

TypeScript

Docente

Mario Perna

prof.perna.mario@darzo.net

A.S.

2025/2026

Materia

TEPSIT (Laboratorio)

Introduzione

TypeScript, sviluppato da Microsoft nel 2012, nasce come **superset di JavaScript** con l'obiettivo di superare alcuni limiti.

Molti programmatori provenienti da linguaggi orientati agli oggetti trovavano infatti difficile adattarsi alla **manca di controlli statici sui tipi**.

JavaScript è effettivamente un linguaggio **tipizzato**, ma in modo **dinamico**: questa caratteristica garantisce grande flessibilità, ma allo stesso tempo espone a errori che possono risultare complessi da individuare e correggere.

Vantaggi

- **Tipizzazione statica** → riduce gli errori a tempo di compilazione
- **Code completion e refactoring** più efficaci negli editor/IDE
- **Supporto alle funzionalità OOP** (classi, interfacce, ereditarietà)
- **Maggiore leggibilità e manutenibilità** del codice
- **Compatibilità totale con JavaScript** → può essere integrato gradualmente
- **Ampia adozione e supporto** → community e strumenti in continua crescita

Ambiti di applicazione

- **Sviluppo di Giochi:** in campo gaming, TypeScript può essere utilizzato con framework come Phaser o Babylon.js per la creazione di giochi web o giochi basati su browser.
- **Sviluppo Web Back-end:** anche sul lato server, TypeScript è utilizzato in progetti Node.js per migliorare la robustezza e la manutenibilità del codice.
- **Sviluppo di App Mobile:** con l'aiuto di framework come React Native o NativeScript, TypeScript può essere utilizzato per sviluppare app mobili cross-platform.

Funzionamento

Codice TypeScript

- File .ts – uso di: classi, interfacce, moduli e tipi.



Compilazione / Transpilazione*

- Uso di TypeScript compiler (tsc)
- Target: ES3/ES5/ES6



Codice JS puro (vanilla)

- JavaScript file (*.js). Codice eseguibile ovunque nei browser ed in ambiente NodeJS

Un **transpiler** è un **tipo di compilatore** che traduce il codice sorgente scritto in un linguaggio di programmazione in un altro linguaggio di programmazione, mantenendo lo stesso livello di astrazione.

Funzionamento

Quando scriviamo codice in **TypeScript**, non possiamo eseguirlo direttamente nel browser o in Node.js, perché questi ambienti leggono solo **JavaScript**.

Per questo entra in gioco il **compilatore di TypeScript (tsc)**, che svolge due compiti:

- Controlla la correttezza del codice e segnala eventuali errori di tipo
- Traduce il file **.ts** in un file **.js** eseguibile.

Questa traduzione non abbassa il linguaggio a un livello più vicino alla macchina (come avviene ad esempio con C o Java), ma converte TypeScript in un linguaggio **allo stesso livello di astrazione**, cioè JavaScript.

Per questo si parla più precisamente di **transpilazione**: una **trasformazione tra linguaggi simili**, dove vengono rimosse le annotazioni di tipo ma la logica del programma rimane identica.

Sitografia

<https://www.typescriptlang.org/docs/handbook/2/basic-types.html>

<https://stackoverflow.com/questions/44931479/compiling-vs-transpiling>