Demystifying TypeScript Decorators

Overview

Decorators

Declarative v Imperative

Declarative Programming

Say what you want

Imperative Programming

Say how to get what you want

Angular 2

```
(function(app) {
  app.AppComponent =
    ng.core.Component({
      selector: 'my-app',
      template: '<h1>My First Angular 2 App</h1>'
    })
    .Class({
      constructor: function() {}
    });
})(window.app || (window.app = {}));
```

```
import { Component } from '@angular/core';

@Component({
  selector: 'my-app',
  template: '<h1>My First Angular 2 App</h1>'
})
export class AppComponent { }
```

Browser Support

Do decorators work in all browsers?

No Native Support

Object.defineProperty

+

Object.getOwnPropertyDescriptor

Transpilers

TypeScript and Babel

What is TypeScript?

3

#1 Superset of JavaScript

#2 Optional Static Typing

#3 Features from the Future

ES6, ES7, ES++

Use the Editor of Your Choice

Demos

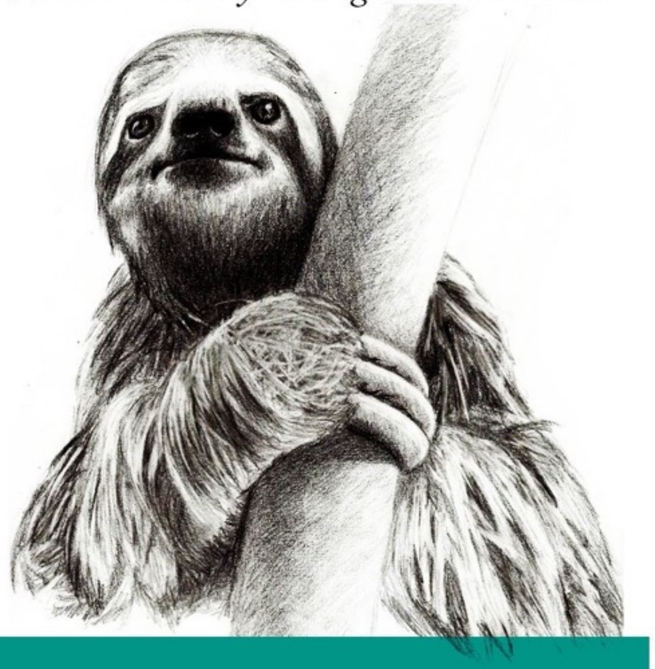
Simple Arbitrary App

```
export default class MyClass {
  prop1: string;
  prop2: number;

  myMethod(arg1: string, arg2: number) {
    // do something else arbitrary
  }
}
```

```
export default class MyAwesomeClass {
  betterProp1: string;
  evenBetterProp2: number;

  mySuperMethod(arg1: string, arg2: number) {
    // do something else arbitrary
  }
}
```

Books and Ratings

Cutting corners to meet arbitrary management deadlines



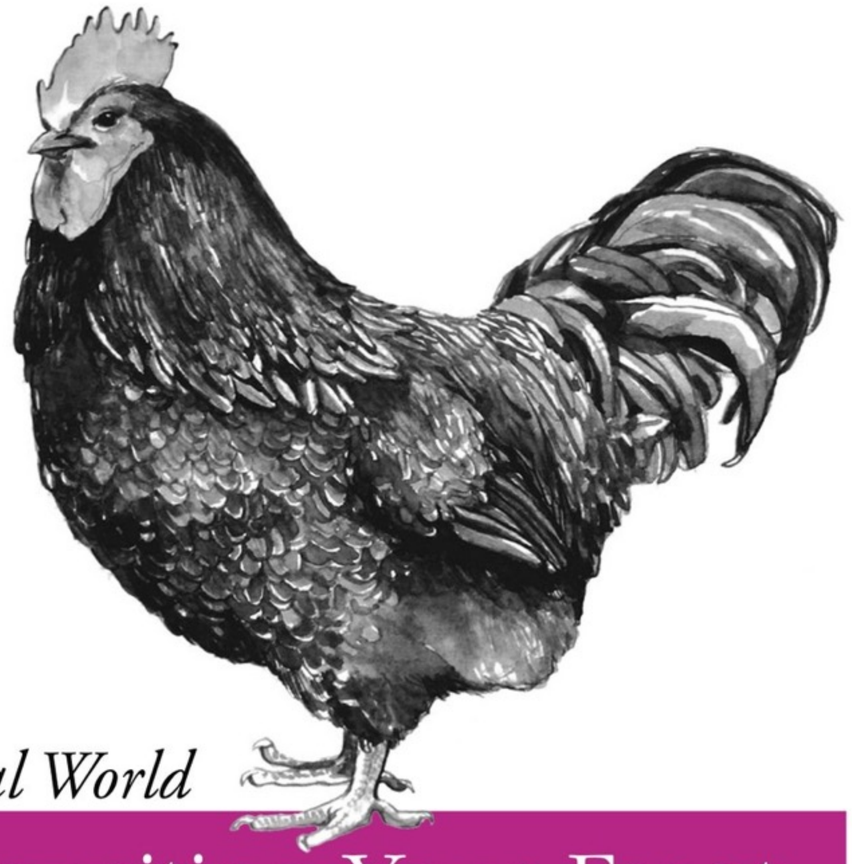
Essential

Copying and Pasting from Stack Overflow

O'REILLY®

The Practical Developer
@ThePracticalDev

This time you have definitely chosen the right libraries and build tools



Real World

Rewriting Your Front End Every Six Weeks

O RLY?

@ThePracticalDev

The internet will make those bad words go away

Essential

Googling the Error Message

O RLY?

The Practical Developer
@ThePracticalDev

Log Decorator Demo

Combining Decorators Demo

Wrap Up

Decorator Libraries

Express Decorators

```
@web.controller('/hello')
public class TestController {
  constructor(target) {
    this.target = target;
  }

  @web.get('/world')
  @web.middleware(myMiddlewareFunction)
  async sayHelloAction(request, response) {
    response.send(`hello, ${this.target}`);
  }

  @web.use
  async otherMiddleware(request, response, next) {
    // this will get called for every action
  }
}

let app = express();
let test = new TestController('world');
test.register(app);
```

lodash Decorators

```
class Person {
  constructor(firstName, lastName) {
    this.firstName = firstName;
    this.lastName = lastName;
  }

  @after(3)
  @debounce(100)
  getFullName() {
    return `${this.firstName} ${this.lastName}`
  }

  @curry(2)
  @memoize()
  doSomeHeavyProcessing(arg1, arg2) {
  }
}
```

React Decorators

```
@connectToStores([FooStore, BarStore], (context, props) => ({
    foo: context.getStore(FooStore).getFoo(),
    bar: context.getStore(BarStore).getBar()
}))
class Component extends React.Component {
   render() {
        return (
            <ul>
                <li>{this.props.foo}</li>
                <li>{this.props.bar}</li>
            </ul>
        );
    }
}

export default Component;
```

Word of Caution


STOP

The Future

TypeScript Roadmap

https://github.com/Microsoft/TypeScript/wiki/Roadmap

Decorators Proposal

https://github.com/wycats/javascript-decorators

Thanks!

James Churchill
Twitter: @SmashDev
GitHub: smashdevcode

https://github.com/smashdevcode/nodepdx-demystifying-typescript-decorators