Gregory Shehet 🖖

@AGambit95

- Software Engineer / Project Lead @
- *Contributor in MobX* and Focal
- Programming languages (*JavaScript*, *ReasonML*, *Elm*, *PureScript*, *Scala*, *Haskell*,...)
- *Believes in the power of ✨FRP and Monads✨*

Deep Diving in *The Typings World*

Few *questions*? 

Start from

scratch 


28/08/2017

React.js — русскоговорящее сообщество

3815 members

v0.53.0

facebook-github-bot released this on Aug 16

This release includes major changes to Flow's model for React. The following links contain detailed documentation on the new model.

- Defining components
- Event handling
- ref functions
- Typing children
- Higher-order components
- Utility type reference

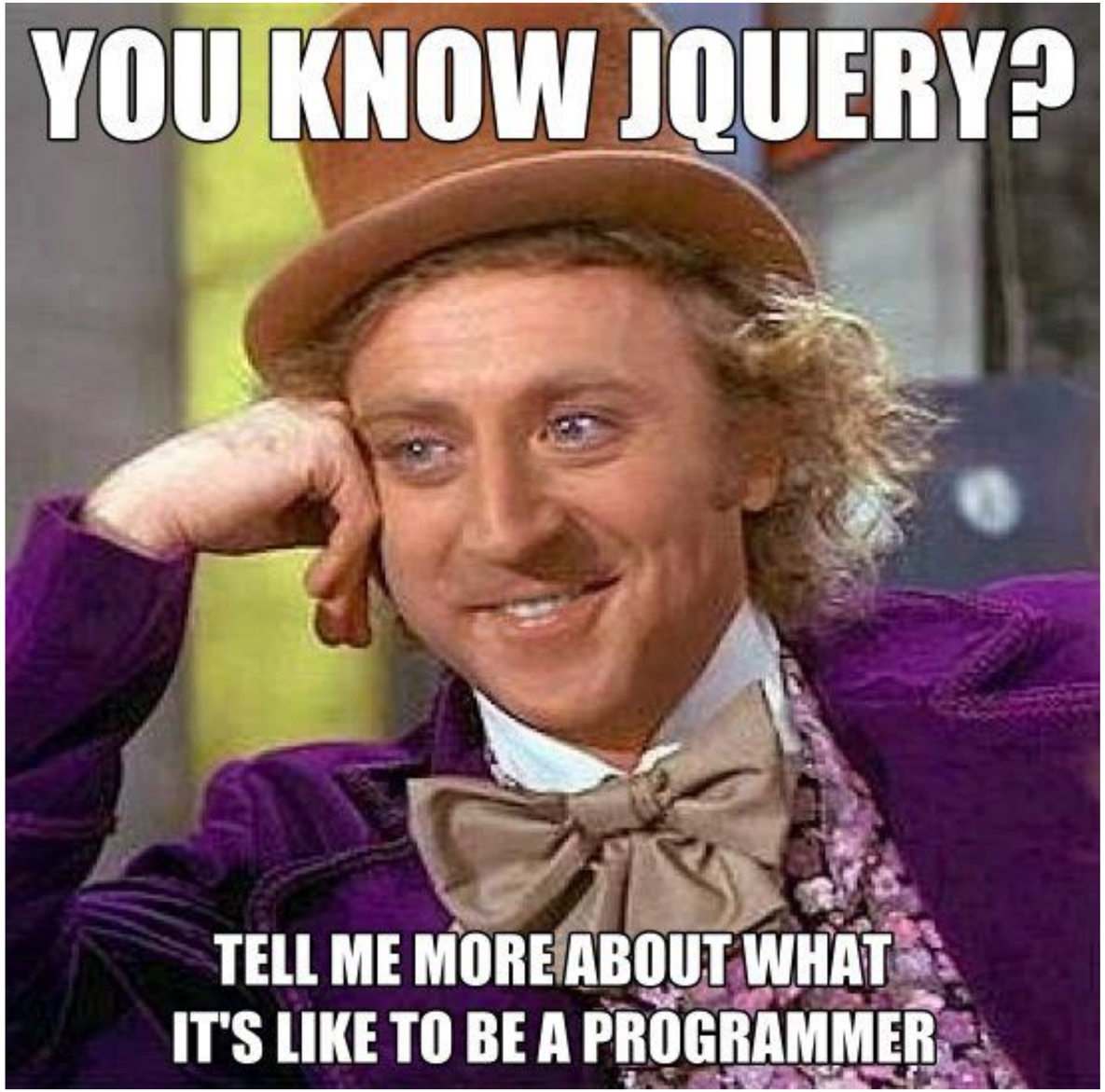
Не нужен!!!

4 years ago

2013

Ancient Times

ES6 New Standards

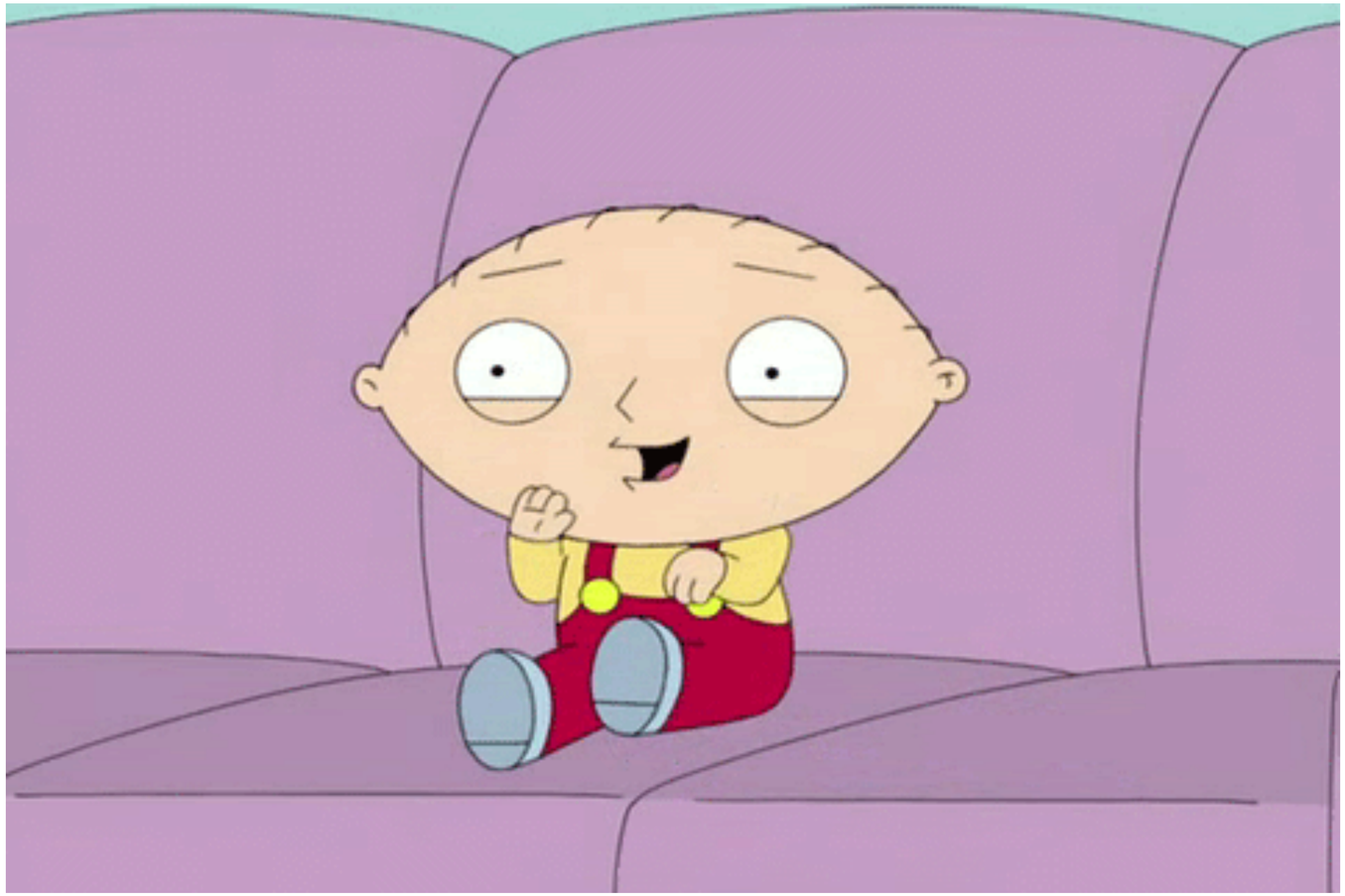

YOU KNOW JQUERY?
TELL ME MORE ABOUT WHAT
IT'S LIKE TO BE A PROGRAMMER

jQuery AGE

JS Development

Flexible

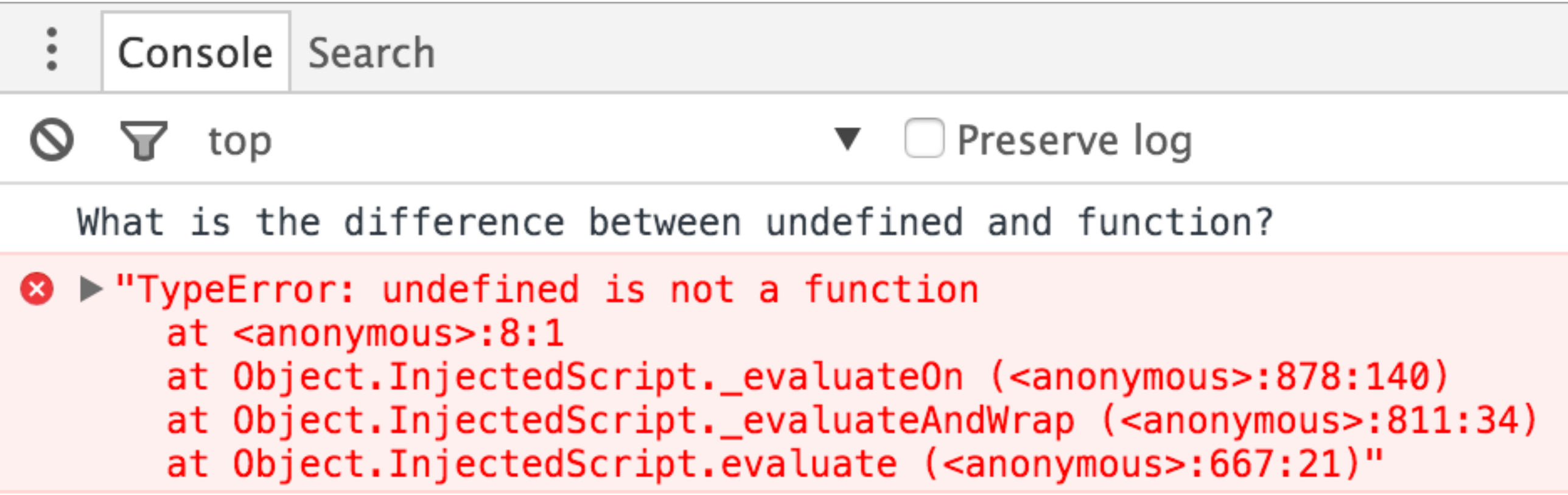
```
function foo() {
  var bar = 'some string'

  ...
  if (condition) {
    bar = 1
    ...
    bar += '100'
  }
  else {
    bar = {}
  }

  ...
  return bar
}
```



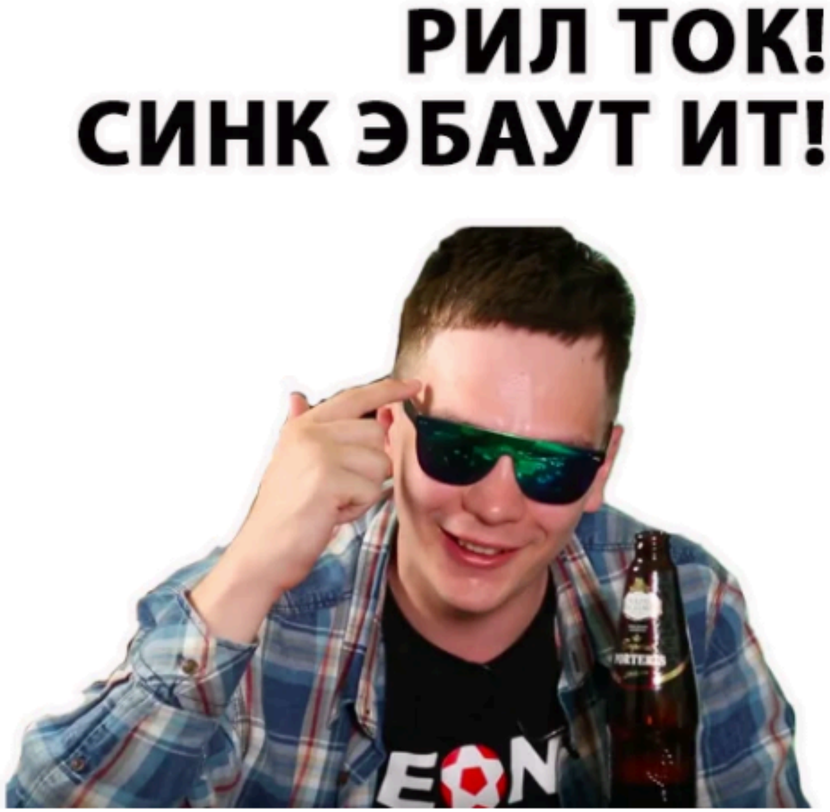
⭐JS is Awesome⭐

Few moments later...🤕

Don’t receive *Undefined* on Production 👌

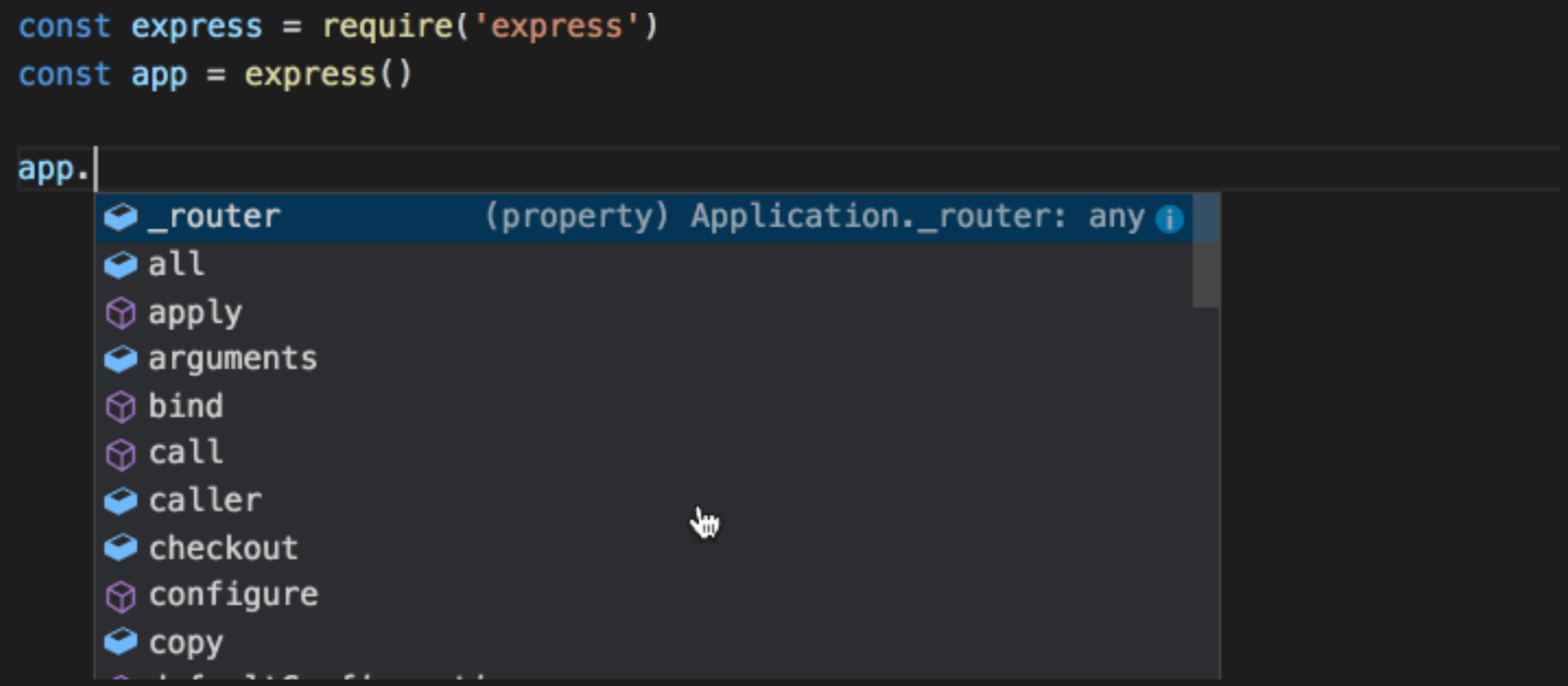


It is worth changing something

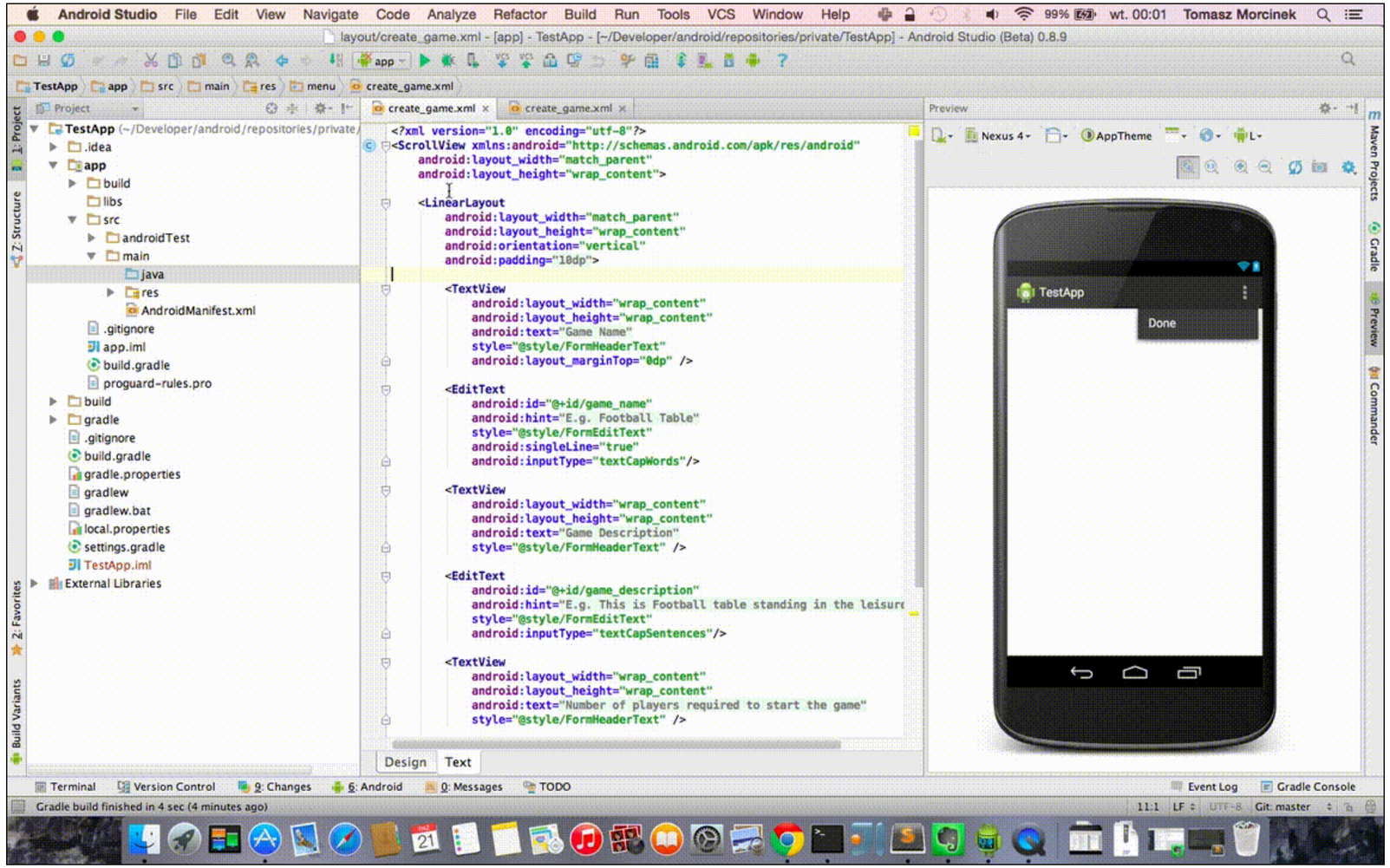


 Why type system is great? 

Autocomplete



Code Generation



IntelliJ IDEA Default Keymap

Editing

Ctrl + Space	Basic code completion (the name of any class, method or variable)
Ctrl + Shift + Space	Smart code completion (filters the list of methods and variables by expected type)
Ctrl + Shift + Enter	Complete statement
Ctrl + P	Parameter info (within method call arguments)
Ctrl + Q	Quick documentation lookup
Shift + F1	External Doc
Ctrl + mouse over code	Brief Info
Ctrl + F1	Show descriptions of error or warning at caret
Alt + Insert	Generate code... (Getters, Setters, Constructors, hashCode/equals, toString)
Ctrl + O	Override methods
Ctrl + I	Implement methods
Ctrl + Alt + T	Surround with... (if..else, try..catch, for, synchronized, etc.)
Ctrl + /	Comment/uncomment with line comment
Ctrl + Shift + /	Comment/uncomment with block comment
Ctrl + W	Select successively increasing code blocks
Ctrl + Shift + W	Decrease current selection to previous state
Alt + Q	Context info
Alt + Enter	Show intention actions and quick-fixes
Ctrl + Alt + L	Reformat code
Ctrl + Alt + O	Optimize imports
Ctrl + Alt + I	Auto-indent line(s)
Tab / Shift + Tab	Indent/unindent selected lines
Ctrl + X or Shift + Delete	Cut current line or selected block to clipboard
Ctrl + C or Ctrl + Insert	Copy current line or selected block to clipboard
Ctrl + V or Shift + Insert	Paste from clipboard
Ctrl + Shift + V	Paste from recent buffers...
Ctrl + D	Duplicate current line or selected block
Ctrl + Y	Delete line at caret
Ctrl + Shift + J	Smart line join
Ctrl + Enter	Smart line split
Shift + Enter	Start new line
Ctrl + Shift + U	Toggle case for word at caret or selected block
Ctrl + Shift +] / [	Select till code block end/start
Ctrl + Delete	Delete to word end
Ctrl + Backspace	Delete to word start
Ctrl + NumPad+/-	Expand/collapse code block
Ctrl + Shift + NumPad+	Expand all
Ctrl + Shift + NumPad-	Collapse all
Ctrl + F4	Close active editor tab

Search/Replace

Double Shift	Search everywhere
Ctrl + F	Find
F3	Find next
Shift + F3	Find previous
Ctrl + R	Replace
Ctrl + Shift + F	Find in path
Ctrl + Shift + R	Replace in path
Ctrl + Shift + S	Search structurally (Ultimate Edition only)
Ctrl + Shift + M	Replace structurally (Ultimate Edition only)

www.jetbrains.com/idea

IntelliJ IDEA Default Keymap

Usage Search

Alt + F7 / Ctrl + F7	Find usages / Find usages in file
Ctrl + Shift + F7	Highlight usages in file
Ctrl + Alt + F7	Show usages

Compile and Run

Ctrl + F9	Make project (compile modifed and dependent)
Ctrl + Shift + F9	Compile selected file, package or module
Alt + Shift + F10	Select configuration and run
Alt + Shift + F9	Select configuration and debug
Shift + F10	Run
Shift + F9	Debug
Ctrl + Shift + F10	Run context configuration from editor

Debugging

F8	Step over
F7	Step into
Shift + F7	Smart step into
Shift + F8	Step out
Alt + F9	Run to cursor
Alt + F8	Evaluate expression
F9	Resume program
Ctrl + F8	Toggle breakpoint
Ctrl + Shift + F8	View breakpoints

Navigation

Ctrl + N	Go to class
Ctrl + Shift + N	Go to file
Ctrl + Alt + Shift + N	Go to symbol
Alt + Right/Left	Go to next/previous editor tab
F12	Go back to previous tool window
Esc	Go to editor (from tool window)
Shift + Esc	Hide active or last active window
Ctrl + Shift + F4	Close active run/messages/find/... tab
Ctrl + G	Go to line
Ctrl + E	Recent files popup
Ctrl + Alt + Left/Right	Navigate back/forward
Ctrl + Shift + Backspace	Navigate to last edit location
Alt + F1	Select current file or symbol in any view
Ctrl + B or Ctrl + Click	Go to declaration
Ctrl + Alt + B	Go to implementation(s)
Ctrl + Shift + I	Open quick definition lookup
Ctrl + Shift + B	Go to type declaration
Ctrl + U	Go to super-method/super-class
Alt + Up/Down	Go to previous/next method
Ctrl +] / [	Move to code block end/start
Ctrl + F12	File structure popup
Ctrl + H	Type hierarchy
Ctrl + Shift + H	Method hierarchy
Ctrl + Alt + H	Call hierarchy
F2 / Shift + F2	Next/previous highlighted error
F4 / Ctrl + Enter	Edit source / View source
Alt + Home	Show navigation bar
F11	Toggle bookmark
Ctrl + F11	Toggle bookmark with mnemonic
Ctrl + #[0-9]	Go to numbered bookmark
Shift + F11	Show bookmarks

blogs.jetbrains.com/idea

IntelliJ IDEA Default Keymap

Refactoring

F5	Copy
F6	Move
Alt + Delete	Safe Delete
Shift + F6	Rename
Ctrl + F6	Change Signature
Ctrl + Alt + N	Inline
Ctrl + Alt + M	Extract Method
Ctrl + Alt + V	Extract Variable
Ctrl + Alt + F	Extract Field
Ctrl + Alt + C	Extract Constant
Ctrl + Alt + P	Extract Parameter

VCS/Local History

Ctrl + K	Commit project to VCS
Ctrl + T	Update project from VCS
Alt + Shift + C	View recent changes
Alt + BackQuote (`)	'VCS' quick popup

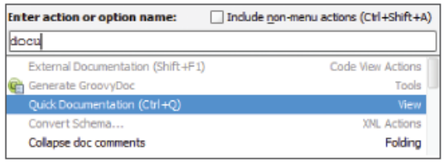
Live Templates

Ctrl + Alt + J	Surround with Live Template
Ctrl + J	Insert Live Template
iter	Iteration according to Java SDK 1.5 style
inst	Check object type with instanceof and downcast it
itco	Iterate elements of java.util.Collection
itit	Iterate elements of java.util.Iterator
itli	Iterate elements of java.util.List
psf	public static final
thr	throw new

General

Alt + #[0-9]	Open corresponding tool window
Ctrl + S	Save all
Ctrl + Alt + Y	Synchronize
Ctrl + Shift + F12	Toggle maximizing editor
Alt + Shift + F	Add to Favorites
Alt + Shift + I	Inspect current file with current profile
Ctrl + BackQuote (`)	Quick switch current scheme
Ctrl + Alt + S	Open Settings dialog
Ctrl + Alt + Shift + S	Open Project Structure dialog
Ctrl + Shift + A	Find Action
Ctrl + Tab	Switch between tabs and tool window

To find any action inside the IDE use Find Action (Ctrl+Shift+A/⇧ ⌘ A)



@intellijidea

JetBRAINS

Idea & Java Word


Alien: ???
Bo' Rai Cho : ???
Cassie Cage: ↓ ↓ ← → △
D'Vorah: → → R2
Ermac: ↓ ↑ ↓ ↓ ×
Erron Black: ↓ ↓ ↓ ○
Ferra/Torr: ← → ← ○
Goro: ↓ ↓ ↓ ↓ □
Jacqui Briggs: ↓ → ↓ ○
Jason: ↓ ← → →
Jax: ↓ → ↓ □
Johnny Cage: ↓ ← → R2
Kano: ↑ ↑ ← ×
Kenshi: ↓ → ↓ ×
Kitana: → ↓ ↓ ×
Kotal Khan: → ↓ ← □
Kung Lao: ↓ → ↓ ×
Leatherface: ???
Liu Kang: ↓ → ← ×
Mileena: ↓ ↓ ↓ □
Predator: → ↓ ← ↑
Quan Chi: ← → ↓ △
Raiden: ↓ ↓ ↓ △
Reptile: → ↓ ↓ R2
Scorpion: ↓ ↓ □
Shinnok: ↓ ← → →
Sonya Blade: ← → ↓ △
Sub-Zero: → ↓ ← △
Takeda: ↑ ↑ ↑ ↓
Tanya: ↑ ↓ ↑ ↓ ○
Tremor: ↓ ↓ ↓ ×
Triborg: ↓ ↓ ↑ R2
Fatality for Developer

Key Bindings for Visual Studio Code

Edit

Visual Studio Code lets you perform most tasks directly from the keyboard. This page lists out the default bindings (keyboard shortcuts) and describes how you can update them.

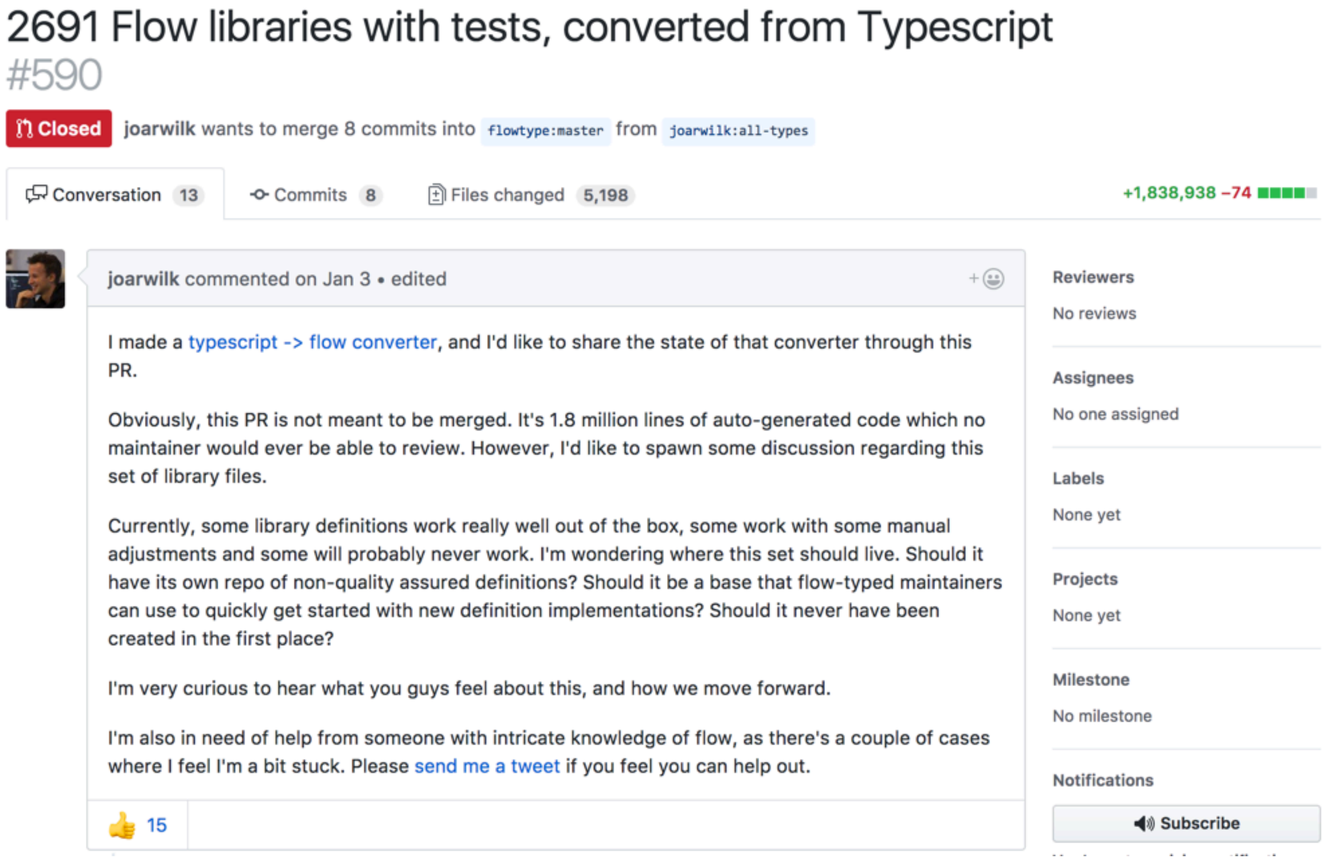
> **Note:** If you visit this page on a Mac, you will see the key bindings for the Mac. If you visit using Windows or Linux, you will see the keys for that platform. If you need the key binding for another platform, hover your mouse over the key you are interested in.

https://code.visualstudio.com/docs/getstarted/keybindings

Compiler Examples

1,8 millions of lines of auto-generated 😱

https://github.com/flowtype/flow-typed/pull/590



2691 Flow libraries with tests, converted from Typescript #590

Closed **joarwilk** wants to merge 8 commits into `flowtype:master` from `joarwilk:all-types`

Conversation 13 · Commits 8 · Files changed 5,198 · +1,838,938 −74

joarwilk commented on Jan 3 • edited

I made a typescript -> flow converter, and I'd like to share the state of that converter through this PR.

Obviously, this PR is not meant to be merged. It's 1.8 million lines of auto-generated code which no maintainer would ever be able to review. However, I'd like to spawn some discussion regarding this set of library files.

Currently, some library definitions work really well out of the box, some work with some manual adjustments and some will probably never work. I'm wondering where this set should live. Should it have its own repo of non-quality assured definitions? Should it be a base that flow-typed maintainers can use to quickly get started with new definition implementations? Should it never have been created in the first place?

I'm very curious to hear what you guys feel about this, and how we move forward.

I'm also in need of help from someone with intricate knowledge of flow, as there's a couple of cases where I feel I'm a bit stuck. Please send me a tweet if you feel you can help out.

👍 15

Reviewers
No reviews

Assignees
No one assigned

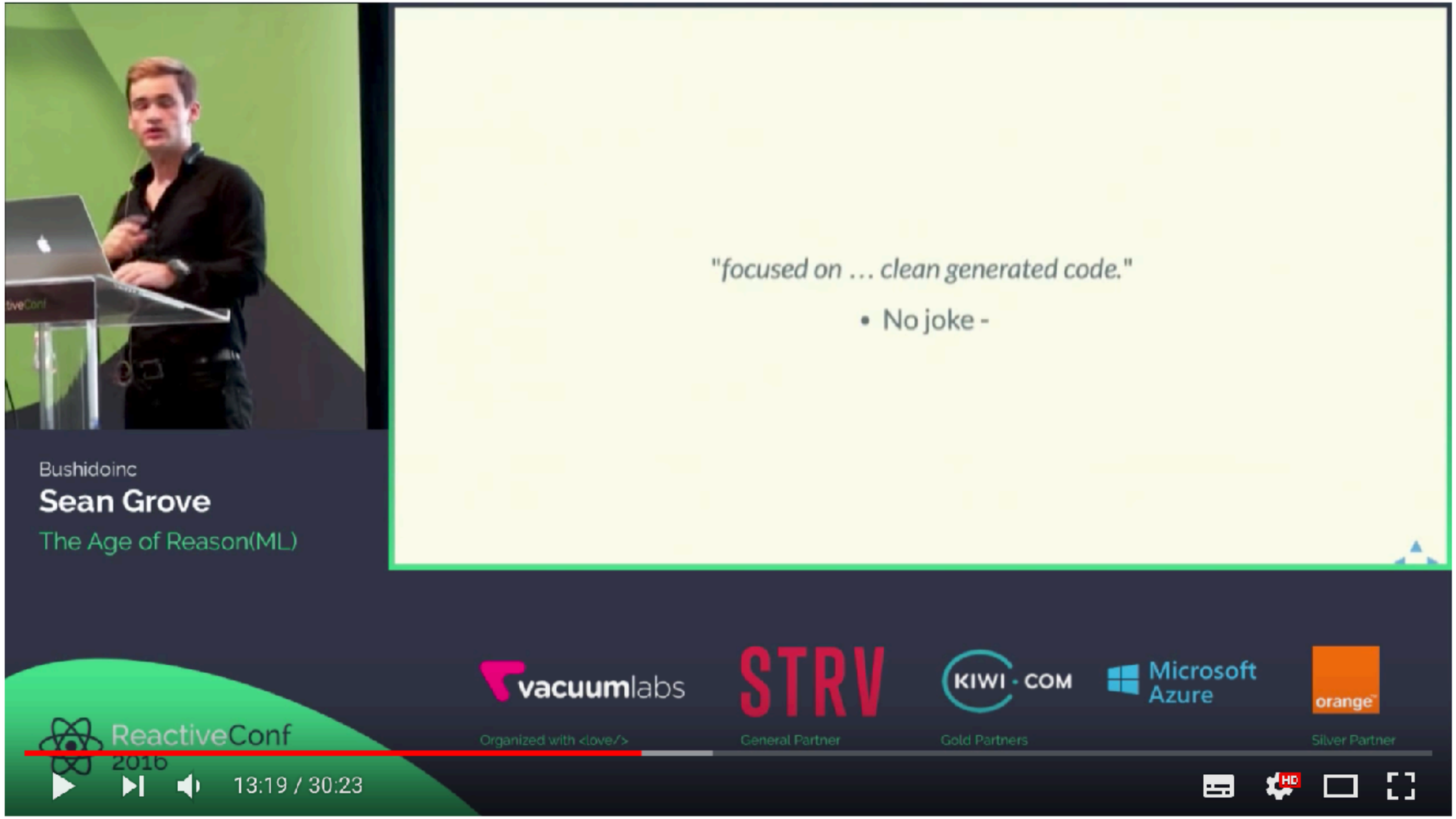
Labels
None yet

Projects
None yet

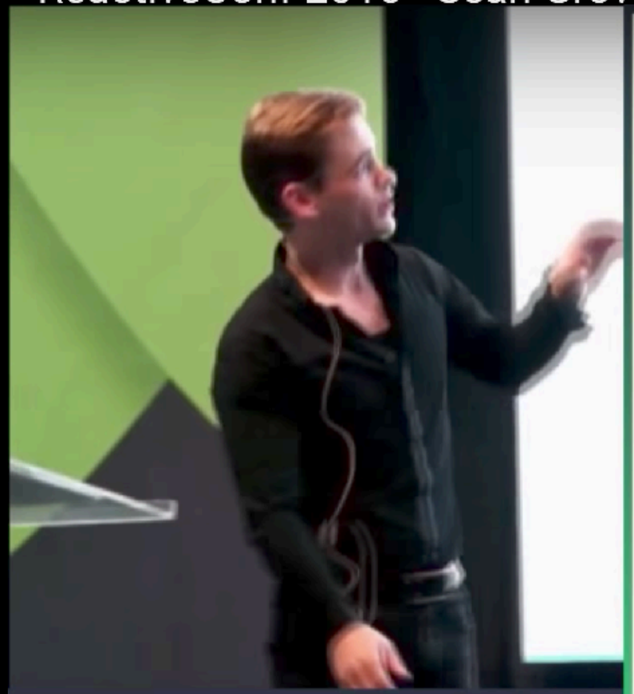
Milestone
No milestone

Notifications
Subscribe

One more thing...



https://youtu.be/8LCmLQ1-YqQ?t=13m19s



Bushidoinc

Sean Grove

The Age of Reason(ML)

4.3.1 COMPUTER OR HUMAN (1/2)?

```
1: "use strict";
2: var Int_map=require("./int_map.js");
3: function test() {
4:   var m = /* Empty */0;
5:   for(var i = 0; i <= 1000000; ++i){
6:     m = add(i, i, m);
7:   }
8:   for(var j = 0; j <= 1000000; ++j){
9:     find(j, m);
10:  }
11:  return /* () */0;
12: }
13: test(/* () */0);
```

Bushidoinc

Sean Grove

The Age of Reason(ML)

4.3.2 COMPUTER OR HUMAN (2/2)?

```
1: 'use strict';
2: var Pervasives = require("bs-platform/lib/js/pervasives");
3: var Http       = require("http");
4:
5: var hostname = "127.0.0.1";
6:
7: function create_server(http) {
8:   var server = http.createServer(function (_, resp) {
9:     resp.statusCode = 200;
10:    resp.setHeader("Content-Type", "text/plain");
11:    return resp.end("Hello world\n");
12:  });
13:  return server.listen(3000, hostname, function () {
14:    console.log("Server running at http://" +
15:    (hostname + (":" + (Pervasives.string_of_int(3000) + "/"))));
16:    return /* () */0;
17:  });
18: }
19:
20: create_server(Http);
```

Documentation & Support

```
function foo(
  error,
  list,
  dateObj,
  diff
) {
  ...
}
```

Documentation & Support

```
function foo(
  error,
  list,
  dateObj,
  diff
) {
  ...
}
```

Documentation & Support

```
function foo(
  error: Error | undefined,
  list: Array<Events>,
  dateObj: Date,
  diff: number | undefined
) {
  ...
}
```

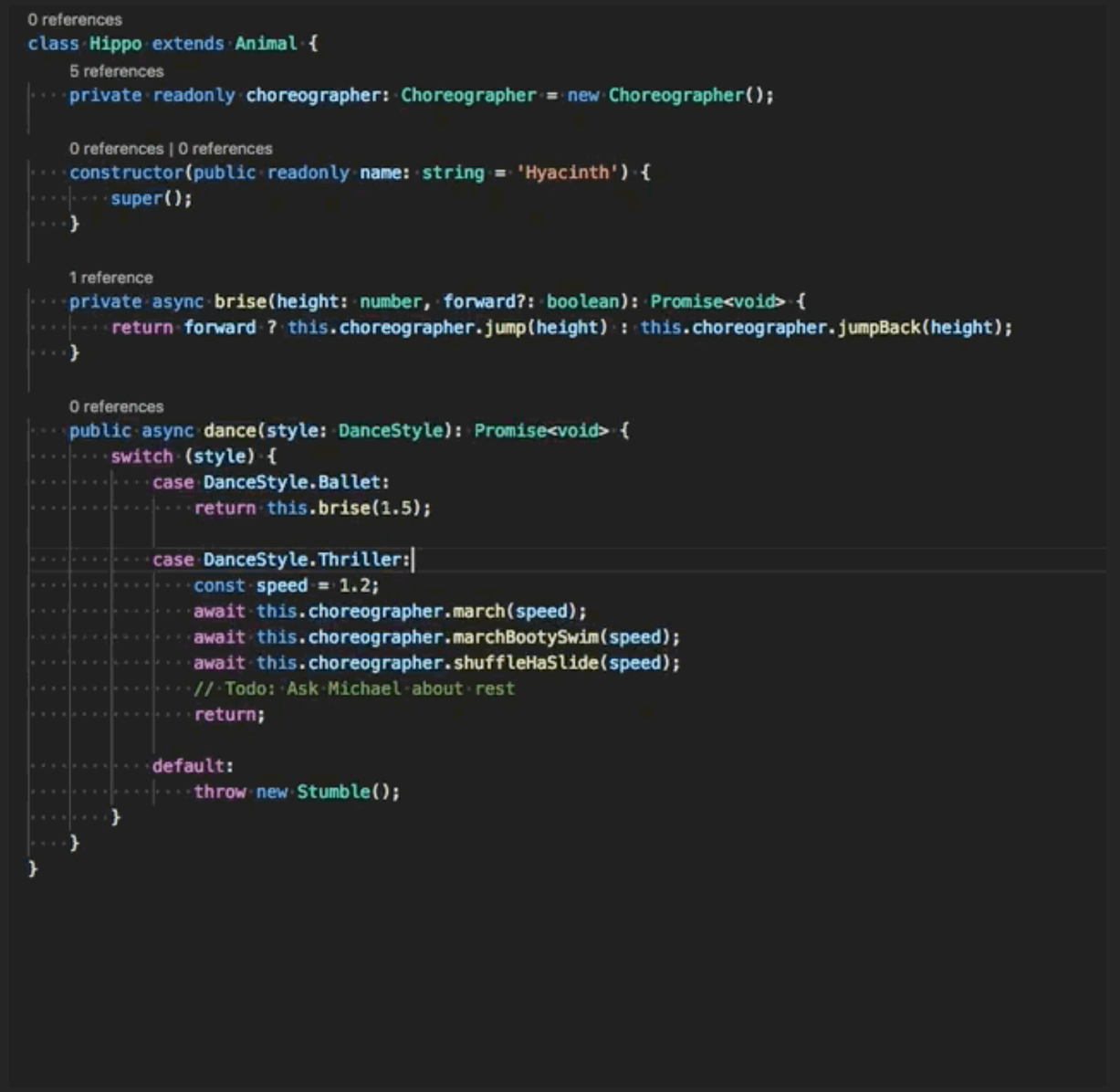
Interface & Types

```
interface Events {
   startDate: Date
   endDate: Dete
   summary: string
}
```

Documentation & Support

```
// Function foo ..
// @params ...
// @return ...
function foo(
  error: Error | undefined,
  list: Array<Events>,
  dateObj: Date,
  diff: number | undefined
) {
  ...
}
```

Refactoring



```
0 references
class Hippo extends Animal {
    5 references
    private readonly choreographer: Choreographer = new Choreographer();

    0 references | 0 references
    constructor(public readonly name: string = 'Hyacinth') {
        super();
    }

    1 reference
    private async brise(height: number, forward?: boolean): Promise<void> {
        return forward ? this.choreographer.jump(height) : this.choreographer.jumpBack(height);
    }

    0 references
    public async dance(style: DanceStyle): Promise<void> {
        switch (style) {
            case DanceStyle.Ballet:
                return this.brise(1.5);

            case DanceStyle.Thriller:
                const speed = 1.2;
                await this.choreographer.march(speed);
                await this.choreographer.marchBootySwim(speed);
                await this.choreographer.shuffleHaSlide(speed);
                // Todo: Ask Michael about rest
                return;

            default:
                throw new Stumble();
        }
    }
}
```

Code Design

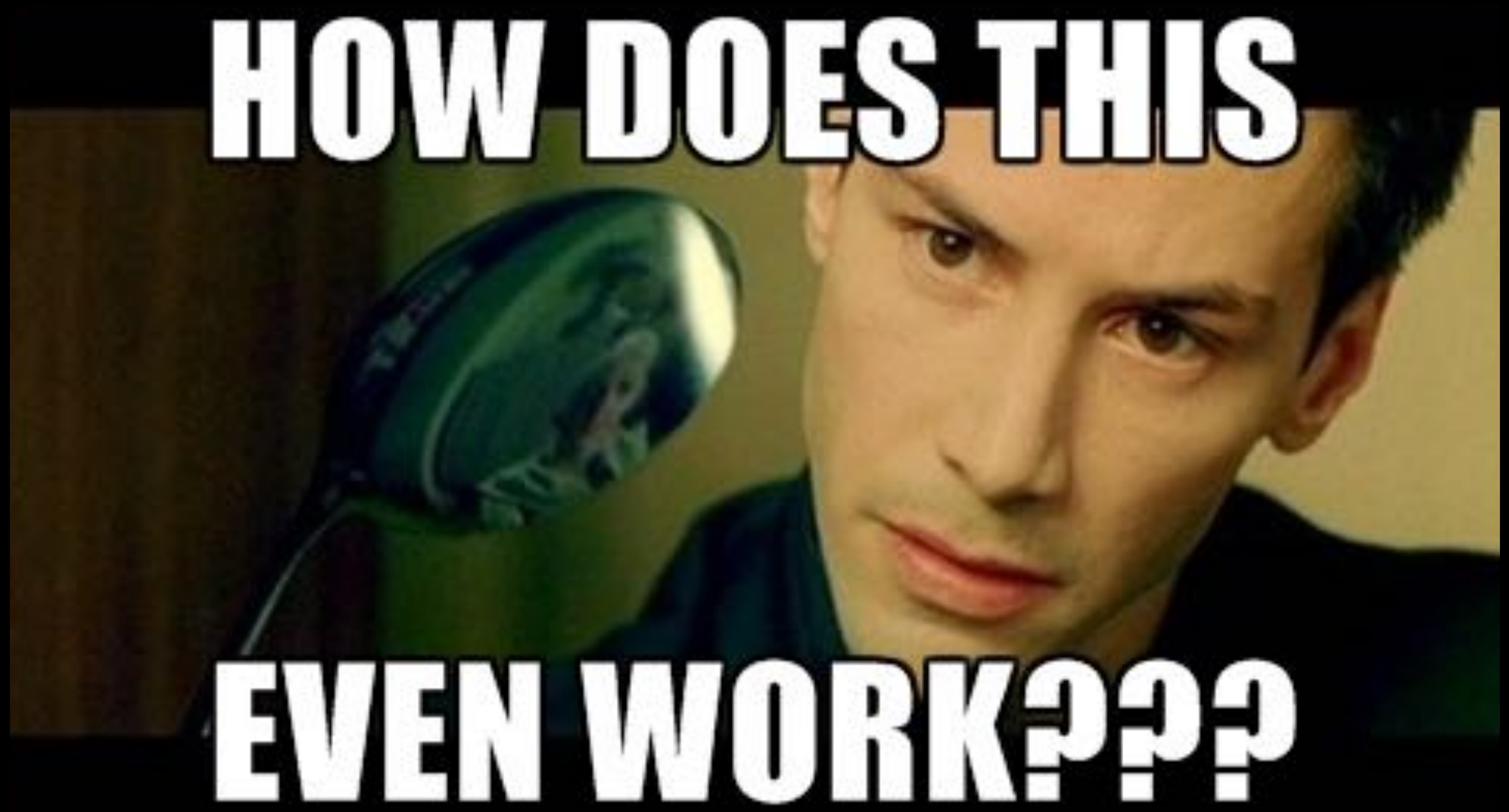
```
interface App {
  todos: Todos

  addTodo: (todo: Todo) => void
  removeTodo: (id: number) => void
  toggleTodo: (id: number) => void
}
```

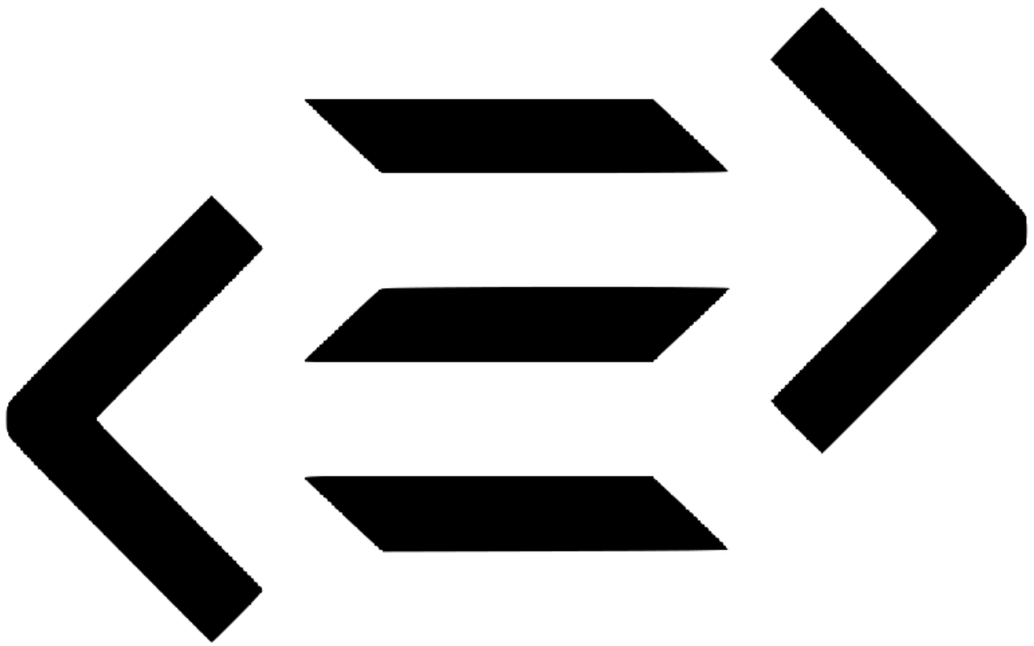
Ok. Typings is awesome.

How to use it?

ELM code

```
-- UPDATE

type Msg
  = ToggleNotifications
  | ToggleAutoplay


update : Msg -> Model -> Model
update msg model =
  case msg of
    ToggleNotifications ->
      { model | notifications = not model.notifications }

    ToggleAutoplay ->
      { model | autoplay = not model.autoplay }

-- VIEW

view : Model -> Html Msg
view model =
  fieldset []
    [ checkbox ToggleNotifications "Email Notifications"
    , checkbox ToggleAutoplay "Video Autoplay"
    ]


checkbox : msg -> String -> Html msg
checkbox msg name =
  label
    [ style [("padding", "20px")]
    ]
    [ input [ type_ "checkbox", onClick msg ] []
    , text name
    ]
```


HOW DOES THIS
EVEN WORK???



Good for business!

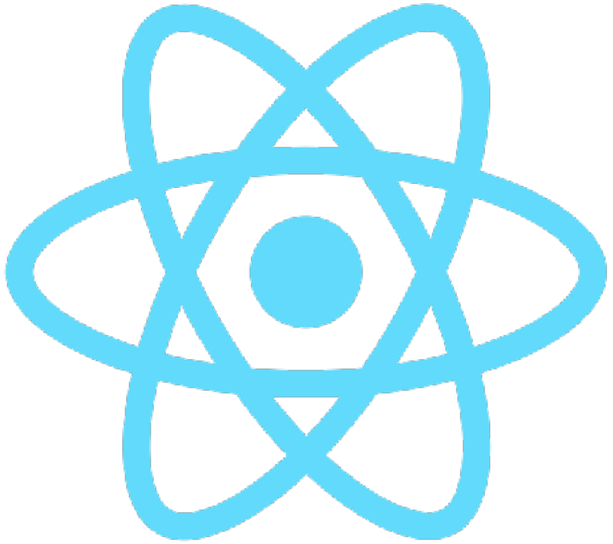
&

VS

What is better? 😨

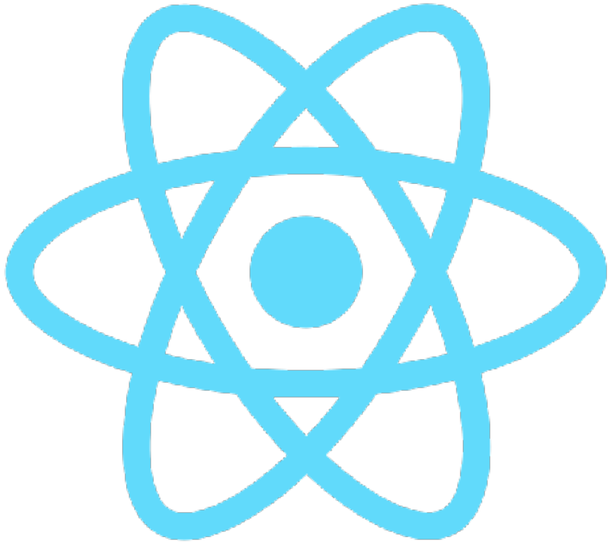
facebook VS Microsoft

What is better? 😨

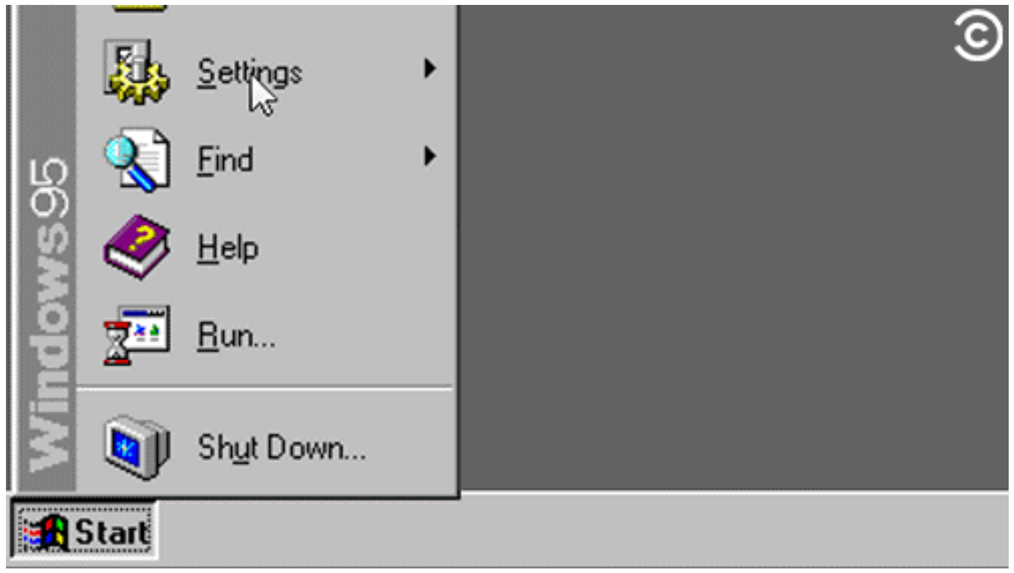


VS

What is better? 😨



VS



What is better? 😨

Ok Let's be Serious 

How use Flow?

How use TypeScript?

Let’s battle it out!


Flow
VERSUS
SLOVO
TypeScript

12 round


Winner 10
Loser 9

Round 1

Philosophy & Goals

Enforce type soundness / safety

https://flow.org/en/docs/lang/

Identify errors in programs through a balance between correctness and productivity

https://github.com/Microsoft/TypeScript/wiki/TypeScript-Design-Goals

Philosophy & Goals

Round 2

Ease of implementation

Flow + JS = 

```
// @flow
```

Just Add: `// @flow`

```
1  // @flow
2
3  function mult(a, b) {
4    return a * b; // Error!
5  }
6
7  mult('1', 2);
8
```

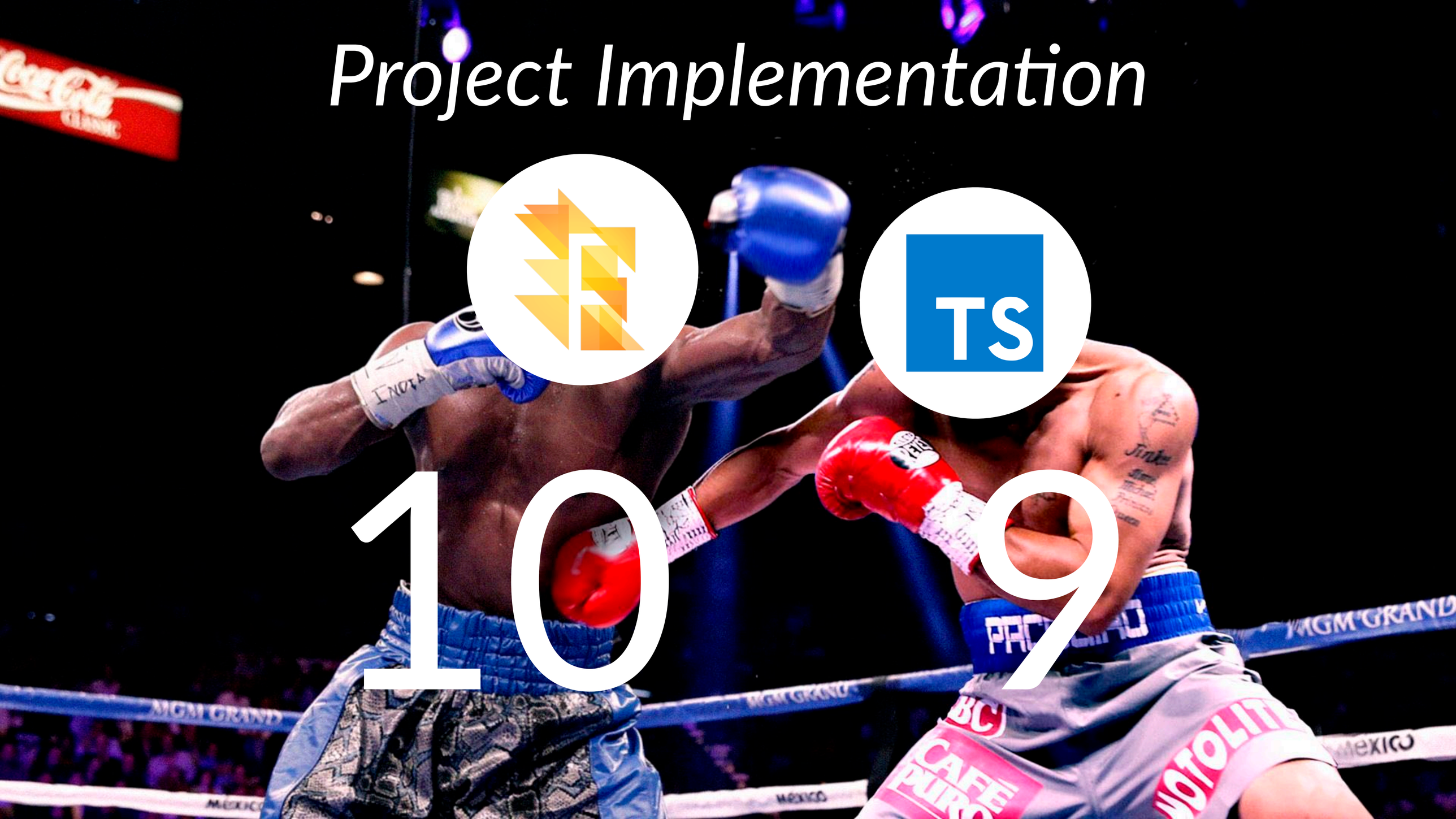
TS + JS = 

Typescript ;(

Compile JS with TS

TypeScript support ❤️

Project Implementation



Round 3

“Intelligence” of type-checking algorithm


OCaml

Example *getLength* Function

```
function getLength(x) {
  return x.length;
}

const total = getLength('Hello')
```

Add New call of *getLength* Function

```
function getLength(x) {
  return x.length;
}

const total = getLength('Hello') + getLength(null);
```

Successful Result 👍

```
function getLength(x) {
  return x.length;
}

const total = getLength('Hello') + getLength(null);
```

```
▶Uncaught TypeError: Cannot read property 'length' of null   VM18407:2
    at getLength (<anonymous>:2:16)
    at <anonymous>:1:36
```

Let's add
Type Checker

```flow
/* @flow */

function getLength(x) {
  return x.length;
}

const total = getLength('Hello') + getLength(null);

```

Flow

```ts
function getLength(x) {
  return x.length;
}

const total = getLength('Hello') + getLength(null);

```

TypeScript

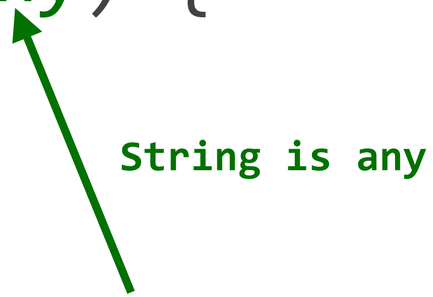
Why TypeScript don’t show an Error?

```
function getLength(x) {
  return x.length;
}
```

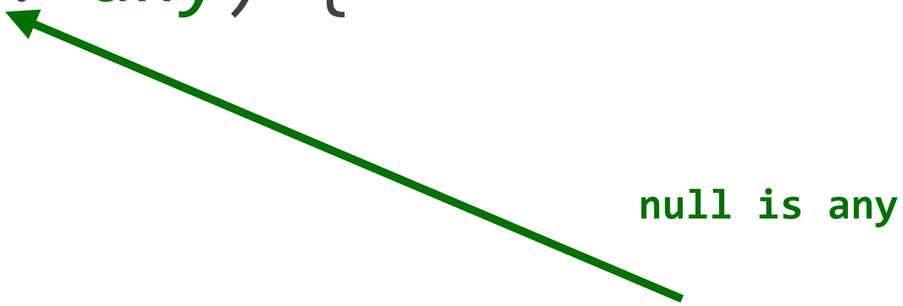
```
function getLength(x: any) {
  return x.length;
}
```

```
function getLength(x: any) {
  return x.length; // any has property length
}
```

```
function getLength(x: any) {
  return x.length;
}

const total = getLength('Hello') + getLength(null);
```



```
function getLength(x: any) {
  return x.length;
}

const total = getLength('Hello') + getLength(null);
```



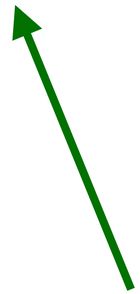
How Flow catch an Error?

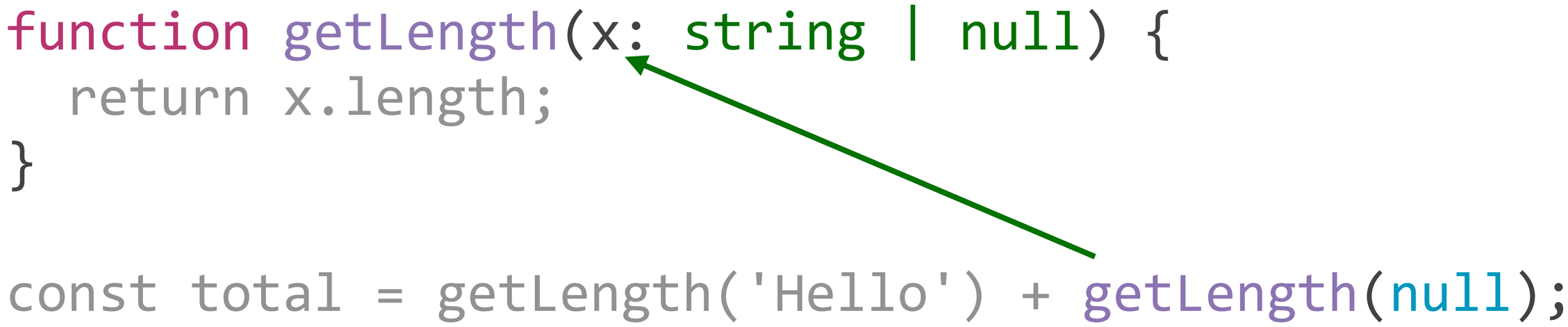
```
function getLength(x) {
  return x.length;
}
```

```
function getLength(x: any) {
  return x.length;
}
```

```
function getLength(x: any) {
  return x.length; // any has property length
}
```

```
function getLength(x: string) {
  return x.length;
}

const total = getLength('Hello') + getLength(null);
```




function getLength(x: string | null) {
return x.length;
}
const total = getLength('Hello') + getLength(null);

```
function getLength(x: string | null) {
  return x.length; // null doesn’t has property length
}

const total = getLength('Hello') + getLength(null);
```


“Intelligence” of type-checking algorithm
10
9
TS

Round 4

Type Safe

Soundness & Completeness

```
class Animal {}

class Cat extends Animal {
  meow() {}
}

class Tomato {}

export function test(animals: Animal[]) {
  animals.push(new Tomato());
}

const cats: Cat[] = [];

test(cats);

cats.forEach(cat => cat.meow()); // runtime error
```

```
class Animal {}

class Cat extends Animal {
  meow() {}
}

class Tomato {}

export function test(animals: Animal[]) {
  animals.push(new Tomato());
}

const cats: Cat[] = [];

test(cats);

cats.forEach(cat => cat.meow()); // runtime error
```

```
class Animal {}

class Cat extends Animal {
  meow() {}
}

class Tomato {}

export function test(animals: Animal[]) {
  animals.push(new Tomato());
}
```

```
const cats: Cat[] = [];

test(cats);

cats.forEach(cat => cat.meow()); // runtime error
```

TypeScript

```
1   class Animal {}
2
3   class Cat extends Animal {
4     meow() {}
5   }
6
7   class Tomato {}
8
9   export function test(animals: Animal[]) {
10    animals.push(new Tomato());
11  }
12
13  const cats: Cat[] = [];
14
15  test(cats);
16
17  cats.forEach(cat => cat.meow()); // runtime error
18
```

Flow

```
1   /* @flow */
2
3   class Animal {}
4
5   class Cat extends Animal {
6     meow() {}
7   }
8
9   class Tomato {}
10
11  export function test(animals: Animal[]) {
12    animals.push(new Tomato());
13  }
14
15  const cats: Cat[] = [];
16
17  test(cats);
18
19  cats.forEach(cat => cat.meow()); // runtime error
20
```

Type Compatibility

```
interface Named {
    name: string;
}

class Person {
    name: string;
}

let p: Named;
// OK, because of structural typing
p = new Person();
```

https://www.typescriptlang.org/docs/handbook/type-compatibility.html

Type Safety

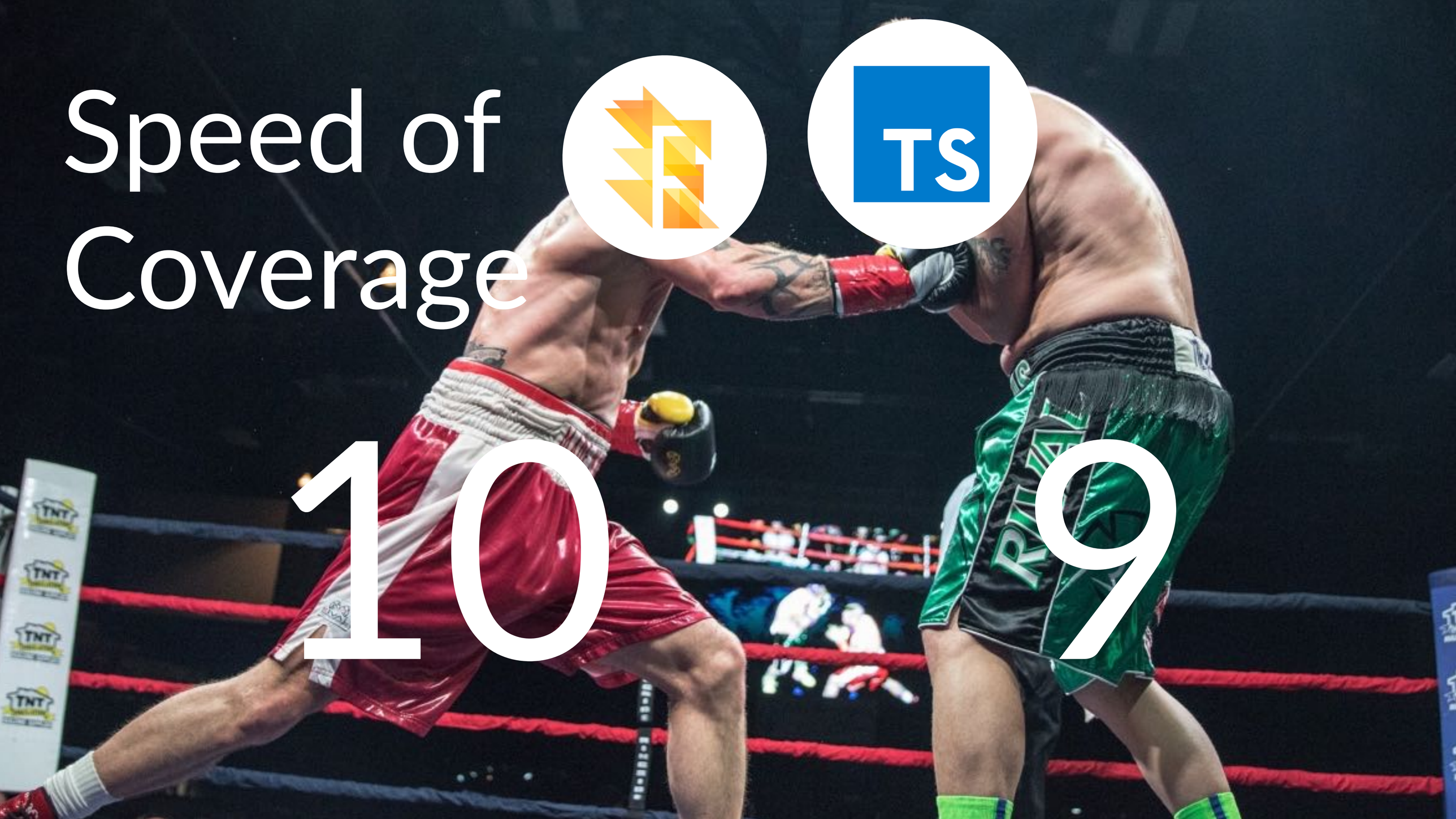
Round 5

Speed of Coverage

JavaScript Lifestyle with less types declarations

```
1  /* @flow */
2
3  function getLength(x) {
4    return x.length;
5  }
6
7  const total = getLength('Hello') + getLength(null);
8
```

Speed of Coverage



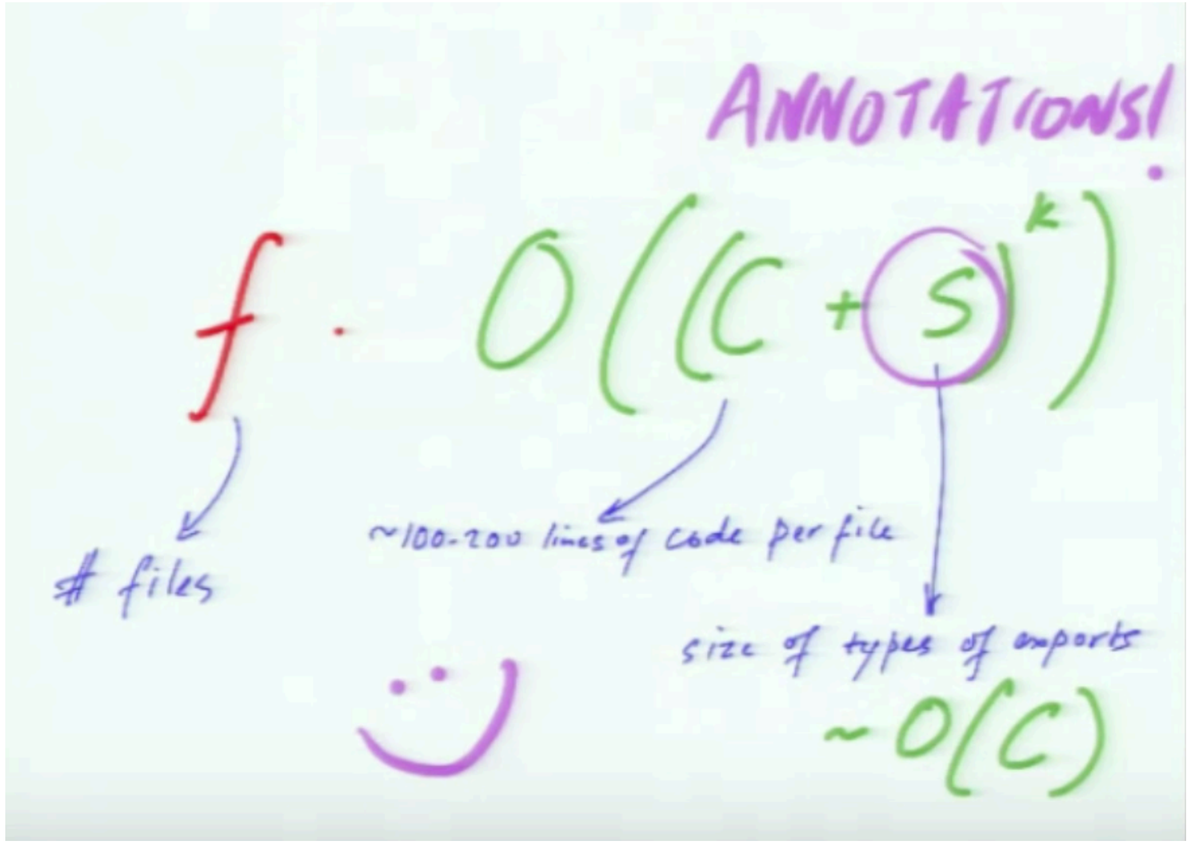
Round 6

Type-checking speed 

- speed degrades with each additional file

- speed does not degrade much as the project grows

Flow Speed



https://youtu.be/WgvAPzOmiP0?t=616


Type-
checking
speed
9
10
TS

Round 7

IDE integrations

Who use?

Who use?


VS Code

Michel Weststrate
@mweststrate

Читаю

My new favorite @VSCode feature: just add // @ts-check to any #javascript file for auto completion & error prevention. Tnx @typescriptlang!

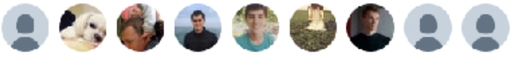
Язык твита: английский

```
// @ts-check
import { types } from "mobx-state-tree"        You, a few seconds ago •

export const Todo = types
    .model({
        text: "stuff",
        completed: types.boolean,
        id: types.number
    })
    .actions
        comp
            self.completed = 3
        }
    }))
```

[js] Type '3' is not assignable to type 'boolean'.
(property) completed: boolean

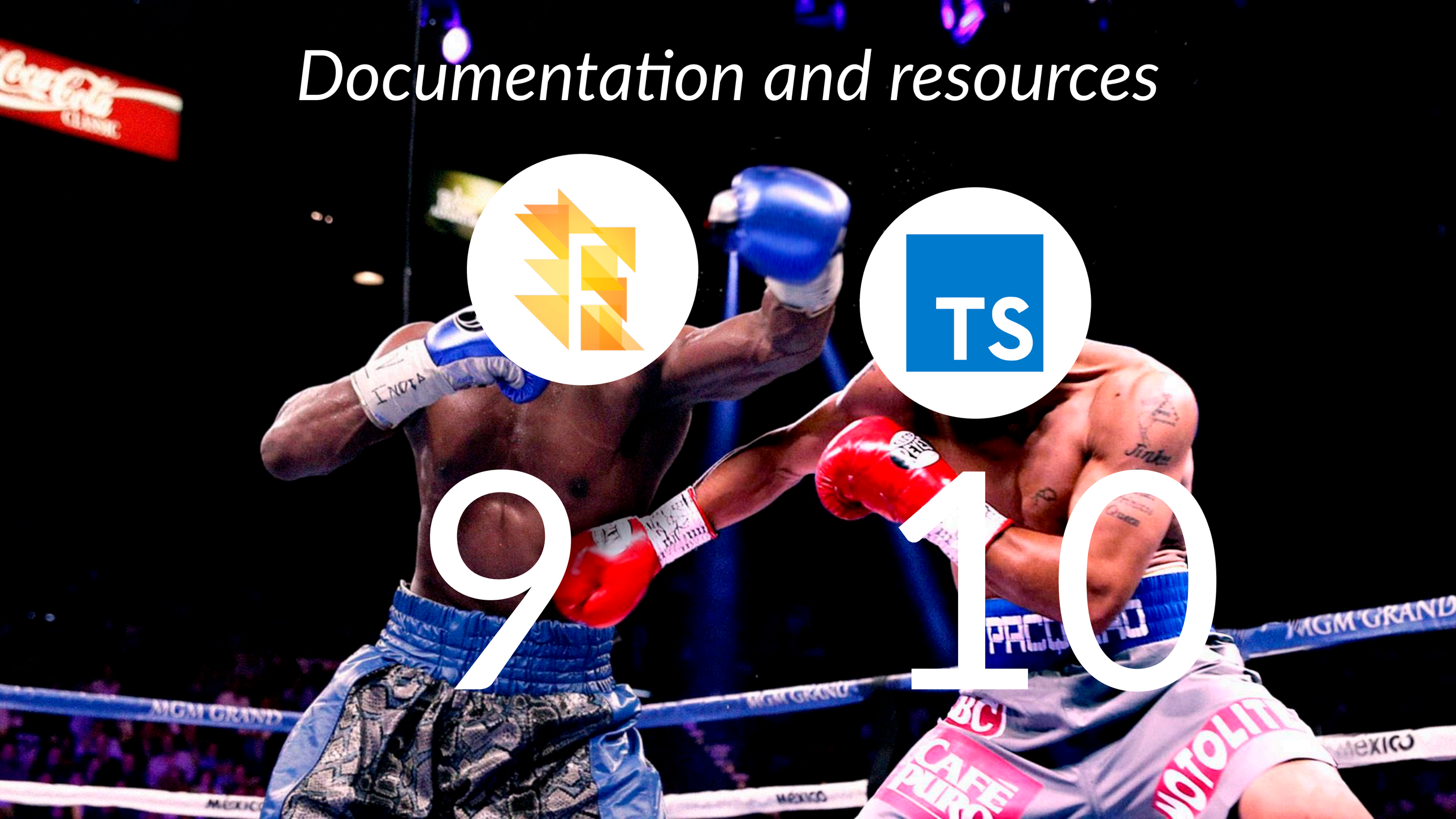
12:39 - 3 окт. 2017 г.

30 ретвитов **74** отметки «Нравится»



7 30 74

Documentation and resources



Round 8

Autocomplete

- only for usage
- feels sluggish (often a second or more of delay)
- feels unreliable (sometimes does not show up at all)

- both during declaration and usage
- feels instantaneous
- feels reliable

Autocomplete suggestions for missing properties when creating new objects/literals of an expected type #3074

New issue

Open niieani opened this issue on Dec 23, 2016 · 1 comment

niieani commented on Dec 23, 2016 • edited

TypeScript gives autocomplete suggestions for object properties and literals when a type is known at the time the object is being created / argument typed, i.e. explicitly declared, inferred or cast.

This is how it works:

tsconfig.json example.ts

```
type Hello = {
    one: string,
    two: string,
    three: string,
}

let a : Hello = {
    one: '123',
    three: '123'
}
```

Assignees

No one assigned

Labels

autocomplete

feature request

Projects

None yet

Milestone

No milestone

Notifications

Subscribe

You're not receiving notifications from this thread.

3 participants

https://github.com/facebook/flow/issues/3074

Autocomplete

Round 9

Error message quality

Code Snippet

```
type Dog = {
  name: string
  age: number
}
```

```
const dog: Dog = {
  name: 'Rolo',
  age: '2'
}
```

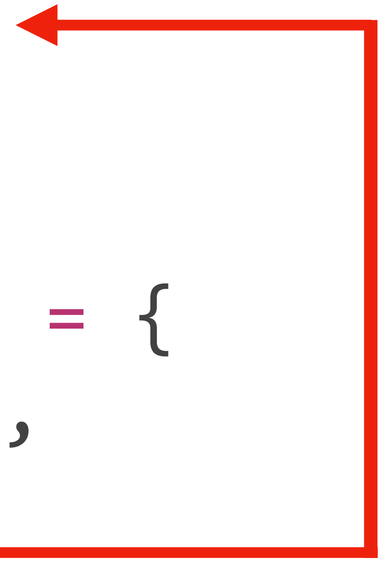
Code Snippet

```
type Dog = {
  name: string
  age: number
}
```

```
const dog: Dog = {
  name: 'Rolo',
  age: '2'
}
```

Code Snippet

```
type Dog = {
  name: string
  age: number
}

const dog: Dog = {
  name: 'Rolo',
  age: '2'
}
```



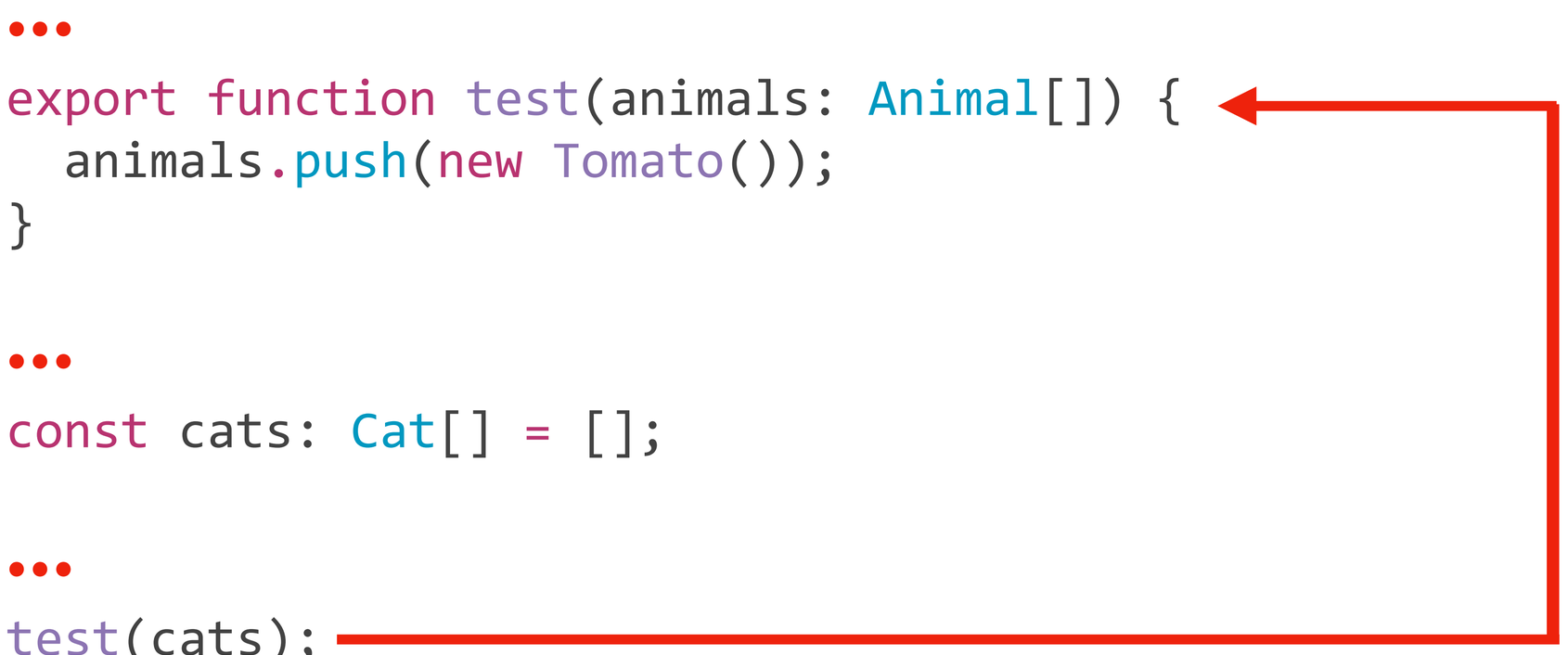
Resultes

```
[ts]

Type '{ name: string; age: string; }' is not assignable to type 'Dog'.
  Types of property 'age' are incompatible.
    Type 'string' is not assignable to type 'number'.
```

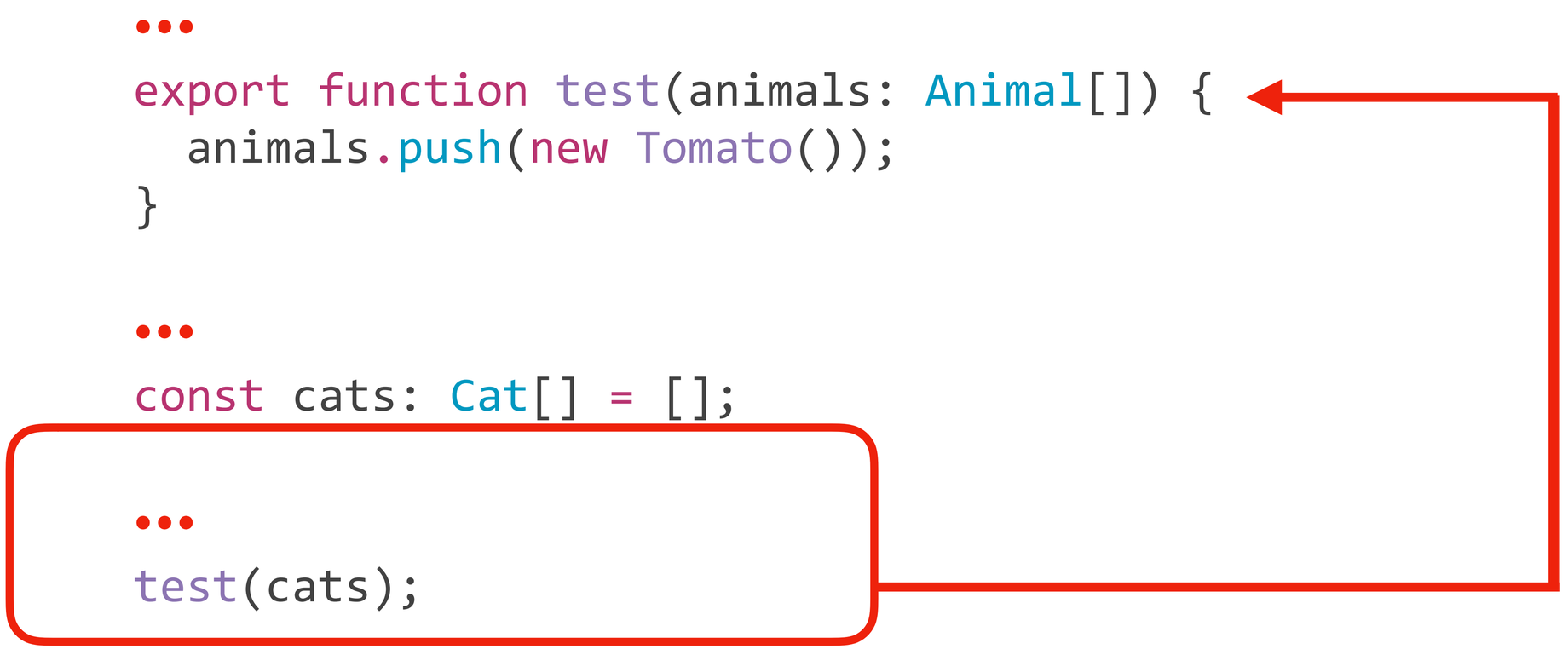
```
[flow] object literal
  (This type is incompatible with object type
  Property `age` is incompatible:)
```

Back to our Type Safe Example

```
/* @flow */

class Animal {}

class Cat extends Animal {
  meow() {}
}

class Tomato {}

export function test(animals: Animal[]) {
  animals.push(new Tomato());
}

const cats: Cat[] = [];

test(cats);

cats.forEach(cat => cat.meow()); // runtime error
```




```
9: export function test(animals: Animal[]) {
                                 ^^^^^^ Animal.This type is incompatible with
13: const cats: Cat[] = [];
                ^^^ Cat
```




```
9: export function test(animals: Animal[]) {
                                 ^^^^^^ Animal.This type is incompatible with
13: const cats: Cat[] = [];
                ^^^ Cat
```

Error Quality

Round 10

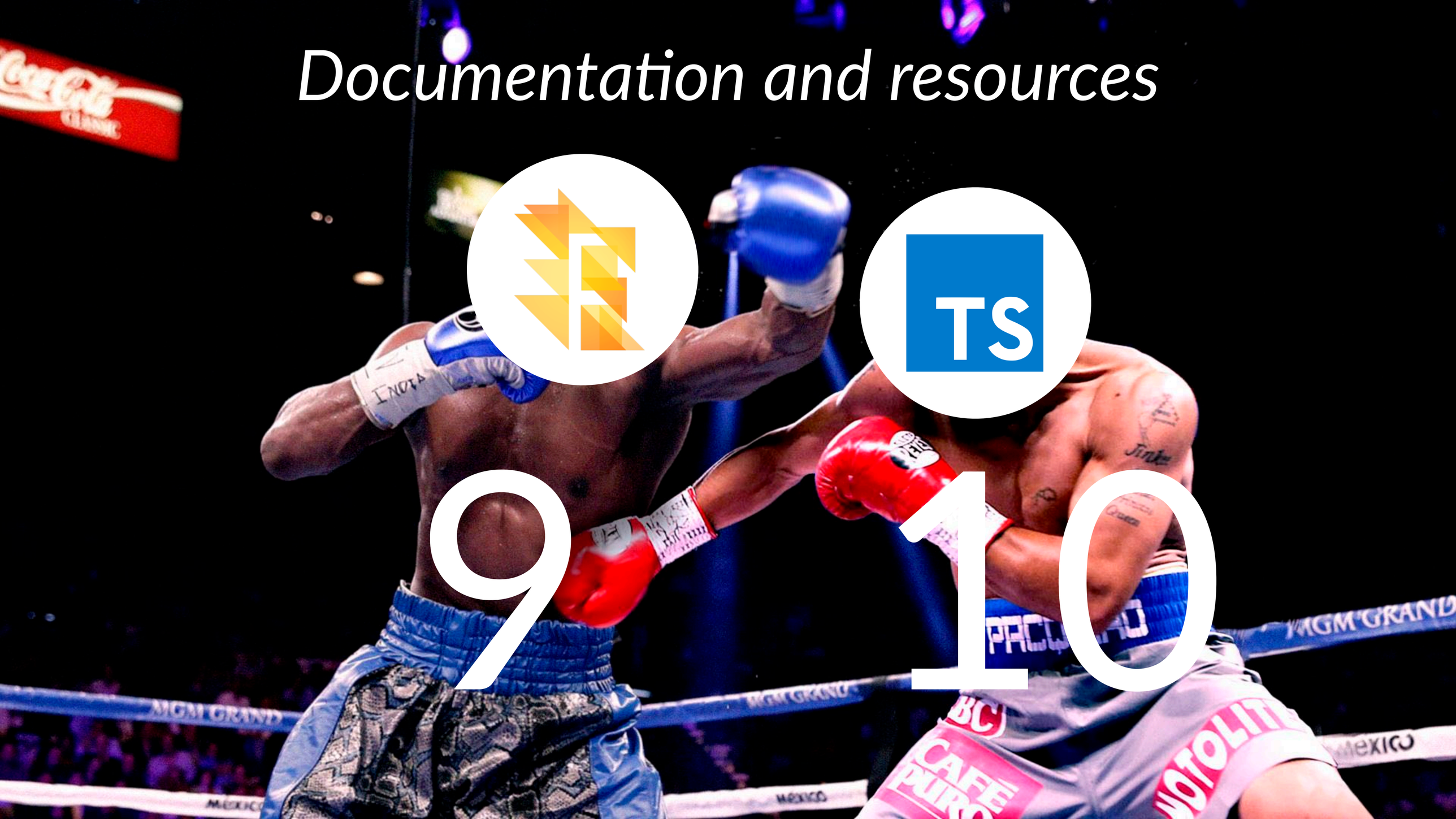
Documentation and resources

- incomplete, often vague docs
- blog posts

- very good docs
- many books
- videos
- blog posts

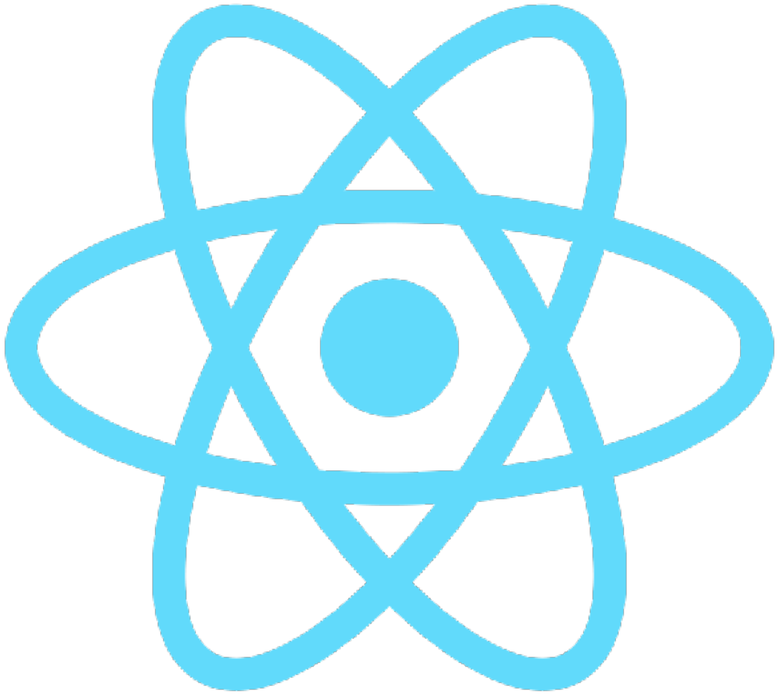
Start learn Reason if you choose Flow :)

Documentation and resources



Round 11

Usage with frameworks and libraries



VS

v0.53.0

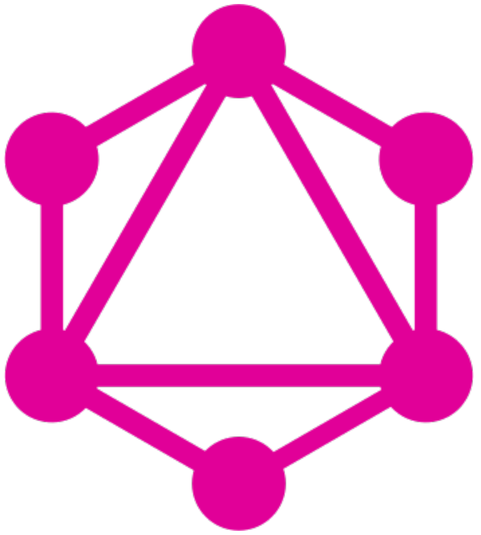
v0.53.0

facebook-github-bot released this on Aug 16

This release includes major changes to Flow's model for React. The following links contain detailed documentation on the new model.

- Defining components
- Event handling
- ref functions
- Typing children
- Higher-order components
- Utility type reference

https://github.com/facebook/flow/releases/tag/v0.53.0



Max Stoiber
@mxstbr

Читаю

This is how I want @GraphQL + @flowtype to integrate. Import GQL types with Flow's "import type", get flow types to use in your code. 😍

How do I make this happen? /cc @stubailo @calebmer

Язык твита: английский

```
// @flow
import type { UserInput } from '../User.graphql';
```

9:15 - 2 окт. 2017 г.

8 ретвитов **107** отметок «Нравится»

Flow

12 8 107

Not full

flowtype / flow-typed

Watch 45 | Star 1,670 | Fork 570

Code | Issues 186 | Pull requests 49 | Projects 1 | Wiki | Insights

Branch: master ▾ flow-typed / definitions / npm /

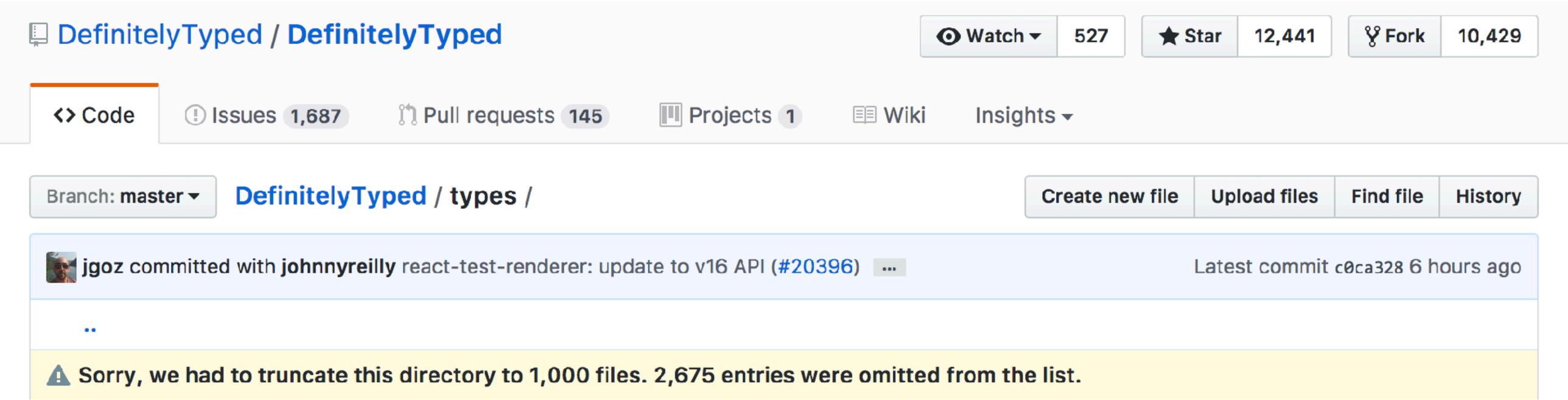
Create new file | Upload files | Find file | History

idris committed with GAntoine Express: error middleware must have error (#641) ... Latest commit 18a6cf4 10 hours ago

..

https://github.com/flowtype/flow-typed/tree/master/definitions/npm

Well maintained typings



https://github.com/DefinitelyTyped/DefinitelyTyped

TypeScript

support in

10 times

more libraries than

Flow

Typings for libraries

Round 12

Unique Features

- variance
- existential types *
- $Diff, $Shape, $Record, $Supertype<T>, $Subtype<T>, $Enum<T>

http://sitr.us/2015/05/31/advanced-features-in-flow.html

- autocomplete for object construction
- more flexible type mapping via iteration
- namespacing
- overloading of functions
- TS generics

Unique Features

Conclusion

So who wins?

Flow or **TypeScript**?

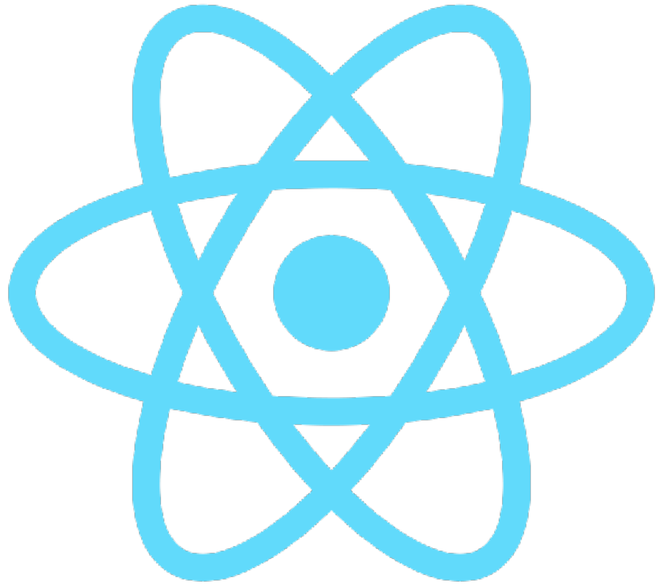

TS
114
112

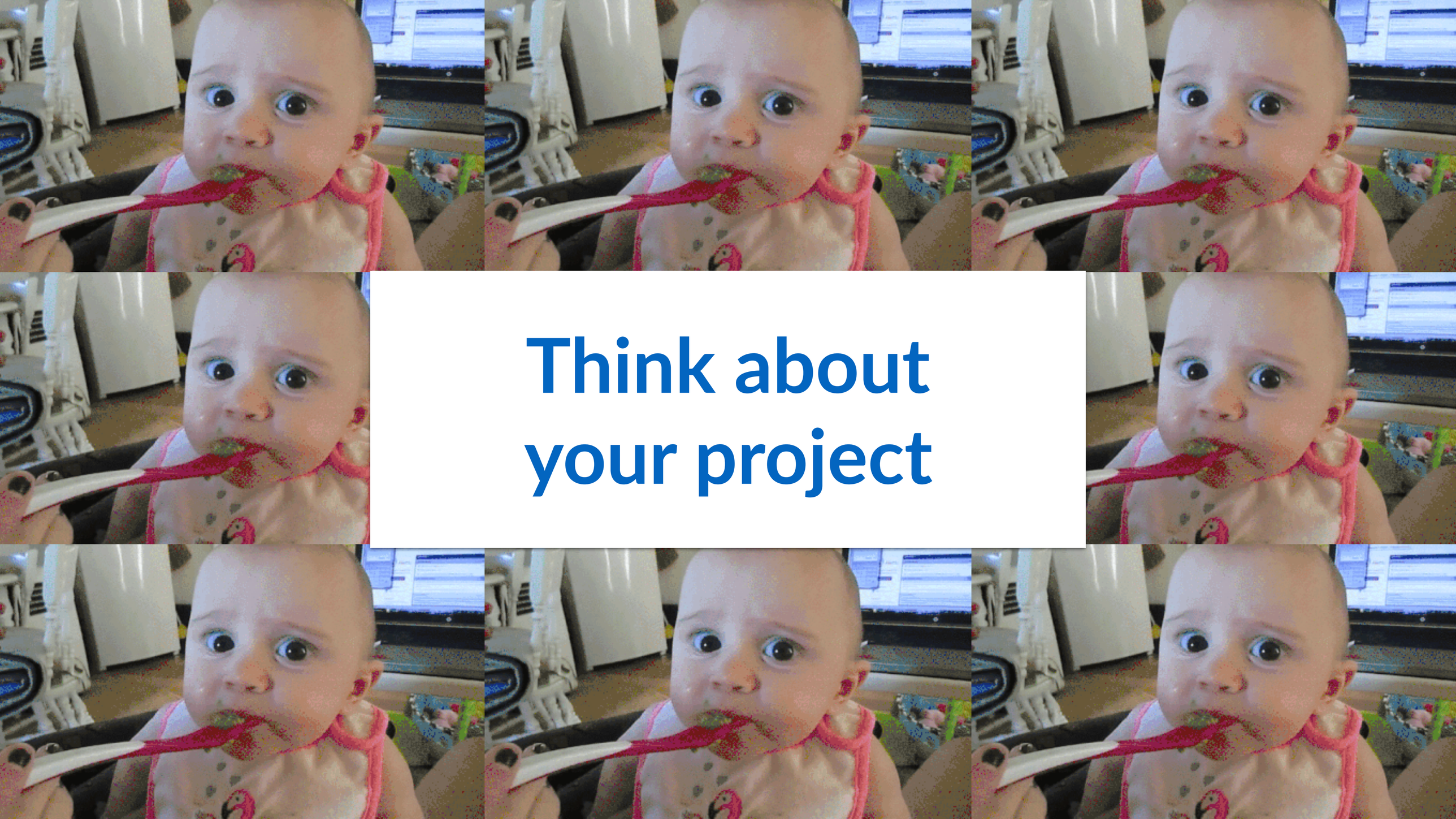
I choose TypeScript

What is next?

Next time…


REASON





Think about your project

Do only cool projects


PEKLAMA


We’re hiring!
https://www.grammarly.com/jobs/engineering
https://tech.grammarly.com

Thank you!

Questions are welcome

@AGambit95