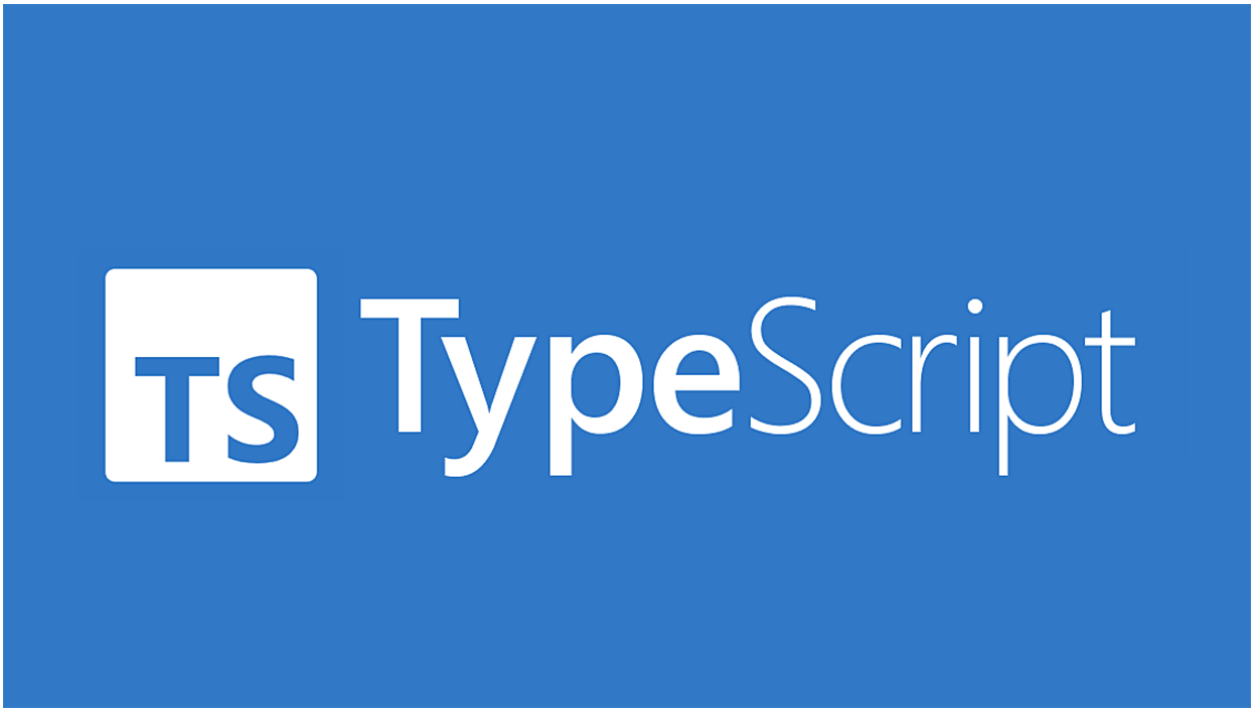
GETTING STARTED WITH TYPESCRIPT

Getting Started with TypeScript: Key Concepts



Introduction to TypeScript

- **TypeScript** is a superset of JavaScript that incorporates static typing and additional features to enhance code robustness and conciseness.

Types of Programming Languages

- **Statically-typed Languages**: The type of variables is determined at compile-time and cannot be changed later. Examples include C++, C#, and Java.

- **Dynamically-typed Languages**: The type of variables is determined at run-time and can change later. Examples include Python, JavaScript, and Ruby.

Benefits of Using TypeScript

- **Static Typing**: Helps catch more bugs at compile-time, improving code reliability.
- **IntelliSense**: Most IDEs and code editors offer excellent auto-completion and active hints, boosting productivity.
- **Refactoring Support**: Better refactoring capabilities due to type information, allowing for structural changes without altering behavior.

Key Features and Tools

- **Type Safety**: Ensures variables are used consistently according to their defined types.
- **Transpiling**: Browsers cannot execute TypeScript directly; it needs to be transpiled into JavaScript using the TypeScript compiler.
- **Source Maps**: These files map TypeScript code to the corresponding JavaScript code, aiding in debugging.
- **Configuration**: The TypeScript compiler can be configured using the `tsconfig.json` file to enable or disable various settings.

Summary

- TypeScript enhances JavaScript by adding static typing and other features.
- It supports better code refactoring, IntelliSense, and type safety.
- TypeScript needs to be transpiled into JavaScript for browser execution, and source maps facilitate debugging.