Introduction

In this lesson, we will talk about strictness of typing and see how to enable strict flags.

WE'LL COVER THE FOLLOWING

Overview

The TypeScript compiler is very configurable. There are tens of options that regulate different aspects of type checking and compilation. In this chapter, we're going to look at some of the compiler flags that let you decide how strict the compiler should be when enforcing type safety.

We use TypeScript because it helps us detect errors at compile time before they hit production. The stricter the checks TypeScript performs, the more errors it will be able to catch. Therefore, we should aim to make the checks as strict as possible. This can be achieved by setting the `strict` compiler flag to `true`. The `strict` flag is responsible for enabling a group of several flags. We're going to focus on the most important flags that will have the biggest impact on how you write code.

Enabling compiler flags

Enabling TypeScript compiler flags is very straightforward. Your project should contain a file called `tsconfig.json` on the top level. Inside the file, there should be a section called `compilerOptions`. Inside this section, you can add Boolean properties for every flag that we discuss in this course.

For example: to enable `strictNullChecks`, you should modify `tsconfig.json` to look like this:

```
{
  // some other settings

  "compilerOptions": {
    // some other options
    "strictNullChecks": true,
  }
}
```

If you're using an IDE such as Visual Studio Code or WebStorm, it should automatically pick up the changes you made in the file.

In the next lesson, we'll talk about how enabling the `noImplicitAny` flag increases strictness of type checking.