****

BINV2120-1

2BIN English as a Foreign Language

BURNY Pauline

Léonard De Vinci University-College – Faculty of Science and Technology

*UNIT 3: Programming languages*

# UNIT 3.1: Programming languages

❤️ Theory: Programming languages

|  |  |  |  |
| --- | --- | --- | --- |
| Language | Paradigm | Key Features | Common Use Cases |
| Python | Multi-paradigm | Readability,  extensive libraries | Web development, data science, automation |
| Java | Object-oriented | Platform independence (JVM), strong typing | Enterprise software,  Android apps |
| JavaScript | Multi-paradigm | Front-end web scripting, asynchronous | Web development,  browser automation |
| C++ | Multi-paradigm | High performance, memory control | Game development, system programming |
| Ruby | Object-oriented | Elegant syntax, dynamic typing | Web development, scripting |
| Swift | Multi-paradigm | Speed, safety, Apple ecosystem | iOS/macOS app development |
| Go (Golang) | Compiled | Concurrency support, simplicity | Cloud services, system programming |
| Rust | Systems | Memory safety, zero-cost abstractions | System programming, game engines |
| PHP | Server-side | Web scripting, easy integration | Web development, server-side scripting |
| SQL | Declarative | Querying relational databases | Database management |

**Some definitions:**

**Object-Oriented:**

* Object-oriented programming (OOP) is a programming paradigm that uses objects, which are instances of classes, to represent and manipulate data and behaviour. It focuses on organizing code into reusable and modular units called classes and encourages the use of inheritance and encapsulation.

**Multi-Paradigm:**

* Multi-paradigm programming languages support more than one programming paradigm, such as procedural, object-oriented, and functional programming. Developers can choose the paradigm that best suits their problem-solving needs.

**Compiled:**

* Compiled programming languages are those that require a separate compilation step before execution. The source code is translated into machine code or an intermediate code by a compiler, and the resulting binary or bytecode is executed directly by the computer's processor or a virtual machine.

**Front-End:**

* Front-end programming involves creating user interfaces and handling interactions on the user's side, typically in web development. Languages like JavaScript are commonly used for front-end scripting to build interactive web applications.

**High Performance:**

* High-performance programming languages are designed to execute code efficiently and quickly. They often offer low-level memory control, which allows developers to optimize code for speed and resource usage.

**Dynamic Typing:**

* Dynamic typing refers to a type system in which variable types are determined at runtime, as opposed to static typing, where types are checked at compile-time. Languages like Python and Ruby use dynamic typing, which offers flexibility but may lead to runtime errors if not used carefully.

**Platform Independence:**

* Platform independence means that code written in a particular language can run on multiple platforms (e.g., different operating systems) without modification. Java achieves platform independence through the Java Virtual Machine (JVM).

**Concurrency Support:**

* Concurrency support in a programming language allows developers to write code that can perform multiple tasks concurrently, often in parallel or asynchronously. It helps improve performance by efficiently utilizing multicore processors.

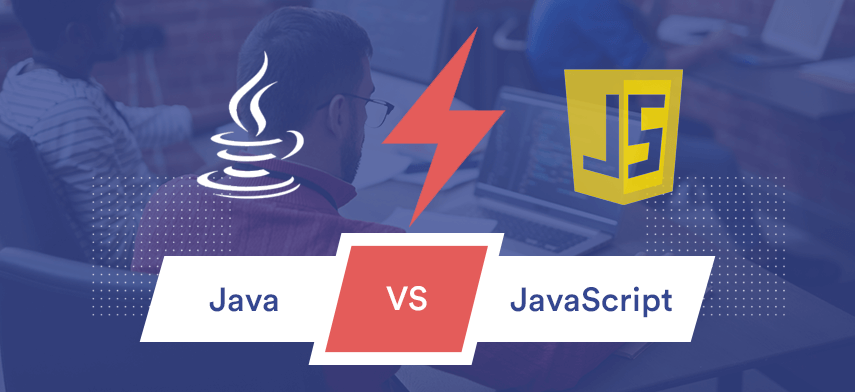
**Memory Safety:**

* Memory safety is a property of some programming languages, like Rust, that helps prevent common memory-related errors such as null pointer dereferences and buffer overflows. It ensures that programs do not access or modify memory locations in unintended ways.

**Declarative**:

* Declarative programming is a style where developers describe what they want to achieve rather than specifying how to achieve it step by step. SQL is a declarative language used for querying databases, where you declare the data you want, and the database system handles the retrieval.

**What is the difference between Java and Javascript?**



Java and JavaScript are two distinct programming languages with different purposes, syntax, and ecosystems. They are often mistaken for each other due to their similar names, but they have very little in common beyond that. Here are the key differences between Java and JavaScript:

Purpose:

* Java: Java is a statically-typed, compiled language primarily used for developing server-side applications, mobile apps (Android), desktop applications, and embedded systems. It's known for its platform independence through the Java Virtual Machine (JVM).
* JavaScript: JavaScript is a dynamically-typed, interpreted language primarily used for web development. It's used to create interactive and dynamic content on websites, including client-side scripting for web browsers.

Static vs. Dynamic Typing:

* Java: Java uses static typing, which means that variable types must be declared explicitly, and type checking is done at compile-time.
* JavaScript: JavaScript uses dynamic typing, allowing variables to change their type at runtime. Type checking is done at runtime.

Compilation:

* Java: Java code is compiled into bytecode, which is then executed by the JVM. This compilation step happens before the code is run.
* JavaScript: JavaScript is an interpreted language. It is executed directly by web browsers or server-side runtime environments like Node.js without a separate compilation step.

Ecosystem:

* Java: Java has a large and mature ecosystem with a wide range of libraries and frameworks for various types of applications.
* JavaScript: JavaScript has a rich ecosystem for web development, with popular libraries and frameworks like React, Angular, and Node.js.

Syntax:

* Java: Java's syntax is more similar to languages like C++ and C#. It uses curly braces {} for blocks and requires a semicolon at the end of statements.
* JavaScript: JavaScript's syntax is influenced by languages like C and Java but is more lightweight. It does not require semicolons (although they are often used for readability), and it uses curly braces for blocks as well.

Object Models:

* Java: Java uses a class-based object-oriented programming model, where objects are instances of classes.
* JavaScript: JavaScript uses a prototype-based object-oriented programming model, where objects can inherit properties and methods directly from other objects.

The confusion between Java and JavaScript often arises from their similar names. They were intentionally named to capitalize on Java's popularity when JavaScript was created by Netscape in the 1990s. However, their fundamental differences in purpose, syntax, and usage make them two entirely separate languages.

To summarize, Java is a statically-typed, compiled language used for a wide range of applications, while JavaScript is a dynamically-typed, interpreted language primarily used for web development.

**What is a paradigm?**

In the context of programming languages and computer science, a paradigm refers to a fundamental style or approach to solving problems and designing software. It encompasses a set of principles, concepts, and practices that guide how code is written, organized, and executed. Paradigms provide a high-level structure for programming languages and influence how developers think about and approach problem-solving in the software development process.

There are several major programming paradigms, each with its own philosophy and characteristics. Some of the most common paradigms include:

* Imperative Programming: In imperative programming, the focus is on specifying a sequence of actions or commands that the computer should perform to achieve a desired outcome. It emphasizes changing program state through statements that modify variables.
* Functional Programming: Functional programming treats computation as the evaluation of mathematical functions and avoids changing-state and mutable data. It emphasizes immutability, pure functions, and higher-order functions.
* Object-Oriented Programming (OOP): Object-oriented programming organizes code around objects, which are instances of classes that encapsulate both data (attributes) and behaviour (methods). It promotes concepts like inheritance, encapsulation, and polymorphism.
* Procedural Programming: Procedural programming focuses on defining procedures (functions or routines) that perform specific tasks. It is often used to break down problems into smaller, manageable steps.
* Declarative Programming: Declarative programming emphasizes specifying what needs to be done rather than how to do it. It describes the desired outcome, and the underlying system or language figures out the best way to achieve it. SQL is an example of a declarative language for database querying.
* Logical Programming: Logical programming is based on formal logic. It involves defining relationships and rules, and the program uses inference engines to deduce conclusions based on the provided logic. *Prolog* is a notable example.
* Event-Driven Programming: Event-driven programming focuses on handling events or messages that occur within a system, often in response to user interactions or external stimuli. It is commonly used in graphical user interfaces and asynchronous systems.
* Parallel and Concurrent Programming: These paradigms deal with designing programs that can execute multiple tasks simultaneously (parallelism) or manage multiple tasks concurrently (concurrency). They are essential for leveraging multi-core processors and distributed systems.

🧩 Vocabulary: Terminology and Programming languages (Part 1)

LOOK INTO - FIGURE OUT - RAN INTO - LOG IN –   
WORKING ON - CARRY OUT - SET UP - BRING UP - ENCOUNTER

1. The team is currently \_\_\_\_\_\_\_\_\_\_ a new feature to improve user experience.
2. The developers \_\_\_\_\_\_\_\_\_\_ some unexpected bugs while testing the application.
3. Before you start coding, make sure to \_\_\_\_\_\_\_\_\_\_ your development environment correctly.
4. We need to \_\_\_\_\_\_\_\_\_\_ why the server crashed to prevent it from happening again.
5. He managed to \_\_\_\_\_\_\_\_\_\_ the complex algorithm and optimize it for better performance.
6. Don't hesitate to \_\_\_\_\_\_\_\_\_\_ any issues you face during the development process.
7. The company decided to \_\_\_\_\_\_\_\_\_\_ a new project that aligns with the latest market trends.
8. As a developer, you often \_\_\_\_\_\_\_\_\_\_ challenges that require creative solutions.
9. Users have to \_\_\_\_\_\_\_\_\_\_ to access their accounts and manage their preferences.

LOOK INTO - CARRY OUT - COMING UP WITH - WORKING ON - BROUGHT UP - RAN INTO - TAKE OVER - FIGURE OUT - SET UP - LOG IN

1. The software developer is ........................... a new algorithm to improve the system's efficiency.
2. When implementing the feature, they .................................. some compatibility issues with older browsers.
3. The team needs to ................................. thorough testing before releasing the application to the public.
4. She will ................................. to the server remotely to check the error logs and diagnose the problem.
5. The team is ................................. integrating the latest API into the application to enhance its functionality.
6. The programmer managed to ................................. a clever workaround for the memory leak issue.
7. During the code review, they ................................. some concerns about the security of the login system.
8. Before starting the project, the team will ................................. a version control system for collaboration.
9. The intern is eager to ................................. the maintenance of the project from the senior developer.
10. The IT department will ................................. the performance bottlenecks and optimize the database queries.

IMPLEMENTED - COMPILED - DEBUGGED - REFACTORED - MIGRATED - OPTIMIZED - DEPLOYED - COMMENTED - COLLABORATED - INHERITED

1. The programmer ......................... the code to identify and fix the errors.
2. The team ......................... effectively using version control, which streamlined the development process.
3. She ......................... the source code to create the executable program for testing.
4. The software engineer ......................... the algorithm to reduce the time complexity.
5. The developer ......................... the codebase to improve readability and maintainability.
6. They ......................... the web application to a cloud server for production use.
7. The team ......................... the new feature according to the specifications provided.
8. He ......................... the code extensively to explain its functionality to other developers.
9. The company ......................... its database to a more robust and scalable system.
10. The programmer ......................... a legacy system and worked on modernizing it.

🧩 Vocabulary: Terminology and Programming languages (Part 2)

INTEGRATED – PARTITIONED – VALIDATED – INTERFACED – ABSTRACTED – DEPRECATED – SYNCHRONIZED – MODULARIZED – AUTOMATED – ENCOUNTERED

1. The software engineer ........................... the codebase to break it into smaller, reusable components.
2. They ........................... with external APIs to fetch data for the application.
3. The team ........................... the testing process to ensure faster and more reliable quality checks.
4. She ........................... a challenging bug that required extensive troubleshooting.
5. The programmer ........................... the data between the local database and the cloud server.
6. They ........................... user inputs to prevent potential security vulnerabilities.
7. The developer ........................... the database to optimize query performance.
8. The team ........................... various third-party libraries to leverage existing functionalities.
9. He ........................... the complex logic into separate functions for better code organization.
10. The company ........................... support for older browsers to focus on modern web standards.

ENCAPSULATED – OVERRODE – VALIDATED – PARSED – PROFILED – SOURCED – ENCOUNTERED – SCHEDULED – INDEXED – INTEGRATED

1. The team ............................... the software design through extensive user testing and feedback.
2. They ............................... the default settings to customize the application's behaviour.
3. The developer ............................... the data from the API response to extract relevant information.
4. She ............................... external libraries to add advanced functionalities to the project.
5. The software engineer ............................... sensitive data to ensure data security.
6. They ............................... the database to improve query performance for large datasets.
7. The programmer ............................... a critical bug and quickly resolved it before deployment.
8. He ............................... the application to identify performance bottlenecks and optimize resource usage.
9. The team ............................... the application with third-party payment gateways for seamless transactions.
10. They ............................... regular backups to protect the data from potential data loss.

POLYMORPHISM – ASYNCHRONOUS – FRONT-END – BACK-END – ENCAPSULATION – PARSING – CRYPTOGRAPHY – DEPENDENCY INJECTION – REGRESSION – FRAMEWORK

1. The software engineer ................................. common functionalities into reusable libraries for future projects.
2. They ................................. the open-source repository and contributed bug fixes to the community.
3. The developer ................................. the debugging information to track the application's behaviour.
4. She ................................. the codebase to follow the industry's best practices and design patterns.
5. The team ................................. the software requirements with the stakeholders before starting development.
6. They ................................. the legacy system to a cloud-based infrastructure for scalability and cost-effectiveness.
7. The programmer ................................. automated tests to ensure continuous integration and delivery.
8. He ................................. the application to test its behaviour in a controlled environment.
9. The company ................................. application performance and user activity to identify potential issues.
10. They ................................. the user interface based on feedback from usability testing.

DOCUMENTED – SYNCHRONIZED – PATCHED – COMPILED – SCOPED – PRIORITIZED – DESIGNED – AUTHENTICATED – VERSIONED – MINIFIED

1. The software engineer ............................... the codebase using Git to keep track of changes over time.
2. They ............................... the security vulnerability and released an urgent update to all users.
3. The developer ............................... the data across multiple servers for load balancing.
4. She ............................... the database schema to ensure efficient data storage and retrieval.
5. The team ............................... the API endpoints and usage for external developers.
6. They ............................... users using OAuth for secure access to third-party services.
7. The programmer ............................... the CSS and JavaScript files to reduce loading times.
8. He ............................... the project requirements and estimated the timeline for completion.
9. The company ............................... bug fixes and feature requests based on user feedback.
10. They ............................... the code to target multiple platforms and operating systems.

🧩 Vocabulary: Terminology and Programming languages (Part 3)

ALGORITHM – DATABASE – COMPILER – API – DEBUGGING – FUNCTION – FRAMEWORK – INTERFACE – VARIABLE – VERSION CONTROL

1. The software engineer used a sorting ............................... to efficiently organize the data.
2. They integrated a third-party ............................... to fetch real-time weather information for the application.
3. The developer spent hours ............................... the code to identify and fix the issues.
4. She decided to use a popular JavaScript ............................... to speed up development.
5. The ............................... translates the high-level code into machine code for execution.
6. The application stores user information in a secure and scalable ................................
7. The programmer declared a ............................... to hold user input and process it later in the code.
8. He created a custom ............................... to calculate and return the area of a circle.
9. The team defined a user-friendly ............................... to interact with the complex backend logic.
10. They adopted ............................... to track changes and collaborate efficiently among team members.

OBJECT-ORIENTED – BACK-END – FRAMEWORK – FRONT-END – BIG O NOTATION – SYNTAX – RECURSION – ALGORITHM – AGILE – IDE

1. The developer used an .............................. to find the shortest path in the graph.
2. They implemented .............................. to solve the problem by dividing it into smaller subproblems.
3. The team follows the .............................. methodology to ensure iterative and incremental development.
4. She fixed the error in the code by correcting the .............................. mistake.
5. The .............................. provides a set of tools and features that enhance productivity during development.
6. The website's .............................. was designed to provide a smooth and intuitive user experience.
7. They set up a secure .............................. server to handle database queries and user authentication.
8. He chose to use a popular JavaScript .............................. for building web applications.
9. The programmer adopted .............................. programming to model real-world entities in the software.
10. The team analyzed the .............................. of the algorithm to understand its time complexity.

EXCEPTION – TESTING – DEPLOYMENT – API – GIT – DEBUGGING – INHERITANCE – LAMBDA – ABSTRACTION – SCALABILITY

1. The software architect used ................................ to hide the underlying complexity of the system's components.
2. They managed version control using ................................ to track changes and collaborate with the development team.
3. Before the release, thorough ................................ ensured the software's reliability and correctness.
4. The application provided an open ................................ for developers to integrate additional functionalities.
5. She handled the ................................ to gracefully recover from unexpected errors during runtime.
6. The developer implemented ................................ to create a new class based on an existing one with additional features.
7. They used a ................................ function to simplify and condense a block of code for repeated use.
8. He spent hours ................................ the code to find and fix the elusive bug.
9. The team successfully completed the ................................ of the web application to a cloud server.
10. The software architecture was designed for ................................, ensuring smooth performance as the user base grows.

POLYMORPHISM – ASYNCHRONOUS – FRONT-END – ENCAPSULATION – BACK-END – CRYPTOGRAPHY – PARSING – DEPENDENCY INJECTION – REGRESSION – FRAMEWORK

1. The website's ........................... was designed to provide a seamless user experience and an attractive interface.
2. They used a popular JavaScript ........................... to simplify the development of complex web applications.
3. The team implemented ........................... to facilitate loose coupling between different components.
4. The application leverages ........................... to handle multiple data types and improve code reusability.
5. She specialized in ........................... and implemented secure communication protocols for data protection.
6. During ..........................., the program analyzes and processes structured data from external sources.
7. The team performed ........................... testing to ensure new updates did not break existing functionalities.
8. The ........................... server handles data processing, database management, and business logic.
9. They applied ........................... to hide internal implementation details and provide a clean interface
10. The application uses ........................... programming to improve responsiveness and performance.

AGILE – MICROSERVICES – AUTHENTICATION – STATIC TYPING – CONCURRENCY – MIDDLEWARE – STACK – LINTER – COMPILATION – ABSTRACTION

1. The team adopted an ......................... approach to continuously improve the software based on user feedback.
2. They implemented ......................... mechanisms to secure user access to sensitive data.
3. The software architecture follows a ......................... approach to modularize and scale components independently.
4. She analyzed the ......................... of technologies used in the project to optimize performance.
5. During ......................... , the source code is translated into machine code for execution.
6. The development team integrated a ......................... to enforce coding standards and maintain code quality
7. They implemented ......................... to catch type-related errors during development.
8. The application efficiently manages ......................... to handle multiple tasks simultaneously.
9. The software system uses ......................... to mediate communication between different components.
10. ......................... allows developers to interact with complex systems using simplified interfaces.

# UNIT 3.4: Frameworks & Libraries

Theory: Frameworks & Libraries

In computer science and programming, frameworks and libraries are both tools that developers use to simplify and streamline the software development process. However, they serve different purposes and have distinct characteristics:

Framework:

A framework is a pre-built, comprehensive, and structured software architecture that provides a foundation for building applications. It includes a set of rules, guidelines, and templates that dictate the overall structure and flow of your application.

Frameworks are more opinionated and tend to impose a specific design pattern or approach on your project. They often have a fixed architecture that you must follow.

Frameworks typically provide a range of built-in functionalities and tools, such as libraries, APIs, and code generators, to help developers with common tasks like handling web requests, database interactions, and user authentication.

Developers build their applications within the framework's constraints and extend their functionality as needed.

Example: Ruby on Rails, Django (Python), Angular (JavaScript/TypeScript).

Library:

A library, on the other hand, is a collection of pre-written code or functions that can be used by developers to perform specific tasks or operations. Libraries are typically more focused and specific in their functionality.

Libraries are not opinionated about the overall structure of your application. Instead, you use them as needed to augment your application's capabilities.

Developers have more flexibility when using libraries because they can be integrated into an application in a modular way. You can pick and choose which parts of a library to use.

Libraries do not provide a strict structure or flow for your application; they are often used to solve particular problems or add specific features.

Example: jQuery (JavaScript), NumPy (Python), Alamofire (iOS), JavaFX (Java).

In summary, the main difference between a framework and a library lies in their level of abstraction and control. Frameworks provide a higher level of abstraction and a more structured environment, while libraries offer lower-level functionality that can be used as building blocks within your own code. The choice between using a framework or a library depends on your project's requirements, development preferences, and the level of control you need over your application's architecture.