Decorators :: TypeScript

May 26, 2023

Contents

1 Higher-order function decorators

The decorators proposal is about decorating class-based code (methods, properties, constructors, etc.). Still, we don't necessarily need any special syntax to decorate functions. ECMAScript, supporting higher-order functions, allows functions to be decorated (enhanced, endowed with more features or super cow powers) using ECMAScript features that exist since its inception.

1.1 Log high-order function decorator

```
const log = console.log.bind(console);
```

We can use it like this:

```
log(1);
//=> 1
```

And here's an "old school" (but not worse or wrong, just a battle-tested approach that works well) way to decorate our ‘log‘ and endow it with the ability of prepending date information date to the logged message:

```
function withDate(logFn) {
  return function logWithDate(msg) {
    logFn.call(null, new Date().toString());
    logFn.call(null, msg);
  };
}
```

And we decorate (enhance, endow with extra powers) our ‘log‘ function and use it like this:

```
const logger = withDate(log);

logger('IT FUCKING WORKS!');
//=> 5/25/2023, 8:57:53 AM: IT FUCKING WORKS!
```