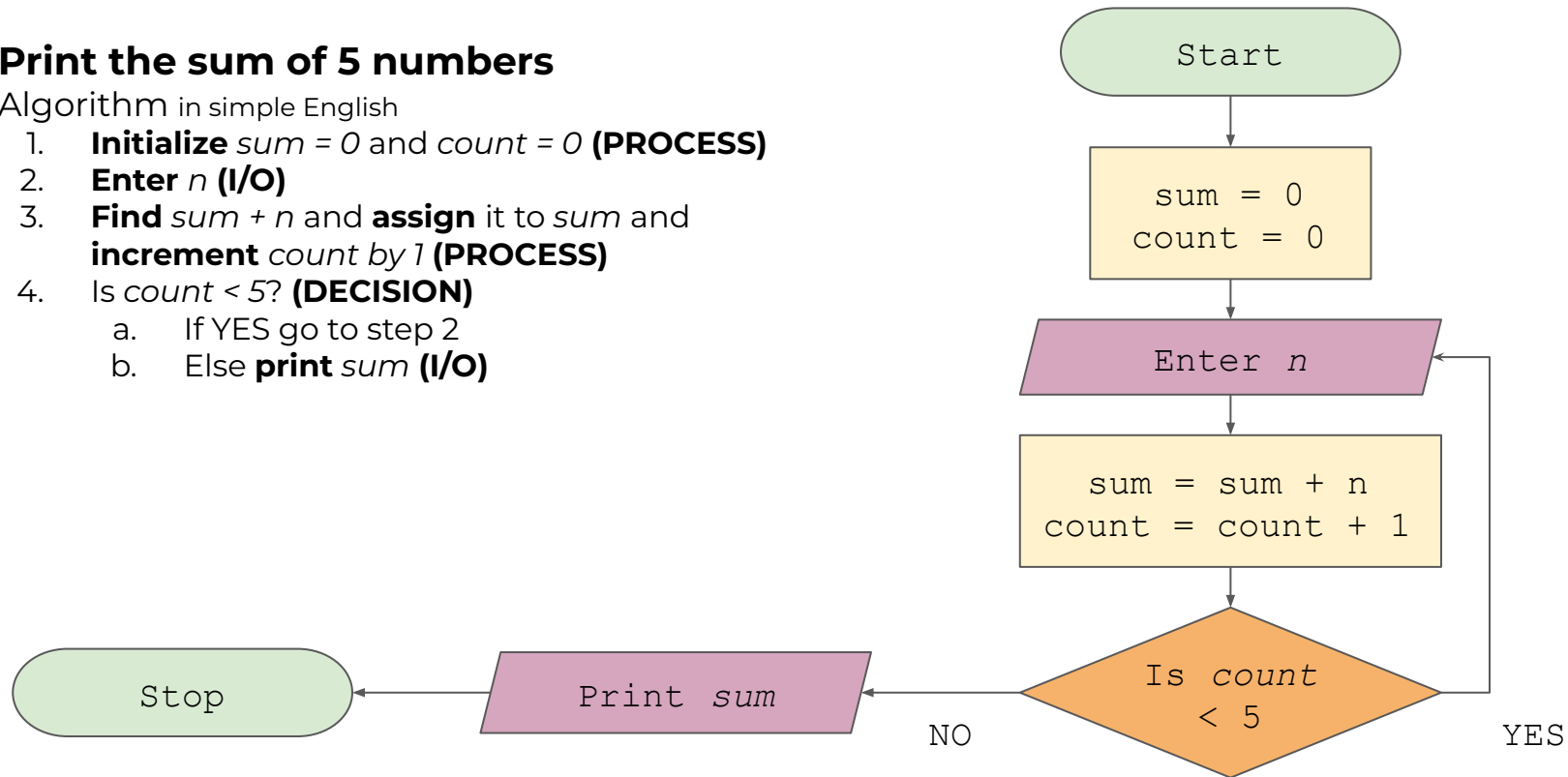
Algorithm

Qualities of Good Algorithms

- **Input and output** should be **defined precisely**.

- Each step in the algorithm should be **clear and unambiguous**.

- Algorithms should be most effective among many different ways to **solve a problem**.

At the core of the lesson

Lessons Learned:

- An algorithm is a set of instructions for solving a problem or accomplishing a task.
- Every computerized device uses algorithms to perform its functions.
- Computer algorithms make life easier by trimming the time it takes to manually do things.
- Algorithms allow workers to be more proficient and focused, making slow processes more proficient.

Developers profiles
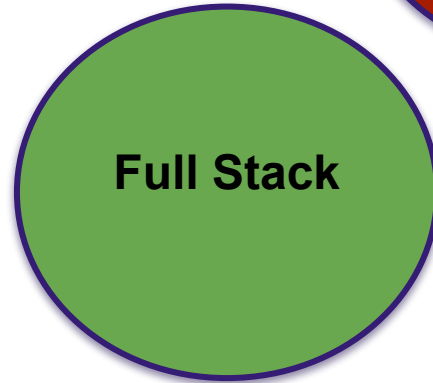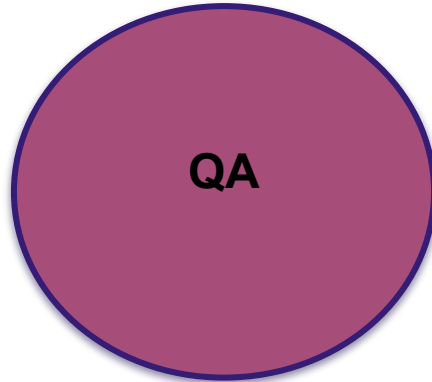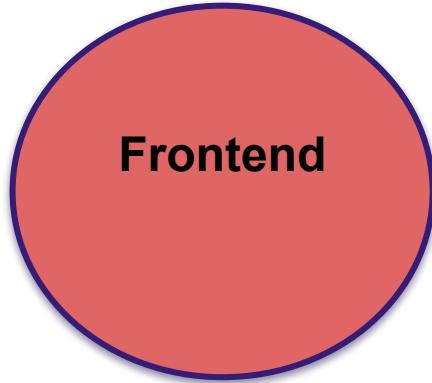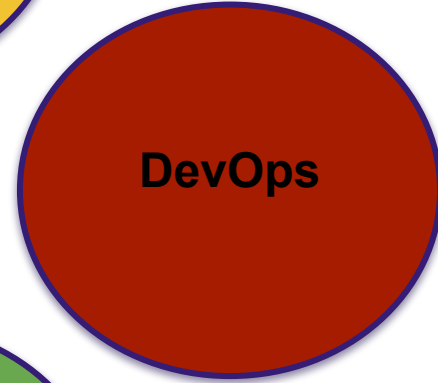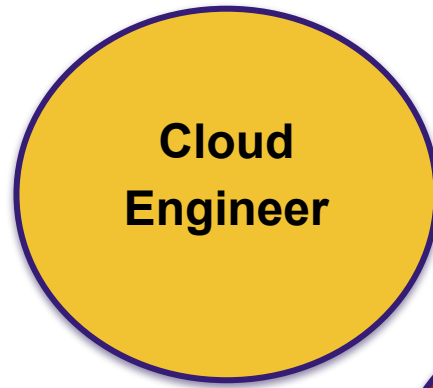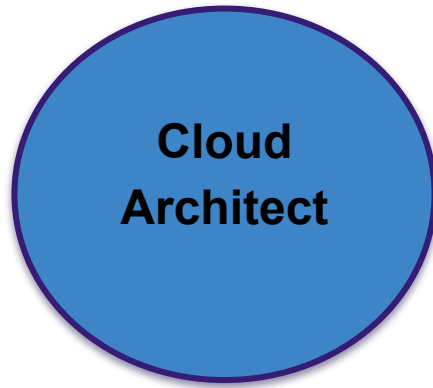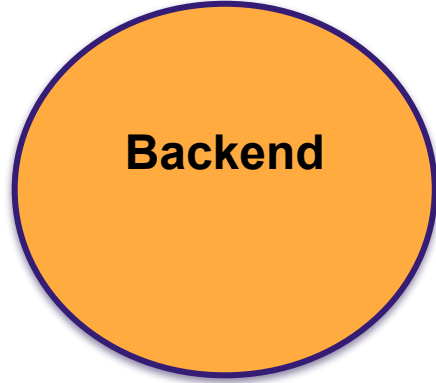
Front-end

Front-end web development, also known as client-side development is the practice of producing HTML, CSS and JavaScript for a website or Web Application so that a user can see and interact with them directly.

Front-end

Two very important parts for any front-end app
- User experience (UX) is the interaction and experience users have with a company's products and services.
- User interface (UI) is the specific asset users interact with. For example, UI can deal with traditional concepts like visual design elements such as colors and typography.

Back-end

A back-end developer builds and maintains the technology that powers (server, applications and database) components which, together, enable the user-facing side of the website to even exist in the first place.

Back-end

In order to make the server, application, and database communicate with each other, back-end devs use server-side languages like PHP, Ruby, Python, Java, and .Net to build an application, and tools like MySQL, Oracle, and SQL Server to find, save, or change data and serve it back to the user in front-end code.

Full-stack

Is a web developer or engineer who works with both the front and back ends of a website or application—meaning they can tackle projects that involve databases, building user-facing websites, or even work with clients during the planning phase of projects.

Src geeksforgeeks.org

Full-stack

A full-stack developer should be

- familiar with HTML, CSS, JavaScript, and one or more back end languages,
- most full stack developers specialize in a particular back end programming language, like Ruby, PHP, Java or Python,
- some full stack developers also learn either project management, visual design, web design, or user experience skills to complete their "stack".

DevOps

A DevOps (Development and Operations) engineer introduces processes, tools, and methodologies to balance needs throughout the software development life cycle, from coding and deployment, to maintenance and updates.

Src software.af.mil
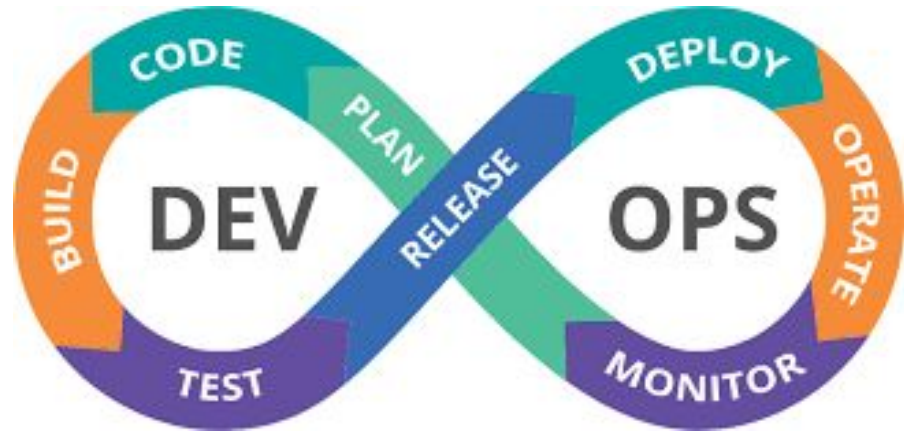
DevOps

DevOps is all about the unification and automation of processes, and DevOps engineers are instrumental in combining code, application maintenance, and application management. All of these tasks rely on understanding not only development life cycles, but DevOps culture, and its philosophy, practices, and tools.

QA

Quality assurance engineers

A QA engineer focuses on improving software development processes and preventing defects in production. In other words, they make sure the software development team is doing the right things the right way.

Src perfectial.com

QA

The QA engineer job scope embraces a number of duties.

- Checking if the product complies with the requirements
- Assessing risks
- Planning ideas to improve product quality
- Planning tests
- Analyzing the test results
- Search and documents bugs

Digital Career Institute

DCI

Cloud Architect

A Cloud Architect is an IT specialist who develops a company's computing strategy
This strategy incorporates cloud adoption plans, cloud application design as well as cloud management and monitoring. Additional responsibilities include support for application architecture and deployment in cloud environments.

The Architect works with various cloud environments such as the public cloud, private cloud, and hybrid cloud.

Digital Career Institute
DCI

Cloud Engineer

Cloud engineers are responsible for assessing the existing infrastructure of a business, and researching solutions for moving different functions (like database storage) to a cloud-based system.
In a short form this person migrates the function to the new system, and maintains it.

Cloud engineers require technical abilities to perform the migration, as well as the ability to negotiate terms with vendors, ensure security of the data, and implement best practices throughout the process.

JSON

JSON

JavaScript Object Notation

is an open standard file format and data interchange format that uses human-readable text to store and transmit data objects consisting of (attribute–value) pairs and arrays (or other serializable values).

It is a common data format with a diverse range of functionality in data interchange including communication of web applications with servers.

JSON is a language-independent data format. It was derived from JavaScript, but many modern programming languages include code to generate and parse JSON-format data. JSON file names use the extension .json

JSON

```
{
  "browsers": {
    "firefox": {
      "name": "Firefox",
      "pref url": "about:config",
      "releases": {
        "1": {
          "release date": "2004-11-09",
          "status": "retired",
          "engine": "Gecko",
          "engine_version": "1.7"
        }
      }
    }
  }
}
```

At the core of the lesson

Lessons Learned:

- JSON is an open standard file format and data interchange format that uses human-readable text to store and transmit data objects

XML- markup language

XML

The Extensible Markup Language (XML) is a simple text-based format for representing structured information: documents, data, configuration, books, transactions, invoices, and much more.

It was derived from an older standard format called SGML (ISO 8879), in order to be more suitable for Web use.

XML

```xml
<?xml version="1.0" encoding="UTF-8"?>
<note>
  <to>ALi</to>
  <from>Olga</from>
  <heading>Hello</heading>
  <body>Let's meet for dinner</body>
</note>
```

At the core of the lesson

Lessons Learned:

- Extensible Markup Language is a markup language that defines a set of rules for encoding documents in a format that is both human-readable and machine-readable

Self Study

Read about and experiment with Xpath

Documentation

- https://www.javatpoint.com/
- https://en.wikipedia.org/wiki/JSON
- https://www.w3.org/standards/xml/core

THANK YOU

Contact Details
DCI Digital Career Institute gGmbH

DCI
Digital Career Institute