All Aboard the Type Train

Kadi Kraman

@kadikraman

Formidable

Why this talk?

Why add types to JavaScript?

Should you use Flow or TypeScript?

... or something else entirely!

All programming languages have a type system

The difference is *when* the type-checking is done

Strong vs Weak

Languages are often colloquially referred to as *strongly typed* or *weakly typed*.

There is no universally accepted definition of what these terms mean.

Static vs Dynamic

Static (e.g. Go, C#, Haskell)

Types are checked before run-time

Dynamic (Python, Lua, Objective C)

Types are checked at run-time, during execution

JavaScript is a Dynamically Typed language

Static typing

(not JavaScript)

Pros

More errors found earlier in development

Fewer errors at run-time and shipped code

No need to write tests for "type-correctness"

Cons

Verbose type declarations

Complex error messages

Excessive boilerplate

Dynamic typing

(e.g. JavaScript)

Pros

Reduces clutter and repetition in code

Implicit polymorphism (the ability to write a single function that handles many data-types)

Cons

More errors detected during run time and in shipped code

Need to write tests for type correctness

DISCLAIMER!

Additional type-checking will not make code bug-free

It is **not** a replacement for testing your code

It **only** helps reduce **type errors**

How to make JavaScript more type-safe?

1.

Static code analysis

2.

Statically typed language that compiles to JavaScript

Flow - static type checker

Infers type information from existing code

```
const four = 4;
const result = four.split('');
```

```
Cannot call four.split because property split is missing in Number.
```

You can choose to enforce types

```
function randomString(): string {
  return 4;
}
```

```
Cannot return 4 because number is incompatible with string.
```

(Atom, with Flow plugin)

2. Statically typed language that compiles to JavaScript

TypeScript - a superset of JavaScript

Microsoft 2012

Infers type information from existing code

```
[ts] Property 'split' does not exist on type '4'. [2339]
any
const four = 4;
const result = four.split('');
```

You can choose to enforce types

```
[ts] Type '4' is not assignable to type 'string'. [2322]
fu
  return 4;
}
```

(VSCode, with TypeScript plugin)

Adding to an existing codebase

Flow

"Opt in" by adding a flow declaration at the top of the file

```
// @flow
```

TypeScript

TS is a superset of JS (so any valid JS file is also a valid TS file)

But you do have to change the file extension to .ts

Configuration

Flow

TypeScript

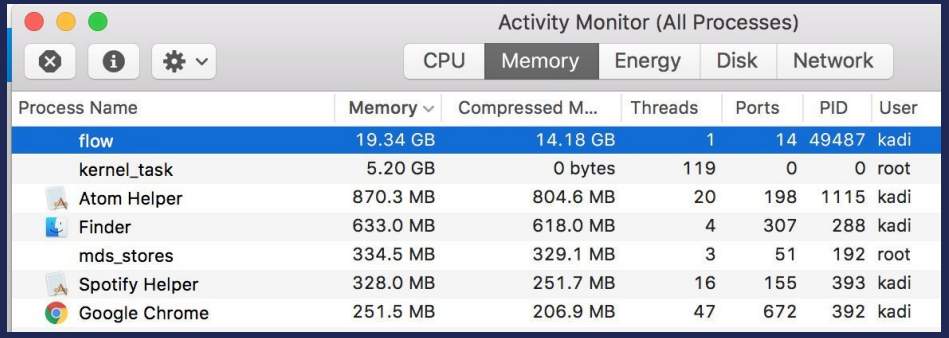
tsconfig.json

```jsonc
{
  "compilerOptions": {
    "allowJs": true,
    "allowSyntheticDefaultImports": true,
    "esModuleInterop": true,
    "isolatedModules": true,
    "jsx": "react-native",
    "lib": ["es6"],
    "moduleResolution": "node",
    "noEmit": true,
    "strict": true,
    "target": "esnext"
  },
  "exclude": ["node_modules"]
}
```

Performance

Flow

Usually fast, but notoriously unstable



Process Name	Memory	Compressed M...	Threads	Ports	PID	User
flow	19.34 GB	14.18 GB	1	14	49487	kadi
kernel_task	5.20 GB	0 bytes	119	0	0	root
Atom Helper	870.3 MB	804.6 MB	20	198	1115	kadi
Finder	633.0 MB	618.0 MB	4	307	288	kadi
mds_stores	334.5 MB	329.1 MB	3	51	192	root
Spotify Helper	328.0 MB	251.7 MB	16	155	393	kadi
Google Chrome	251.5 MB	206.9 MB	47	672	392	kadi

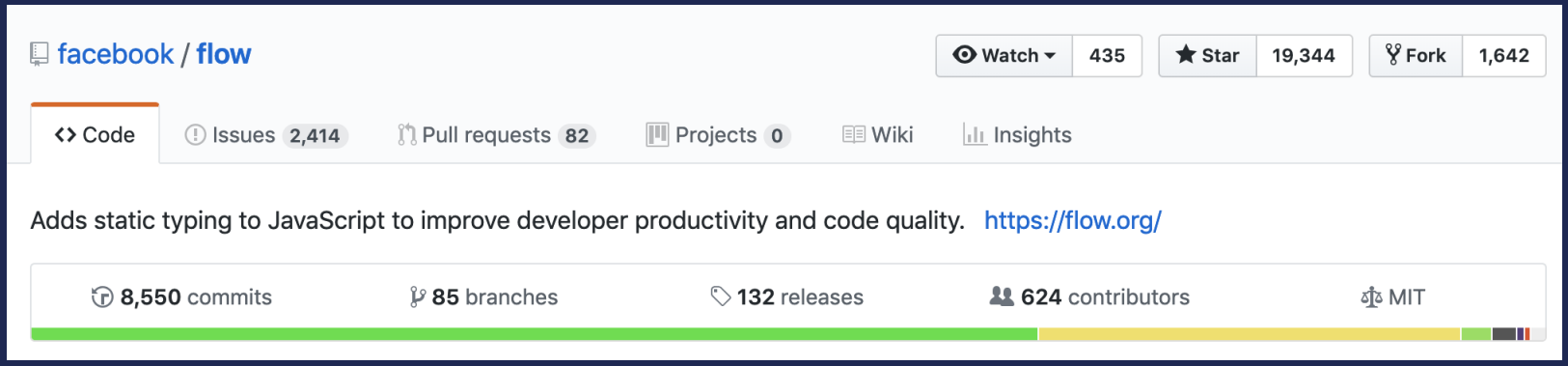
😱

TypeScript

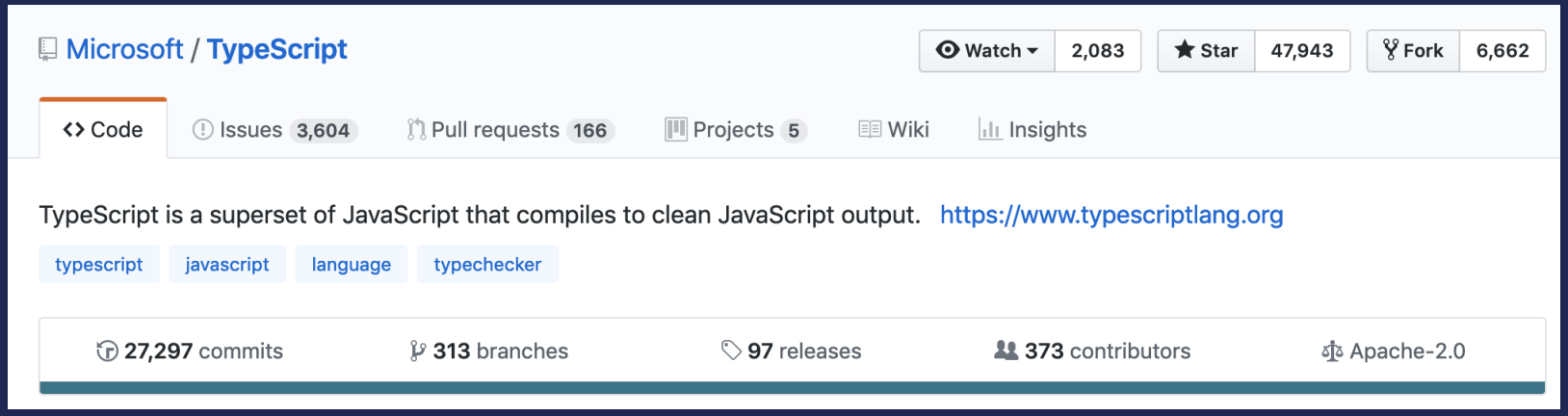
Slow to recompile on large projects

Community

Flow (by Facebook, 2014)

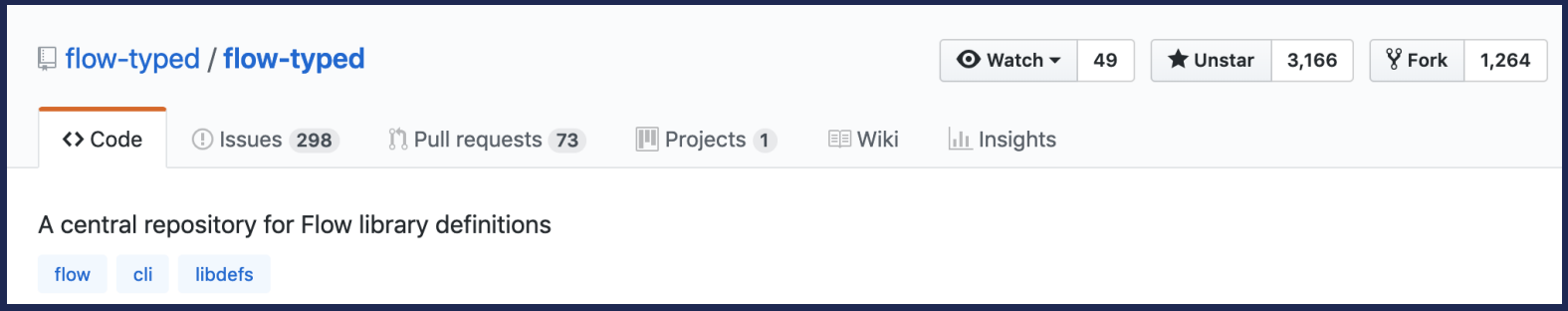


TypeScript (by Microsoft, 2012)

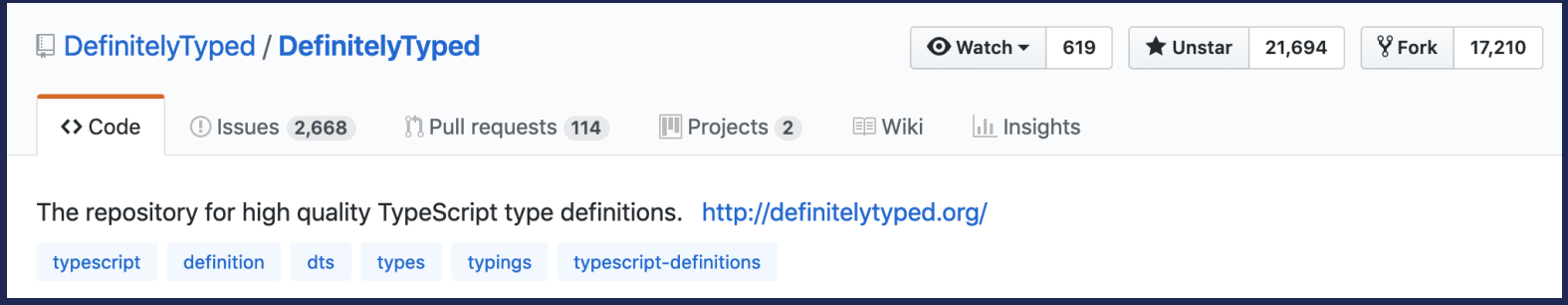


Typing Node Modules

Flow



TypeScript



So why are a lot of projects moving to TypeScript?

Larger community

Faster release cycle

More reliable

The features that made flow "better" have been implemented in TypeScript

What you should know before jumping on the TypeScript "Type Train"

For best results, use **VSCode**

tslint (the TypeSctipt linter) will be deprecated in 2019
So use **typescript-eslint**

Be prepared for a lot of Object Oriented influence

TypeScript is a **compiled language**, not a static type-checker - if you have type errors in your code, **it will not compile**

"Why not just use a proper statically typed language?"

Elm (2012 by Evan Czaplicki)

Compiled, statically typed, type declarations are optional, purely functional.

ReasonML (2016 by Jordan Walke at Facebook)

Transpiles to OCaml which compiles to JS, statically typed

Thank You 🤗

All Aboard the Type Train

by Kadi Kraman (@kadikraman)