Typing Octane

Using TypeScript with Ember 3.15+

James C. Davis

Front-end Lead, Center for Open Science

@jamescdavis @jamscdavis @jamescdavis

Yarn 2

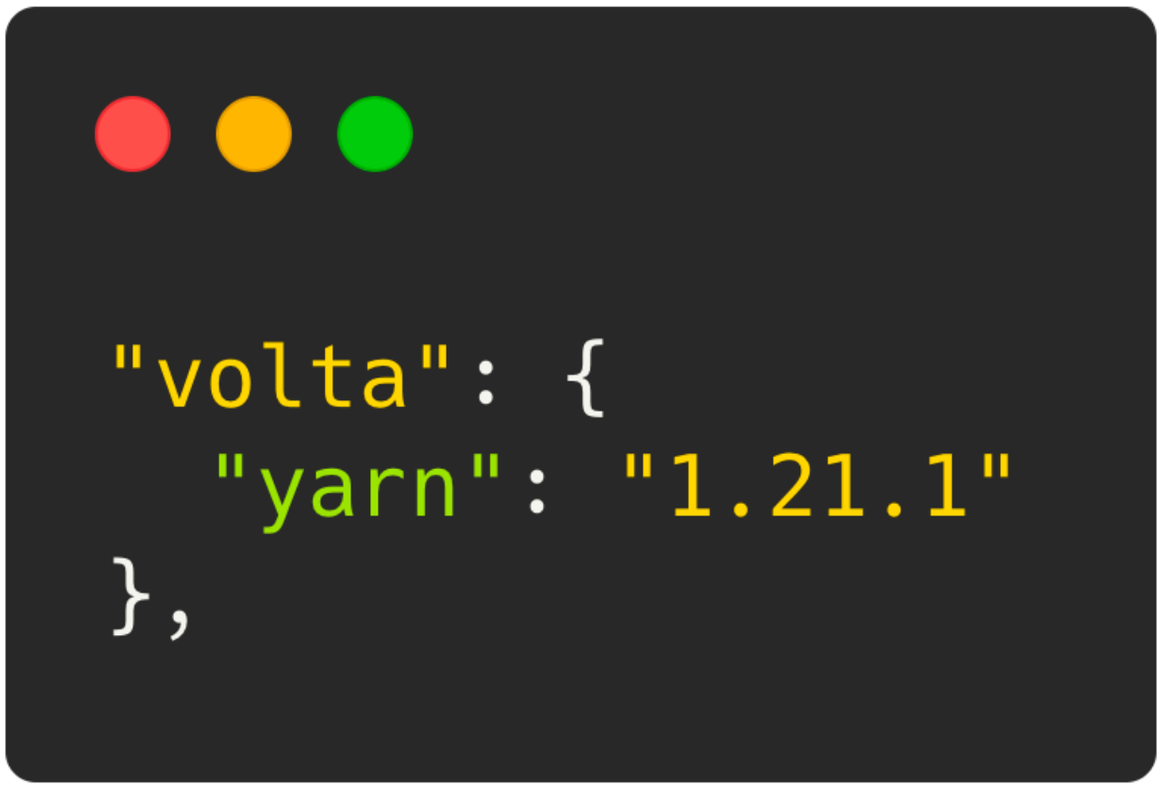
@jamescdavis @jamscdavis @jamescdavis

~~Yarn 2~~

@jamescdavis @jamscdavis @jamescdavis

Volta!

```
$ volta pin yarn@1
```



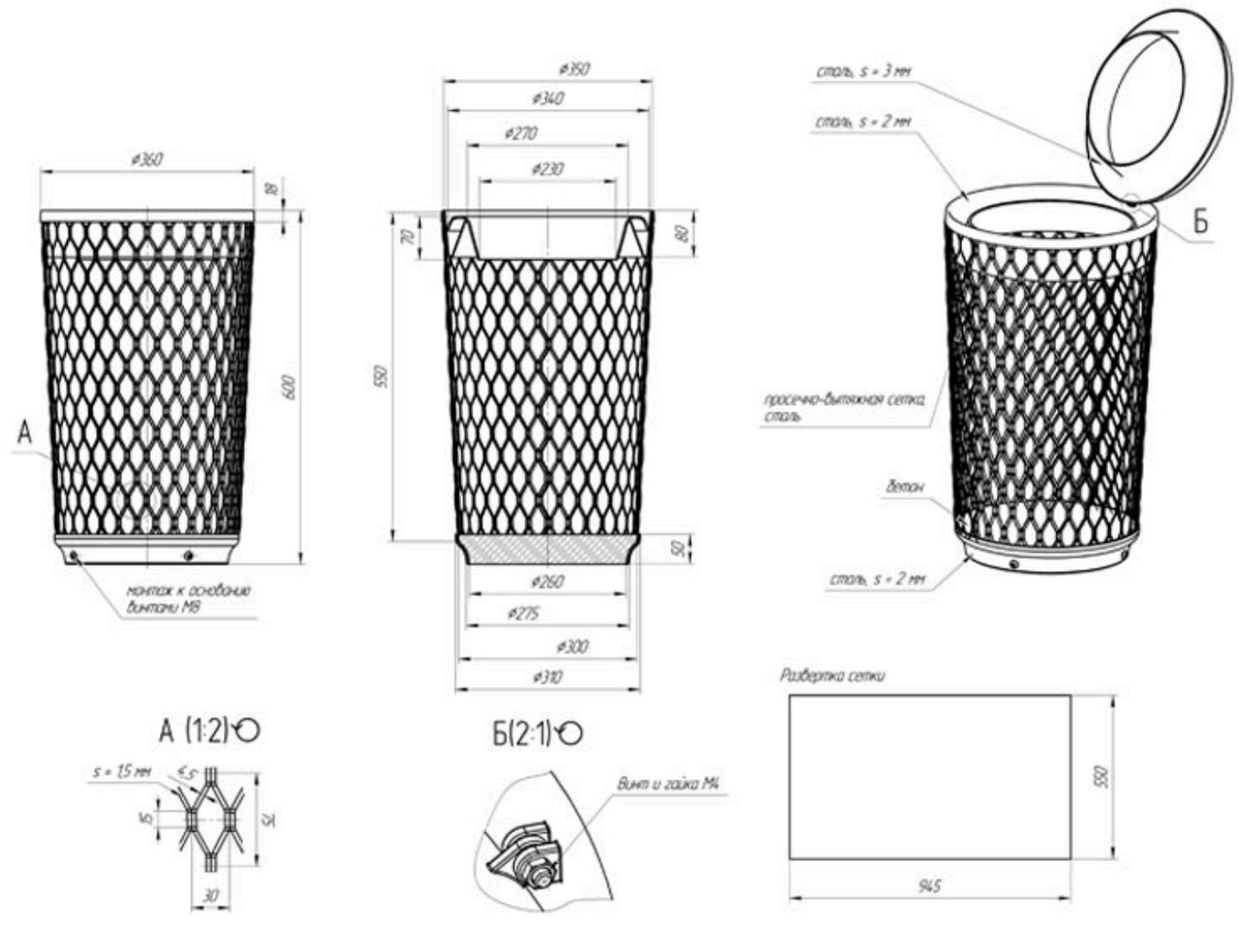
https://volta.sh

@jamescdavis @jamscdavis @jamescdavis

Install ember-cli-typescript

```
$ ember install ember-cli-typescript
```

Throw out the blueprints!




```
$ yarn remove ember-cli-typescript-blueprints
```

```
$ npm uninstall ember-cli-typescript-blueprints
```

@jamescdavis @jamscdavis @jamescdavis

But how do I get TypeScript things?

@jamescdavis @jamscdavis @jamescdavis

But how do I get TypeScript things?

```
$ cd app/components/
$ mv my-comp.js my-comp.ts
```

Install @glimmer/tracking

```
import { tracked } from '@glimmer/tracking';
```

```
$ yarn install @glimmer/tracking
```

```
    "devDependencies": {
      "@ember/optional-features": "^1.1.0",
      "@glimmer/component": "^1.0.0",
+     "@glimmer/tracking": "^1.0.0",
      "@types/ember": "^3.1.1",
      "@types/ember-data": "^3.1.9",
      "@types/ember-qunit": "^3.4.7",
```

Components

```hbs
<button {{on "click" this.minus}}>&minus;</button>
{{this.count}}
<button {{on "click" this.plus}}>+</button>
```

```js
import Component from '@glimmer/component';
import { tracked } from '@glimmer/tracking';
import { action } from '@ember/object';

export default class Counter extends Component {
  @tracked count = 0;

  @action plus() {
    this.count += 1;
  }

  @action minus() {
    this.count -= 1;
  }
}
```

@jamescdavis @jamscdavis @jamescdavis

Components

```hbs
<button {{on "click" this.minus}}>&minus;</button>
{{this.count}}
<button {{on "click" this.plus}}>+</button>
```

```js
import Component from '@glimmer/component';
import { tracked } from '@glimmer/tracking';
import { action } from '@ember/object';

export default class Counter extends Component {
  @tracked count = 0;

  @action plus() {
    this.count = 'hello!';
  }

  @action minus() {
    this.count -= 1;
  }
}
```

@jamescdavis @jamscdavis @jamescdavis

Components

```hbs
<button {{on "click" this.minus}}>&minus;</button>
{{this.count}}
<button {{on "click" this.plus}}>+</button>
```

```ts
import Component from '@glimmer/component';
import { tracked } from '@glimmer/tracking';
import { action } from '@ember/object';

export default class Counter extends Component {
  @tracked count = 0;

  @action plus() {
    this.count = 'hello!';
  }

  @action minus() {
    this.count -= 1;
  }
}
```

```
Type '"hello!"' is not assignable to type 'number'.
```

@jamescdavis @jamscdavis @jamescdavis

Component args

```
<button {{on "click" this.logArgs}}>Log 'em!</button>
```

```
import Component from '@glimmer/component';
import { action } from '@ember/object';

export default class ArgsDisplay extends Component {
  @action
  logArgs() {
    console.log(this.args);
  }
}
```

Component args

```
<button {{on "click" this.logArgs}}>Log 'em!</button>
```

```
import Component from '@glimmer/component';
import { action } from '@ember/object';

export default class ArgsDisplay extends Component {
  @action
  logArgs() {
    console.log(this.args.foo);
  }
}
```

Component args

```
<button {{on "click" this.logArgs}}>Log 'em!</button>
```

```
import Component from '@glimmer/component';
import { action } from '@ember/object';

export default class ArgsDisplay extends Component {
  @action
  logArgs() {
    console.log(this.args.foo);
  }
}
```

```
Property 'foo' does not exist on type '{}'.
```

Typing args

```ts
import Component from '@glimmer/component';
import { action } from '@ember/object';

export default class ArgsDisplay extends Component {
  args: { foo: string };

  @action
  logArgs() {
    console.log(this.args.foo);
  }
}
```

Typing args

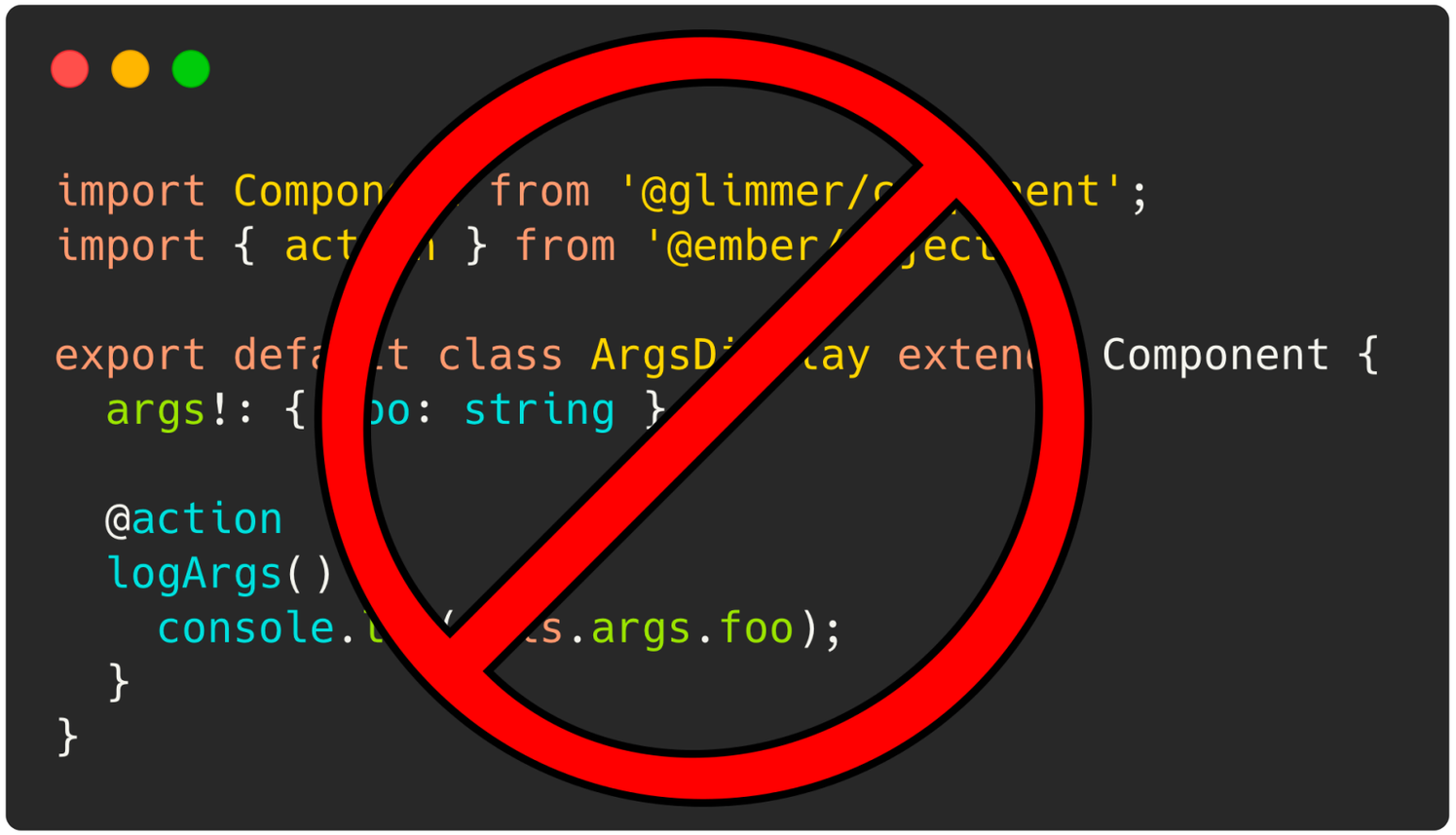
```ts
import Component from '@glimmer/component';
import { action } from '@ember/object';

export default class ArgsDisplay extends Component {
  args: { foo: string };

  @action
  logArgs() {
    console.log(this.args.foo);
  }
}
```

```
Property 'args' has no initializer and is
not definitely assigned in the constructor.
```

Typing args

```ts
import Component from '@glimmer/component';
import { action } from '@ember/object';

export default class ArgsDisplay extends Component {
  args!: { foo: string };

  @action
  logArgs() {
    console.log(this.args.foo);
  }
}
```

Typing args


import Compon from '@glimmer/c ent';
import { act n } from '@ember/ ject
export defa t class ArgsDi tay exten Component {
args!: { o: string }
@action
logArgs()
console.l (s.args.foo);
}
}

Typing args

```ts
import Component from '@glimmer/component';
import { action } from '@ember/object';

export default class ArgsDisplay extends Component {
  declare args: { foo: string };

  @action
  logArgs() {
    console.log(this.args.foo);
  }
}
```

@jamescdavis @jamscdavis @jamescdavis

Typing args

@jamescdavis @jamscdavis @jamescdavis

Constructor args

```
import Component from '@glimmer/component';

export default class ArgsDisplay extends Component {
  constructor(owner, args) {
    super(owner, args);

    console.log(args.foo);
  }
}
```

Constructor args

```ts
import Component from '@glimmer/component';

export default class ArgsDisplay extends Component {
  constructor(owner: unknown, args: { foo: string }) {
    super(owner, args);

    console.log(args.foo);
  }
}
```

Constructor args

```ts
import Component from '@glimmer/component';
import { action } from '@ember/object';

interface Args {
  foo: string;
}

export default class ArgsDisplay extends Component {
  args!: Args;

  constructor(owner: unknown, args: Args) {
    super(owner, args);

    console.log(args.foo);
  }

  @action
  logArgs() {
     console.log(this.args.foo);
  }
}
```

There’s a better way!

@jamescdavis @jamscdavis @jamescdavis

There’s a better way!

```ts
export default class Component<Args extends {} = {}> {
  args: Args;

  constructor(owner: unknown, args: Args);
}
```

@jamescdavis @jamscdavis @jamescdavis

There’s a better way!

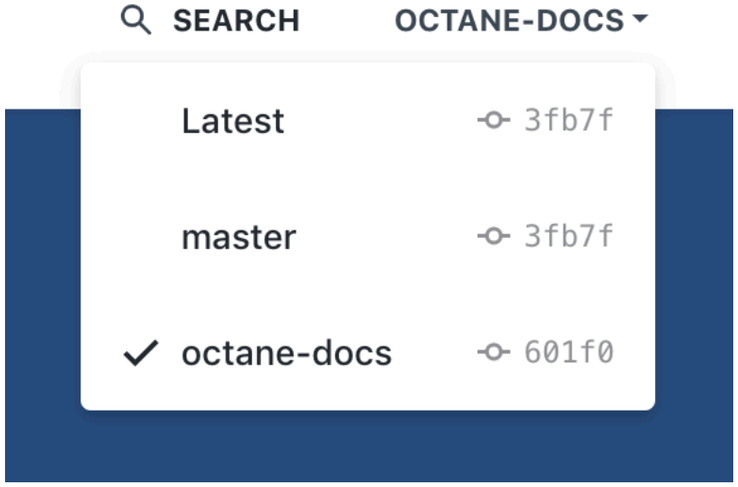
```ts
export default class Component<Args extends {} = {}> {
  args: Args;

  constructor(owner: unknown, args: Args);
}
```

```ts
import Component from '@glimmer/component';
import { action } from '@ember/object';

interface Args {
  foo: string;
}

export default class ArgsDisplay extends Component<Args> {
  constructor(owner: unknown, args: Args) {
    super(owner, args);

    console.log(args.foo);
  }

  @action
  logArgs() {
     console.log(this.args.foo);
  }
}
```

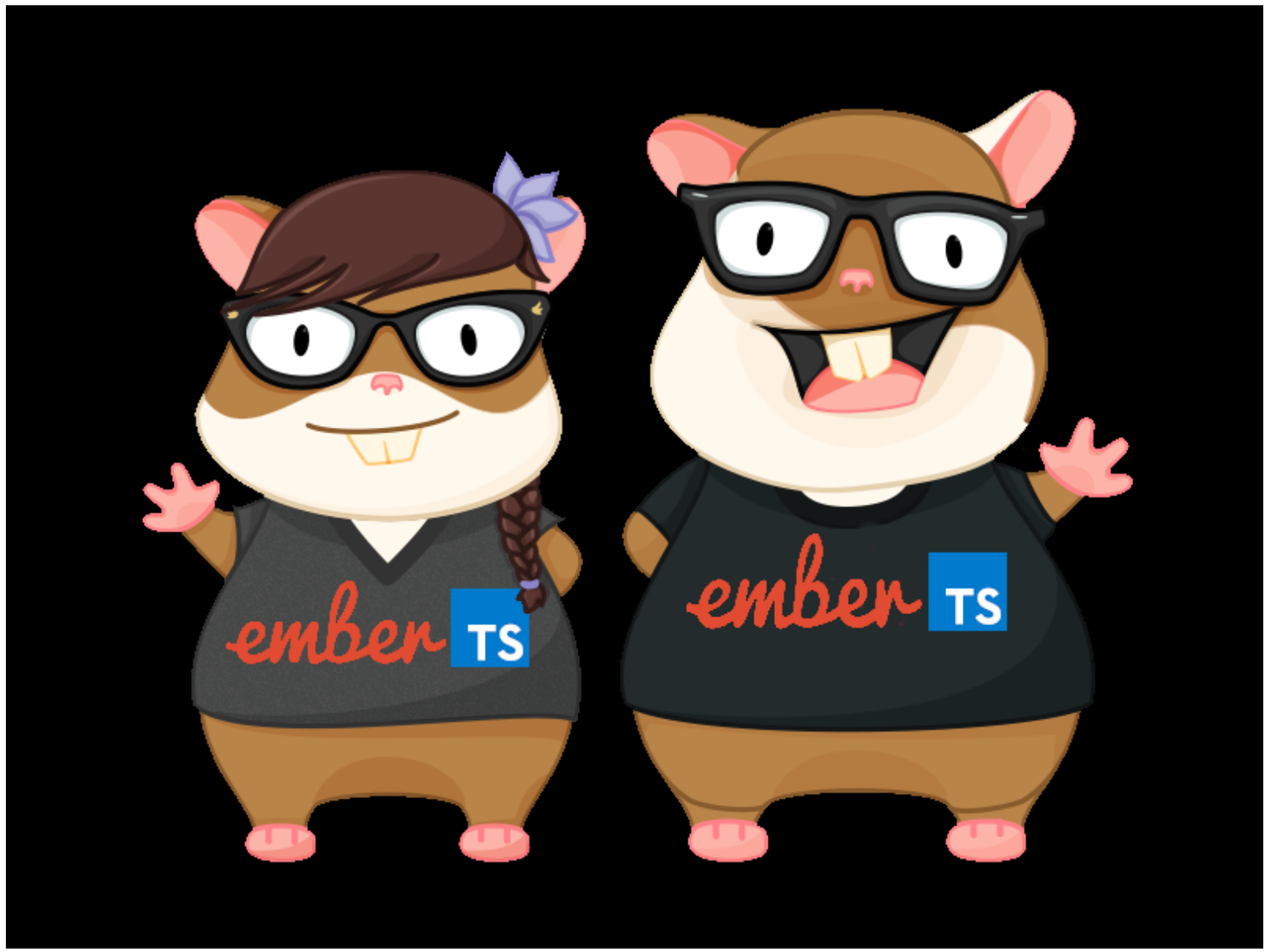
@jamescdavis @jamscdavis @jamescdavis

Octane guides are coming!

https://ember-cli-typescript.com/versions/octane-docs/



Thank you!



@jamescdavis @jamscdavis @jamescdavis