Choosing Configuration and Compilation Options

Chris B. Behrens
Senior Software Architect

@chrisbbehrens

How We've Been Doing Things So Far

Ground-up

This is how you learn when you just wander around (a good thing)

The Limits of This Approach

The ideal:

An entirely intuitive and lore-free system

That's just not realistic

A firm foundation, I hope

Details from here on out

Running the Compiler and Using Compiler Options

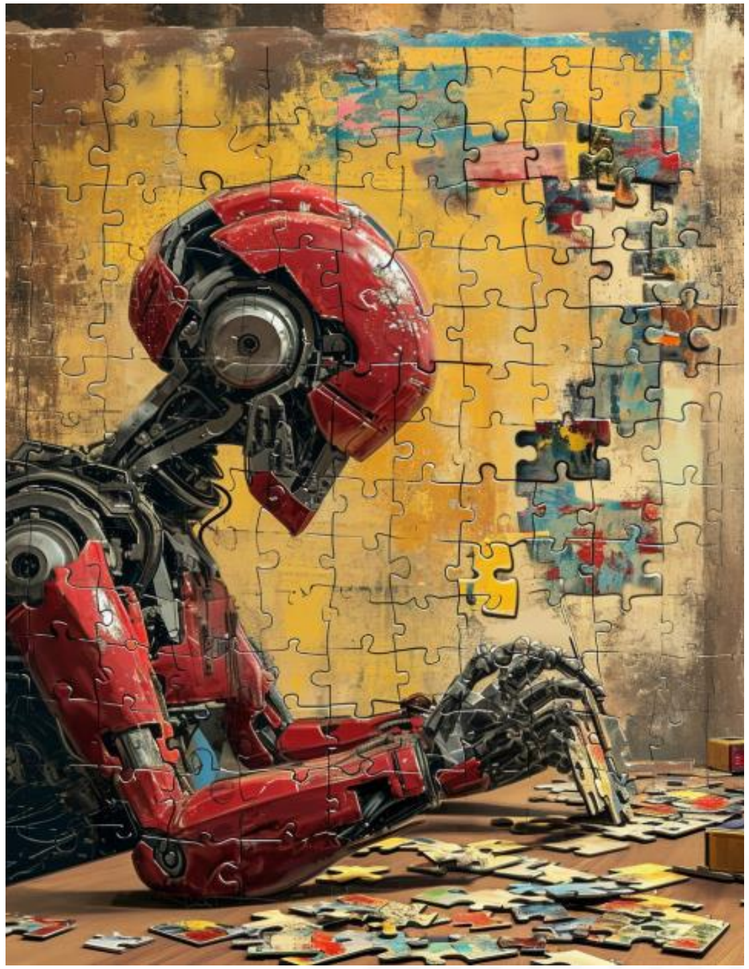
TypeScript Compiler Essentials

Here's what we know

"translate"

No execution without transpilation

The Compilation Process



A good understanding of the process

Read the file, enforce the rules

And translate it into a *syntax tree*

A form independent of the language

ts2python - https://pypi.org/project/ts2python

https://bit.ly/3tM4J0p

Static analysis and more rules to enforce

Demo: A Closer Look at Our Transpilation

The abstract syntax tree

For our Person class

Walk through the elements therein

So, we can better understand what the compiler is doing

Configuring Your Project with tsconfig Files

Some Go-tos in TSConfig

target

module

allowJs

removeComments

newline

strict

Try to always make it true

Maybe set it to false while you get stuff straightened out

An umbrella option

Turn on as many of the options as you can at least

Demo: Working with tsconfig Files

- The structure of the compilation options
- Demonstrate how Visual Studio enforces the target attribute
- The other stuff you can do in tsconfig files
- Make life easier for you

Using Type Declaration Files

The Two Definitions

Runtime code

A type definition

Third-party libraries only guarantee runtime code

The Good News

Many packages *are* including type definitions

***Any* at runtime (functionally)**

Generating type definitions is trivial when you're working in ts

The Better News

You can usually get it somewhere else

https://github.com/DefinitelyTyped

```
npm install --save-dev @types/react
```

Where That Will Get You

Pretty far

A major library – either supplied or supported

A narrow window for one with no support

Demo: Inferring Types from Type Declaration Files

How to generate type definition files

To sit alongside your emitted JavaScript

Untyped – a simple JavaScript module I created

How we can get type definitions available

Use strong typing in TypeScript

What’s Going on Here

```
import { untypedCalculator } from "./untyped"

/* Yes, it is pointing at the untyped.js file or it is
pointing at the untyped.d.ts file, depending on whether
it’s at runtime or design-time */
```

Summary

Some compilation API magic

The abstract syntax tree that the compiler generates

The details of the compiler options in the tsconfig

The other options like file, include, and exclude in tsconfig

All about type declarations

- How to consume them
- How to get them
- How to create them manually as a last resort