Common features of programing languages

A programming language issues a series of commands that help computers, mobile phones, tablets, and other electronic devices function as intended and perform various tasks. There are many types of programming languages, and the correct one must be chosen based on the philosophy and objectives of a particular device or program. In this article, we will explore a range of programming languages and the features that distinguish them from one another. We will also examine the importance of programming languages in the field of integration.

**What is a programming language?**

A **programming language** is a set of grammatical rules (both syntactic and semantic) that instruct a computer or a device to behave in a certain way. Each programming language has a vocabulary—a unique set of keywords that follows a special syntax to form and organise computer instructions.

**Differences between natural language and programming language**

Natural languages are spoken by people, while programming languages are intended for machines. Both languages contain important similarities, such as the differentiation they make between syntax and semantics and the existence of a basic composition. Essentially, the two types were created to communicate ideas, expressions, and instructions.

Whilst there are several other similarities and points in common between them, it is also possible to identify some of their differences.

|  |  |
| --- | --- |
| **Natural Languages** | **Programming Languages** |
| More ambiguous. Human beings have the ability to clarify the meaning of an expression. The built-in redundancy of human languages allows some ambiguity to be resolved using context. | Stricter and less tolerant. Computers are very precise about the instructions they like to receive. Therefore, programming languages have practically no redundancy to prevent ambiguity and issue the correct commands. |
| Are open and allow combinations without the risk of making mistakes. | Are closed and fixed to avoid confusion and mistakes. |

**Programming languages: Types and Features**

The evolution of computers has led to the creation of hundreds of different programming languages for various types of development. The field of programming is vast, so the use of a particular language will depend on the objectives to be achieved.

**Types of programming languages**

Programming languages can mainly be classified as low-level and high-level programming languages. Although simple compared to human languages, high-level languages are more complex than low-level languages. At the same time, a high-level language affords more readability in comparison to its low-level counterpart, which needs specialist knowledge in computer architecture to interpret.

* **Low-level languages** include assembly and machine languages.
* An **assembly language** contains a list of basic instructions and is much harder to read than a high-level language. It is just one level above machine code in terms of abstraction, using simple codes that are easily converted to strings of 1s and 0s (binary representation). It cannot be used to structure and manipulate complex information.
* **Machine language** is directly understood by the computer’s processing unit. A programmer will first write his code in a high-level language, then compile it into a machine-readable format where instructions are represented in binary.
* **High-level languages**, on the other hand, are designed to be easy to read and understood, allowing programmers to write source code using logical, meaningful words and symbols. They encapsulate everything from early algorithmic languages such as FORTRAN to more widespread, object-oriented languages like C++, C#, and Java.

The following activities can be performed using high-level programming languages:

* Programs and applications development.
* Artificial intelligence development.
* Database development.
* Video game development.
* Development of drivers and hardware interface.
* Internet and web pages development.
* Script development.

**Main features of programming languages**

The popularity of a programming language depends on the features and utilities it provides to programmers. But, what are the main features of programming languages? Here are the main features that a programming language must possess to stand out from all the rest:

* **Simplicity:** the language must offer clear and simple concepts that are easy to understand, facilitating learning and application. But simplicity can be a difficult balance to strike without compromising the overall capability of the language.
* **Capability:**apart from being easy to use, the language must be well-equipped with a robust set of features to perform a wide range of tasks. If a programming language was designed to be used in a specific area, it must provide the necessary means (operators, structures, and syntax) to achieve ideal results.
* **Abstraction**: it is the language’s ability to define and use complicated structures or operations while ignoring certain low level details.
* **Efficiency**: programming languages that can be translated and executed efficiently help avoid the excessive consumption of memory and time.
* **Structuring**: the language allows programmers to write their code according to structured programming concepts to avoid creating errors.
* **Compactness**: a language with this characteristic can express operations concisely without having to write too many details.
* **Principle of Locality**: also known as the **locality of reference**, this phenomenon describes a computer program’s preference for continually accessing the same areas of memory over a short span of time. By enabling the usage of loops and subroutines, a programming language can exploit the principle of locality for optimising the overall performance of an application.

**What is the best web programming language?**

Web development can be done through different programming languages that allow you to build a site or design a web application. Defining which is the best web programming language is complicated because it depends on several factors. However, the following options might be worth considering:

* **Java:** multipurpose language that adjusts efficiently to web development.
* **Go:** this is a general-purpose, flexible language that facilitates the creation of applications.
* **Ruby on Rails:** allows you to design web applications quickly.
* **Python:** it works in a wide variety of contexts and is especially suitable for the web due to its technical advantages.
* **JavaScript:** it is on the client’s side and can be extended to the server for different functions.

**Why are programming languages important for Integration?**

The importance of programming languages in integration mainly boils down to the variety of options you can choose from. Before deciding on one of them, it is important to consider two points. First, an enterprise possesses many digital assets or systems that are often intricate in design. Consider the numerous databases, APIs, applications, and hardware—both on-premise and on the cloud—that must communicate with each other. Secondly, each company differs from the rest in terms of its size, goals, internal hierarchies, and so on. Thus, the right choice for an integration solution depends on a case-by-case analysis.

In some scenarios, especially when a company requires centralised management of distributed systems, a traditional ESB would be appropriate. Here, we prioritise the usage of an ESB configuration language (such as the XML-based language used to configure Synapse ESB).

In other scenarios, the extensive feature set of modern-day programming languages could eliminate the bottlenecks of legacy technologies such as the ESB, making them better alternatives. These programming languages are:

* **Simple to use**
* **Safe and resilient**
* **Agile**

In particular, web programming languages have a major role to play in the evolution of integration. For instance, **Go** is used along with Kubernetes to create microservices, which are a fine-grained version of service-oriented architectures that rely on ESBs for communication. Similarly, **Ballerina** is a programming language proposed by WSO2 for the improved development of cloud-native, distributed applications that could substitute ESB technologies in certain cases.

**Conclusion**

To sum up, all programming languages share some common features without surrendering their individual identities. Although every language is made to direct digital systems towards producing desired outcomes, some of them shine in their own niche while others have a more generalist application.

If you have an elaborate problem to solve, such as an integration project you’d like to implement, finding a suitable programming language could be the first step towards a remedy.