

FUTURA

LA SCUOLA PER L'ITALIA DI DOMANI



Finanziato
dall'Unione europea
NextGenerationEU



Ministero dell'Istruzione
e del Merito



Italiadomani
PIANO NAZIONALE DI RIPRESA E RESILIENZA

DOCENTE	Shadi Lahham
Corso	Web Developer
Unità Formativa	Programmazione - Javascript e Typescript
Argomento	Specificato nel titolo della slide successiva

Intervento realizzato da
ITS
TECNOLOGIE
DELL'INFORMAZIONE
E DELLA COMUNICAZIONE

COESIONE
ITALIA 21-27
PIEMONTE



Cofinanziato
dall'Unione europea



REGIONE
PIEMONTE

Javascript

Language introduction

Shadi Lahham - Web development

Language

Javascript

- A programming language used to make web pages interactive
- Runs **client-side** in the visitor's browser
 - Client-side code is code that runs on the user's computer
- Responsible for the "behavior" of a Website
- Constitutes the third layer of the standard web technologies layer cake, alongside HTML and CSS

Javascript

JavaScript allows you to implement complex things on web pages such as

- Updating website content (e.g. news updates)
- Interactive maps
- Drawing and animation
- Image galleries and lightboxes
- Full featured web applications
- Keep track of users with cookies
- Interactive elements like tabs, sliders and accordions

Node.js

- Node.js is a platform built on Chrome's JavaScript runtime for easily building fast and scalable network applications
- Node.js runs **server-side**. Server-side code runs on the server, then its results are downloaded and displayed in the browser
- Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient, perfect for data-intensive real-time applications that run across distributed devices

Compiled & interpreted languages

Compiled & interpreted languages

- Interpretation and compilation are characteristics of how a language is implemented
- It's incorrect to categorize a language as solely interpreted or compiled because these processes depend on the implementation rather than inherent properties of the language
- Therefore, any language could potentially be interpreted or compiled, depending upon the specific implementation being utilized

What exactly is compilation?

- In the compiled execution of a programming language, the compiler converts the program directly into machine code tailored to the target machine, referring to code designed for a particular processor and operating system
- Subsequently, the computer independently executes this machine code.

What exactly is interpretation?

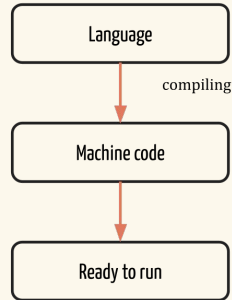
- In an interpreted implementation, the source code isn't directly run by the machine. Instead, another program, known as the interpreter, reads and executes it
- This interpreter is tailored for the native machine
- For example, when encountering the "*" operation, the interpreter calls its own "multiply(x,y)" function, which then executes the machine code's equivalent instruction

Too many words!

Compiled vs interpreted languages

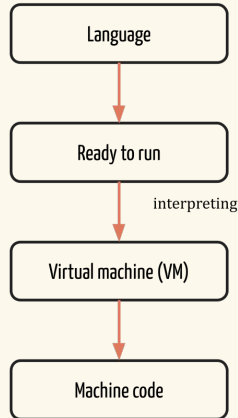
Compiled

Examples of compiled languages
C, C++, Fortran, Pascal



Interpreted

Examples of interpreted languages
Python, PHP, Ruby, Javascript



Comparison

Interpreted

Requires interpreter

Interpreted on the fly

Platform independent

Compiled

Requires compiler

Depends on platform

Slow compilation

Interpreted: advantages & disadvantages

Advantages

- Easy to learn and use
- More portable
- Allow complex tasks to be performed in relatively few steps
- Allow simple creation and editing in a variety of text editors
- Allow the addition of dynamic and interactive activities to web pages
- Editing and running of code is fast

Interpreted: advantages & disadvantages

Disadvantages

- Usually run slower
- Limited access to low level and speed optimization code
- Limited commands to run detailed operations on graphics
- Limited access to the device

Compiled: advantages & disadvantages

Advantages

- Fast execution
- Optimised for the target hardware

Disadvantages

- Require a compiler
- Editing and deploying the code is a lot slower than interpreters

Compiled & interpreted error handling

Compiled Languages

- Errors caught at compile time: syntax, type mismatches
- Reduces runtime errors
- Logic errors discovered during runtime
- Errors less likely to affect end-users if thoroughly tested

Interpreted Languages

- Errors exposed at runtime due to direct execution, not compilation
- Runtime-dependent behavior: environment variations like browsers, OS
- Low-probability errors harder to replicate without extensive testing
- Untested runtime errors often affect end-users directly

Your turn

1.Languages

- Make a list of all the programming languages that you know
- Classify the languages into the groups: compiled, interpreted, other
- For each language, explain why it is compile, interpreted or other
- Try to find additional programming languages and add them to the list

Create a folder named **01-languages**

Inside the folder create a **.txt** or **.doc** or **.md** file with your answers

Note: all files should be in kebab-case (italiano)

2.Levels

Read the following articles and write a short summary in Italian or English

- [Compiler and Interpreter Critical Differences](#)
- [Levels of Programming Languages](#)
- **Bonus** [Machine Language vs. Assembly Language](#)

Create a folder named **02-levels**

Inside the folder create a **.txt** or **.doc** or **.md** file with your answers

Note: all files should be in [kebab-case](#) ([italiano](#))

© Copyright & Attribution

Unless otherwise stated, all materials are © 2017–2025 [Shadi Lahham](#)

For personal use only. May not be shared or reproduced without written permission

Brief excerpts may be used with proper attribution

External links and resources are copyrighted by their respective owners