Scripting Languages vs Programming Languages vs Markup

Scripting Languages?

Scripting languages can perform different actions within a particular runtime environment, such as automating task execution, enhancing the functionality of the parent software, performing configurations, extracting data from data sets, and others

Platform-Specific vs Platform-Agnostic

Scripting languages are platform-specific, while programming languages are platform-agnostic (cross-platform) as they have the ability to execute themselves. For instance, you can run a Java program on any operating system.

(Mostly) Interpreted vs Compiled

While programming languages are compiled, scripting languages are mostly interpreted — even though there are some scripting languages that are both compiled and interpreted, such as Python and Groovy.

‘Compiled’ means that a programming language has its own compiler that translates the syntax into machine code before runtime.

In contrast, scripting languages are interpreted line by line during runtime by the interpreter of the platform they are running on.

Faster vs Slower at Runtime

Because of this difference in implementation, programming languages run faster than scripting languages as they don’t have to be compiled in real-time

More vs Less Code-Intensive

Programming languages are more code-intensive as you have to do many things manually that are handled by the platform in the case of scripting languages

Scripting languages	Programming languages
Platform-specific	Platform-agnostic (cross-platform)
(Mostly) interpreted	Compiled
Slower at runtime	Faster at runtime
Less code-intensive	More code-intensive
Creates apps as part of a stack	Creates standalone apps

Markup language

refers to a text-encoding system consisting of a set of symbols inserted in a text document to control its structure, formatting, or the relationship between its parts

for example XML

the widely used HTML