Let's Build a Great Typing Experience with 3rd Party Modules

Patrick Stapfer

Twitter: @ryyppy
Github: ryyppy

Flowtype

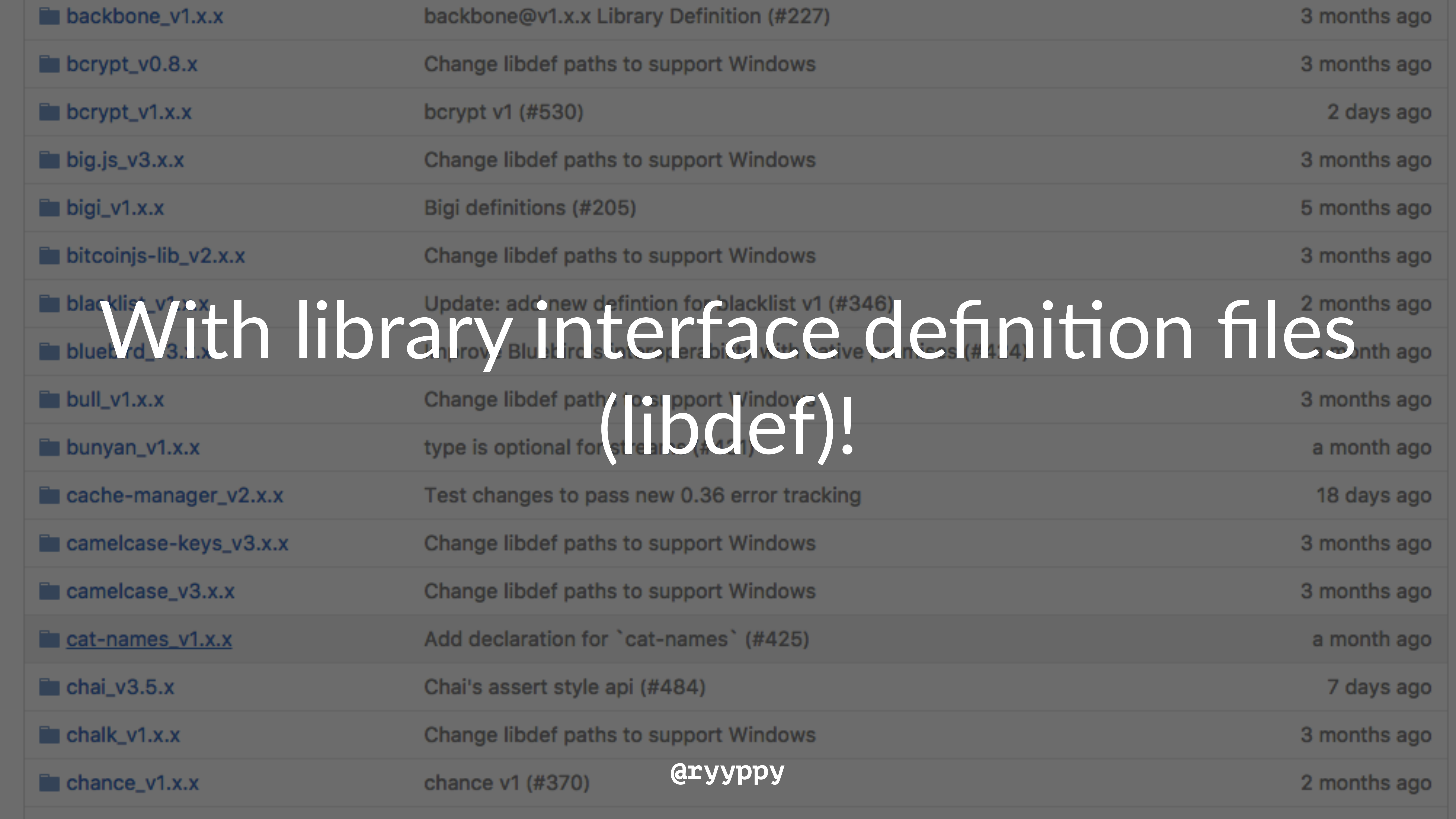
Adds a type layer to your vanilla JavaScript code

```
function doSomething(a: string, b: number): string {
  return `${a} + ${b}`;
}
```

How do we support types for 3rd party code?

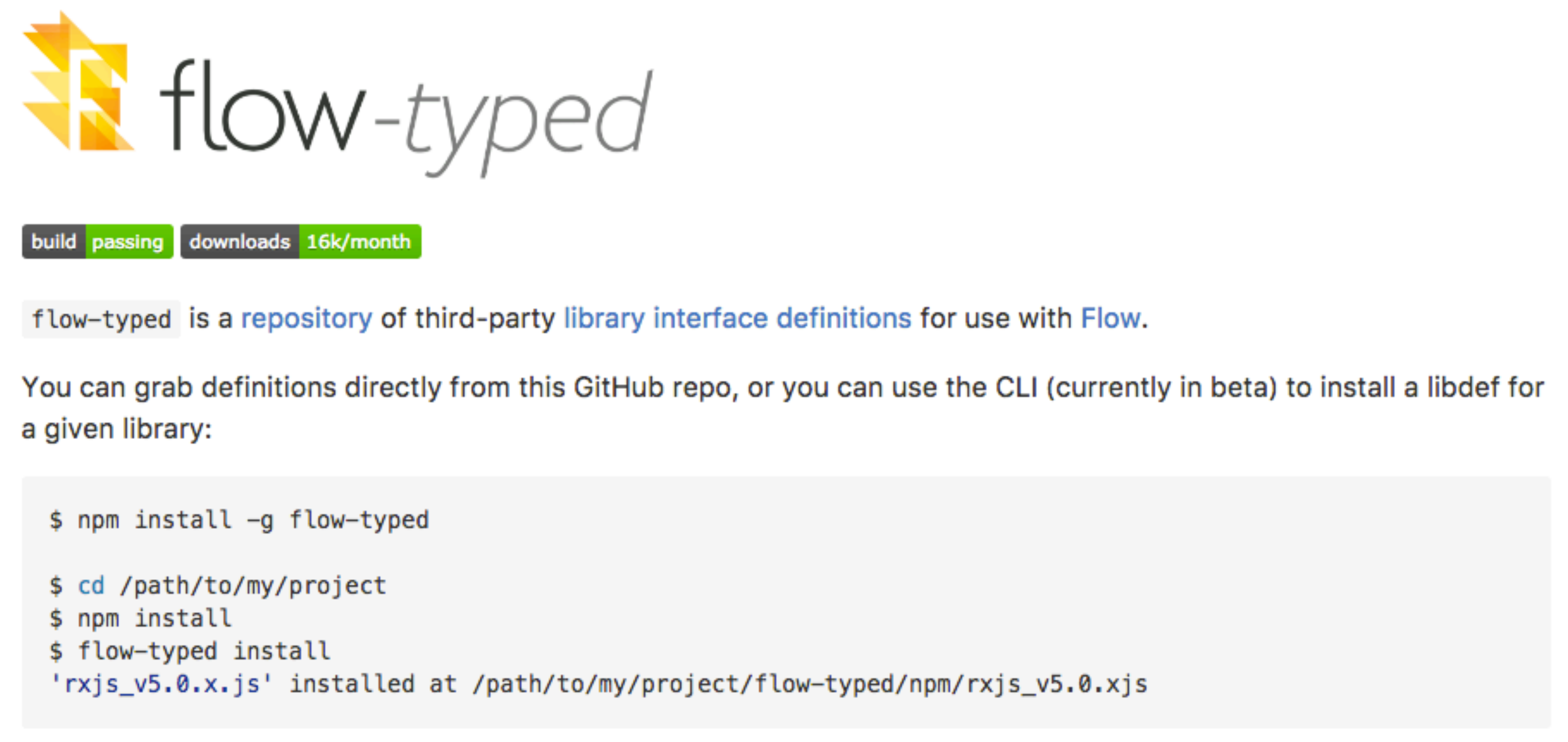
```
import { map } from 'lodash';

// How do I teach flow about map?
map('foo');
```

With library interface definition files (libdef)!



backbone_v1.x.x	backbone@v1.x.x Library Definition (#227)	3 months ago
bcrypt_v0.8.x	Change libdef paths to support Windows	3 months ago
bcrypt_v1.x.x	bcrypt v1 (#530)	2 days ago
big.js_v3.x.x	Change libdef paths to support Windows	3 months ago
bigi_v1.x.x	Bigi definitions (#205)	5 months ago
bitcoinjs-lib_v2.x.x	Change libdef paths to support Windows	3 months ago
blacklist_v1.x.x	Update: add new defintion for blacklist v1 (#346)	2 months ago
bluebird_v3.x.x	Improve Bluebird's interoperability with native promises (#414)	a month ago
bull_v1.x.x	Change libdef paths to support Windows	3 months ago
bunyan_v1.x.x	type is optional for streams (#441)	a month ago
cache-manager_v2.x.x	Test changes to pass new 0.36 error tracking	18 days ago
camelcase-keys_v3.x.x	Change libdef paths to support Windows	3 months ago
camelcase_v3.x.x	Change libdef paths to support Windows	3 months ago
cat-names_v1.x.x	Add declaration for `cat-names` (#425)	a month ago
chai_v3.5.x	Chai's assert style api (#484)	7 days ago
chalk_v1.x.x	Change libdef paths to support Windows	3 months ago
chance_v1.x.x	chance v1 (#370)	2 months ago

@ryyppy



flow-typed

build passing downloads 16k/month

`flow-typed` is a repository of third-party library interface definitions for use with Flow.

You can grab definitions directly from this GitHub repo, or you can use the CLI (currently in beta) to install a libdef for a given library:

```
$ npm install -g flow-typed

$ cd /path/to/my/project
$ npm install
$ flow-typed install
'rxjs_v5.0.x.js' installed at /path/to/my/project/flow-typed/npm/rxjs_v5.0.xjs
```

Flow-Typed

- Collection of High-Quality Library Definitions
- Tagged to specific flow / library versions
- Independent from the actual implementation
- CLI for easy management

```
$ flow-typed search jest

Found definitions:
╔══════╤═════════════════╤══════════════╗
║ Name │ Package Version │ Flow Version ║
╟──────┼─────────────────┼──────────────╢
║ jest │ v17.x.x         │ >=v0.33.x    ║
╟──────┼─────────────────┼──────────────╢
║ jest │ v16.x.x         │ >=v0.33.x    ║
╟──────┼─────────────────┼──────────────╢
║ jest │ v14.x.x         │ >=v0.23.x    ║
╟──────┼─────────────────┼──────────────╢
║ jest │ v12.0.x         │ >=v0.23.x    ║
╚══════╧═════════════════╧══════════════╝
```

```
$ flow-typed install jest@17.x.x -f 0.36

• rebasing flow-typed cache...done.
• Installing 1 libdefs...
  • jest_v17.x.x.js
    └> ./flow-typed/npm/jest_v17.x.x.js
```

`"./flow-typed"` is our default libdef directory

```
#.flowconfig
[ignore]

[include]

[libs]
flow-typed

[options]
```

How does a libdef file look like?

Declaration Syntax

```
// Type declaration
declare type MyObject = { test: boolean };

// Function declaration
declare function add(a: number, b: number): number;

// Module declaration
declare module 'mymodule' { /* */ }

// Special exports declaration (CommonJS)
declare var exports: MyObject;

// Variable declaration
declare var MY_CONSTANT: string;
```

Module Scoping

@ryyppy

Global Scope

-> Everything declared outside a `declare module` context:

```
// ./flow-typed/global.js
declare function foo(a: string, b: number): void;

// ./main.js
foo('test', 1); // Okay
foo(1, ''); // Flow error
```

CommonJS Module

```
declare module 'commonJSModule' {
  declare function notExposed(): void;
  declare type MyEntity = { id: string, name: string };

  declare type MyExport = {
    toEntity: (obj: Object) => MyEntity
  };


  declare var exports: MyExport;
}

// ./main.js
import commonJSModule from 'commonJSModule';

// $FlowExpectedError: hidden internal
import { notExposed } from 'commonJSModule';
```

ES6 Module

```
// ./flow-typed/es6module.js
declare module 'es6module' {
  declare export var buffers: Array<string>;
  declare export var otherBuffer: string;

  declare type MyES6Module = {
    buffers: typeof buffers,
  };

  declare export default MyES6Module;
}
```

```
// ./main.js
import { buffers, otherBuffer, default as es6module } from 'es6module';
```

"Automatic" CommonJS Module

-> If you don't provide any module specific `export` declarations

```
// ./flow-typed/autoCommonJSModule.js
declare module 'autoCommonJSModule' {
  declare function exposed1(): void;
  declare var SOME_CONSTANT: string;
}
```

```
// ./main.js
import { SOME_CONSTANT, exposed1 } from 'autoCommonJSModule';
import autoCommonJSModule from 'autoCommonJSModule';
```

Submodules / Namespacing

-> Namespacing is important to reuse types for submodules

```
// ./flow-typed/submodules.js
declare type npm$mymodule$Shared = number;

declare module 'mymodule/lib' {
  declare type MyExport = npm$mymodule$Shared;
  declare export default MyExport;
}

declare module 'mymodule/lib/something' {
  declare var exports: {
    shared: npm$mymodule$Shared,
  };
}


// ./main.js
import myModule from 'mymodule/lib';
import myModuleSomething from 'mymodule/lib/something';
```

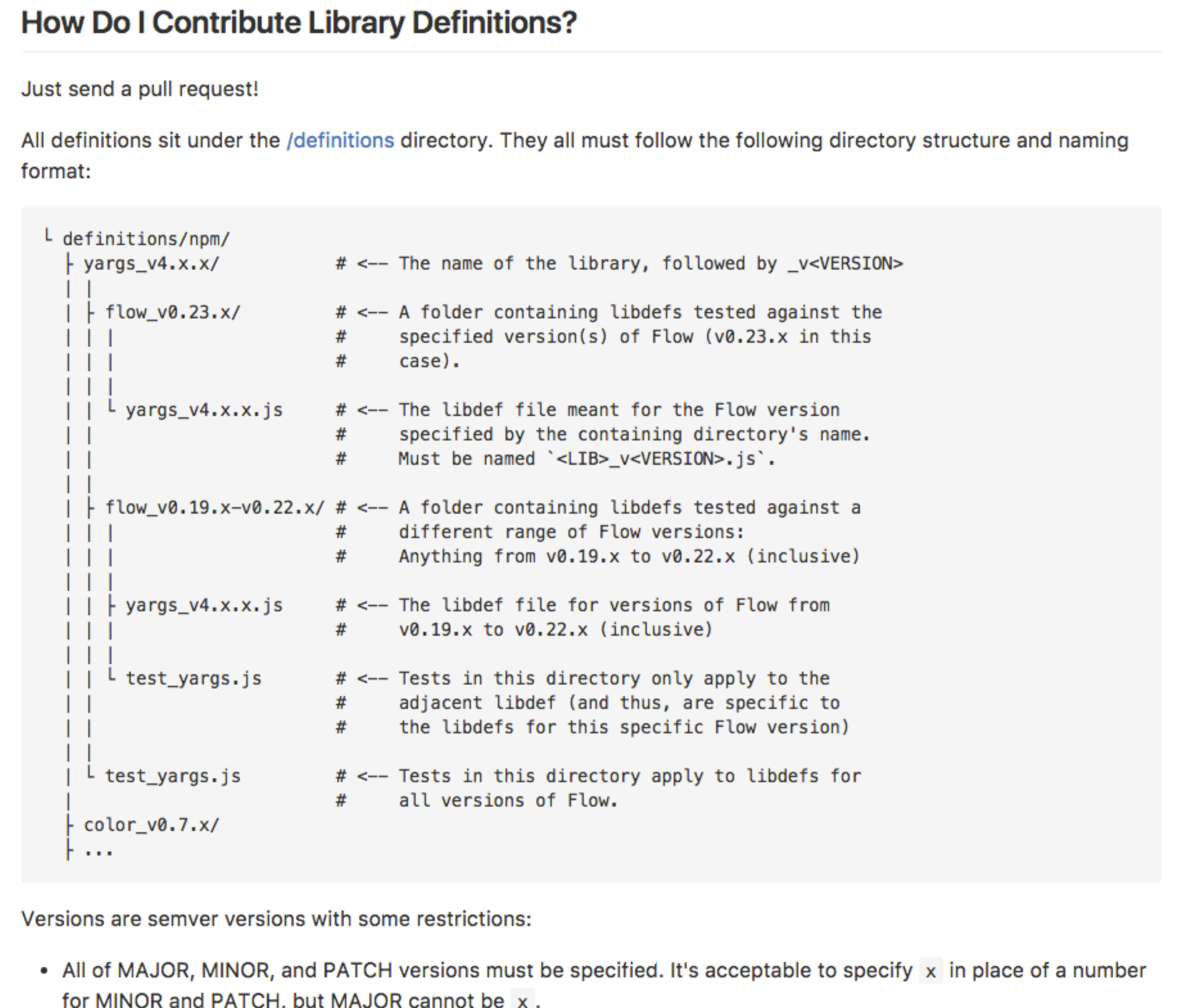
Export / Import types from libdef

-> It's also possible to export types

```
declare module 'typeModule' {
  declare export type SomeType = {
    foo: string,
  };
}

import type { SomeType } from 'typeModule';
const foo: SomeType = { foo: 'test' };
```

Contributing libdefs to flow-typed

@ryyppy



How Do I Contribute Library Definitions?

Just send a pull request!

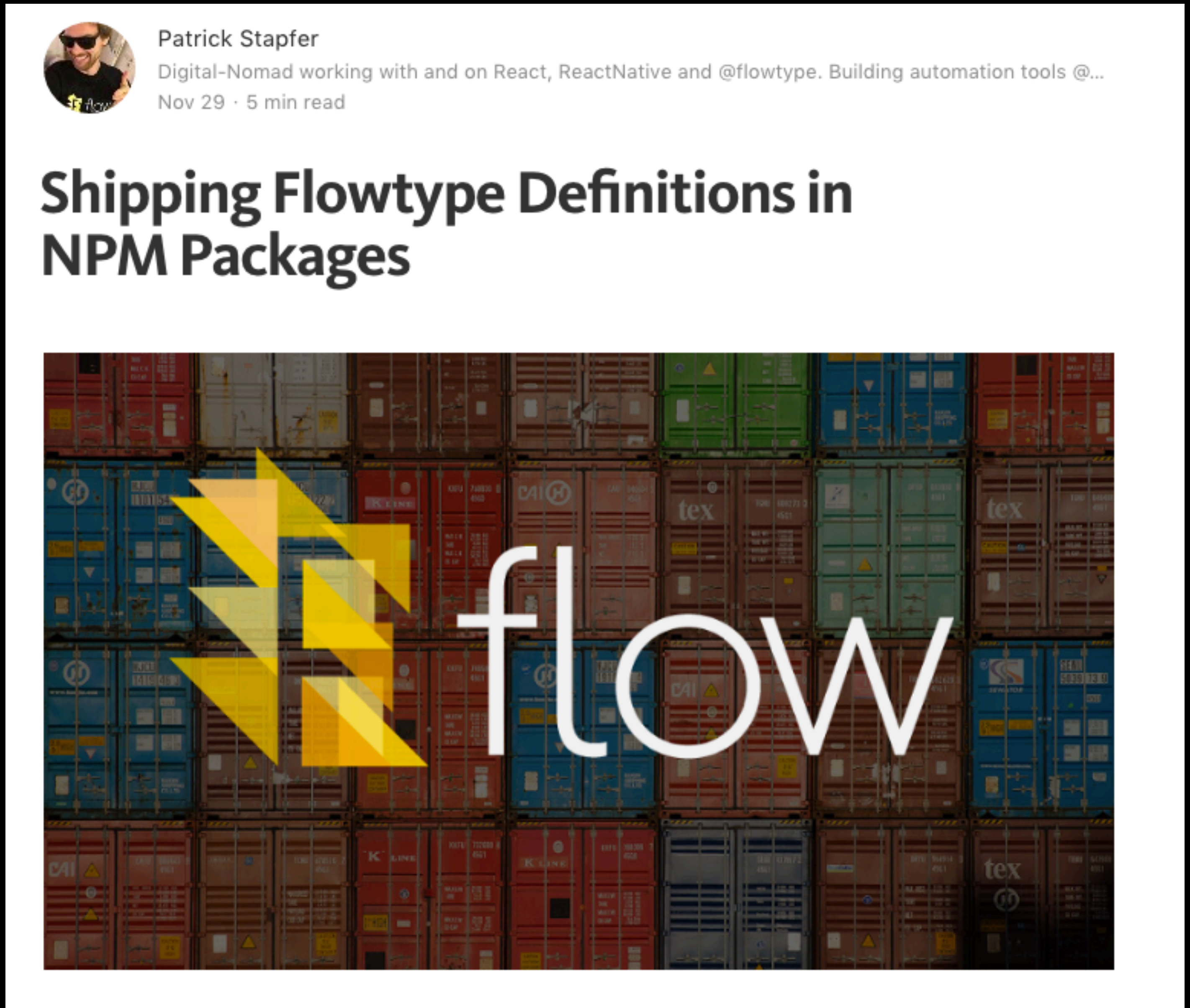
All definitions sit under the /definitions directory. They all must follow the following directory structure and naming format:

```
└ definitions/npm/
  ├ yargs_v4.x.x/           # <-- The name of the library, followed by _v<VERSION>
  | |
  | ├ flow_v0.23.x/         # <-- A folder containing libdefs tested against the
  | | |                     #     specified version(s) of Flow (v0.23.x in this
  | | |                     #     case).
  | | |
  | | └ yargs_v4.x.x.js     # <-- The libdef file meant for the Flow version
  | |                       #     specified by the containing directory's name.
  | |                       #     Must be named `<LIB>_v<VERSION>.js`.
  | |
  | ├ flow_v0.19.x-v0.22.x/ # <-- A folder containing libdefs tested against a
  | | |                     #     different range of Flow versions:
  | | |                     #     Anything from v0.19.x to v0.22.x (inclusive)
  | | |
  | | ├ yargs_v4.x.x.js     # <-- The libdef file for versions of Flow from
  | | |                     #     v0.19.x to v0.22.x (inclusive)
  | | |
  | | └ test_yargs.js       # <-- Tests in this directory only apply to the
  | |                       #     adjacent libdef (and thus, are specific to
  | |                       #     the libdefs for this specific Flow version)
  | |
  | └ test_yargs.js         # <-- Tests in this directory apply to libdefs for
  |                         #     all versions of Flow.
  ├ color_v0.7.x/
  ├ ...
```

Versions are semver versions with some restrictions:

- All of MAJOR, MINOR, and PATCH versions must be specified. It's acceptable to specify `x` in place of a number for MINOR and PATCH, but MAJOR cannot be `x`.

Other Alternatives



Patrick Stapfer

Digital-Nomad working with and on React, ReactNative and @flowtype. Building automation tools @...

Nov 29 · 5 min read

Shipping Flowtype Definitions in NPM Packages

Typed JavaScript Is the Future

@ryyppy

Thanks

@ryyppy

References

- Source-Code (Presentation)
- Flow-Typed
- Shipping Flowtype Definitions in NPM Packages
- Flowtype Declaration Syntax