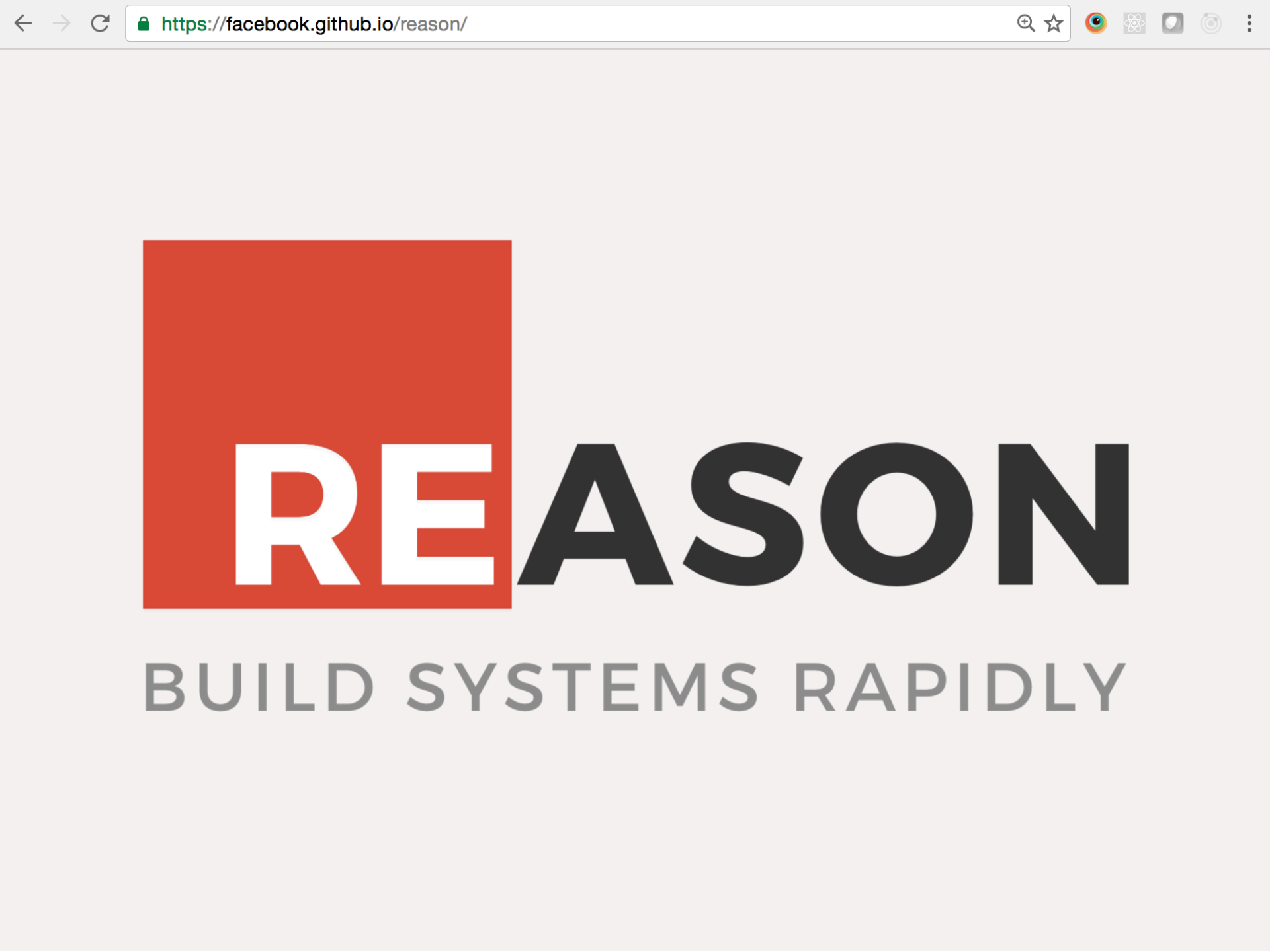

https://facebook.github.io/reason/
REASON
BUILD SYSTEMS RAPIDLY

Basic Language Primitives

JavaScript	Reason
`3`	`3`
`3.1415`	`3.1415`
`"Hello world!"`	`"Hello world!"`
`'Hello world!'`	Strings must use “
Characters are strings	`'a'`
`true`	`true`
`[1,2,3]`	`[1,2,3]`
`null`	`()`
`const x = y;`	`let x = y;`
`let x = y;`	`reference cells`
`var x = y;`	No equivalent (thankfully)
`[x, ...lst] (linear time)`	`[x, ...lst] (constant time)`
`[...lst, x] (linear time)`	`Not supported`
`{...obj, x: y}`	`{...obj, x: y}`

Objects and Records

JavaScript	Reason
“Objects”	“Records”
no static types	`type point = {x: int, mutable y: int};`
`{x: 30, y: 20}`	`{x: 30, y: 20}`
`point.x`	`point.x`
`point.y = 30;`	`point.y = 30;`
`{...point, x: 30}`	`{...point, x: 30}`

JavaScript	Reason
`const incr = x => x + 1;`	`let incr = fun x => x + 1;`
`const five = incr(4);`	`let five = incr 4;`
`const add = (x, y) => x+y;`	`let add = fun x y => x+y;`
`const x = add(3, 4);`	`let x = add 3 4;`
`const y = add(3, add(0, 1));`	`let y = add 3 (add 0 1);`

OCaml

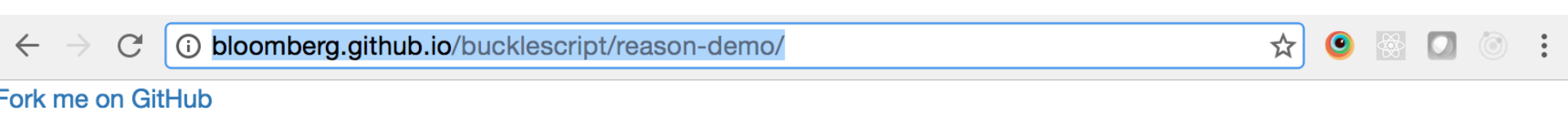
```
1
2 (** Imperative style *)
3 let sum n =
4     let v  = ref 0 in
5     for i = 0 to n do
6        v := !v + i
7     done;
8     !v
9
10 let () = Js.log (sum 100)
```

JavaScript

```
1 Warnings:


1 > 5050
2

1 // Generated by BUCKLESCRIPT VERSION 1.1.2 ,
  PLEASE EDIT WITH CARE
2 'use strict';
3
4
5 function sum(n) {
6   var v = 0;
7   for(var i = 0; i <= n; ++i){
8     v = v + i | 0;
9   }
10  return v;
11 }
12
13 console.log(sum(100));
14
15 exports.sum = sum;
16 /*  Not a pure module */
17
```



Fork me on GitHub

```
external to_str : 'a => string =
"js_anything_to_string";

external to_json_string : 'a => string =
"js_json_stringify";

let debug x => print_endline (to_str x);

let pprint x => print_endline (to_json_string
x);

let rec fib =
  fun
  | 1
  | 2 => 1
  | n => fib (n - 1) + fib (n - 2);

/** Imperative style */
let sum n => {
  let v = ref 0;
  for i in 0 to n {
    v.contents = v.contents + i
  };
  v.contents
};

let tail_sum n => {
```

```
Warnings:


>"h,e,l,l,o,o,c,a,m,l"
```

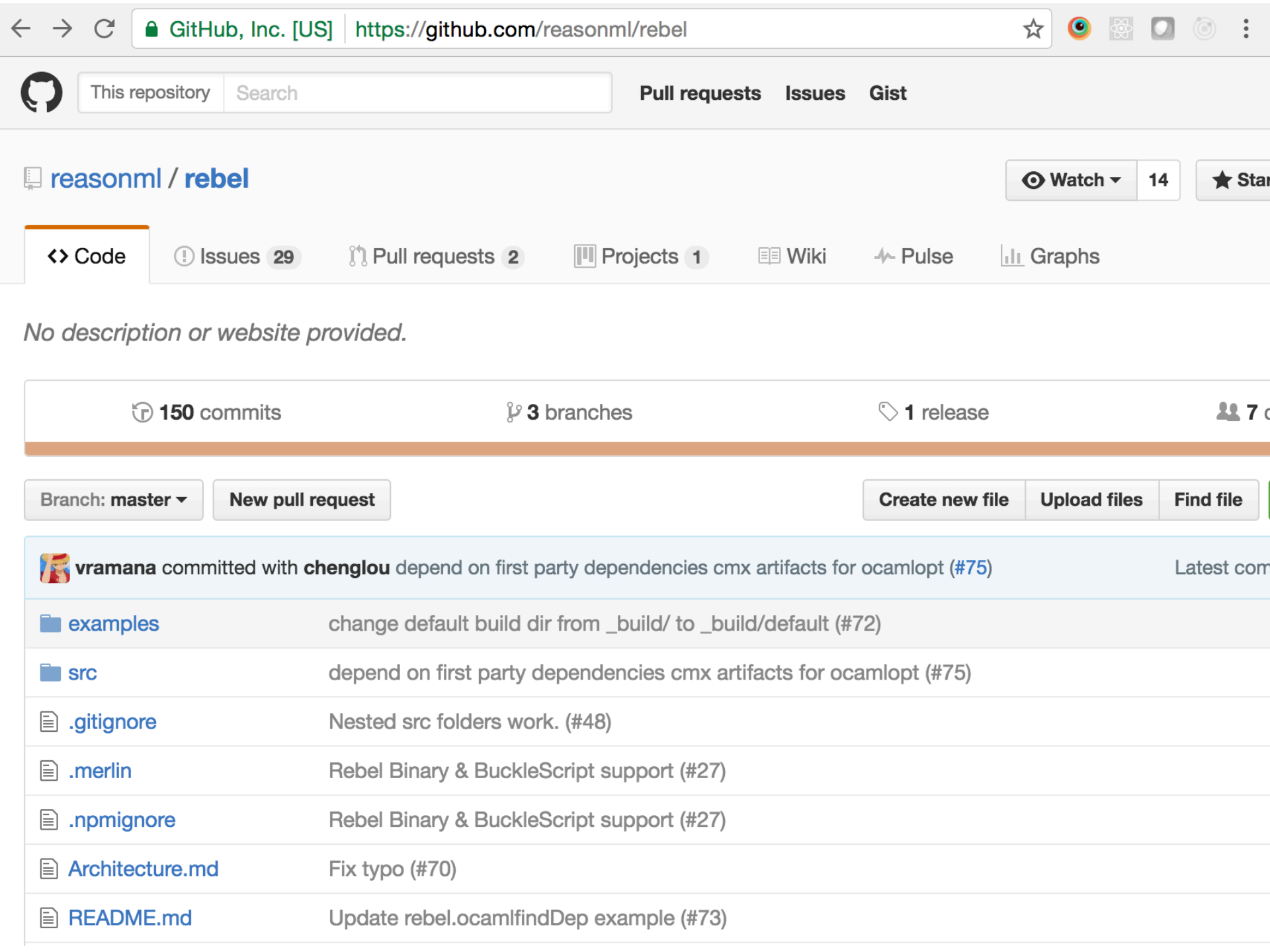
```
// GENERATED CODE BY BUCKLESCRIPT VERSION 0.3 ,
PLEASE EDIT WITH CARE
'use strict';

var Curry    = require("./runtime/curry");
var $$Array  = require("./stdlib/array");
var $$String = require("./stdlib/string");

function debug(x) {
  console.log("" + x);
  return /* () */0;
}

function pprint(x) {
  console.log(JSON.stringify(x));
  return /* () */0;
}

function fib(n) {
  if (n === 2 || n === 1) {
    return 1;
```


GitHub, Inc. [US]
https://github.com/reasonml/rebel
This repository
Search
Pull requests
Issues
Gist
reasonml / rebel
Watch
14
Code
Issues 29
Pull requests 2
Projects 1
Wiki
Pulse
Graphs
No description or website provided.
150 commits
3 branches
1 release
Branch: master
New pull request
Create new file
Upload files
Find file
vramana committed with chenglou depend on first party dependencies cmx artifacts for ocamlopt (#75)
examples
change default build dir from _build/ to _build/default (#72)
src
depend on first party dependencies cmx artifacts for ocamlopt (#75)
.gitignore
Nested src folders work. (#48)
.merlin
Rebel Binary & BuckleScript support (#27)
.npmignore
Rebel Binary & BuckleScript support (#27)
Architecture.md
Fix typo (#70)
README.md
Update rebel.ocamlfindDep example (#73)

Features

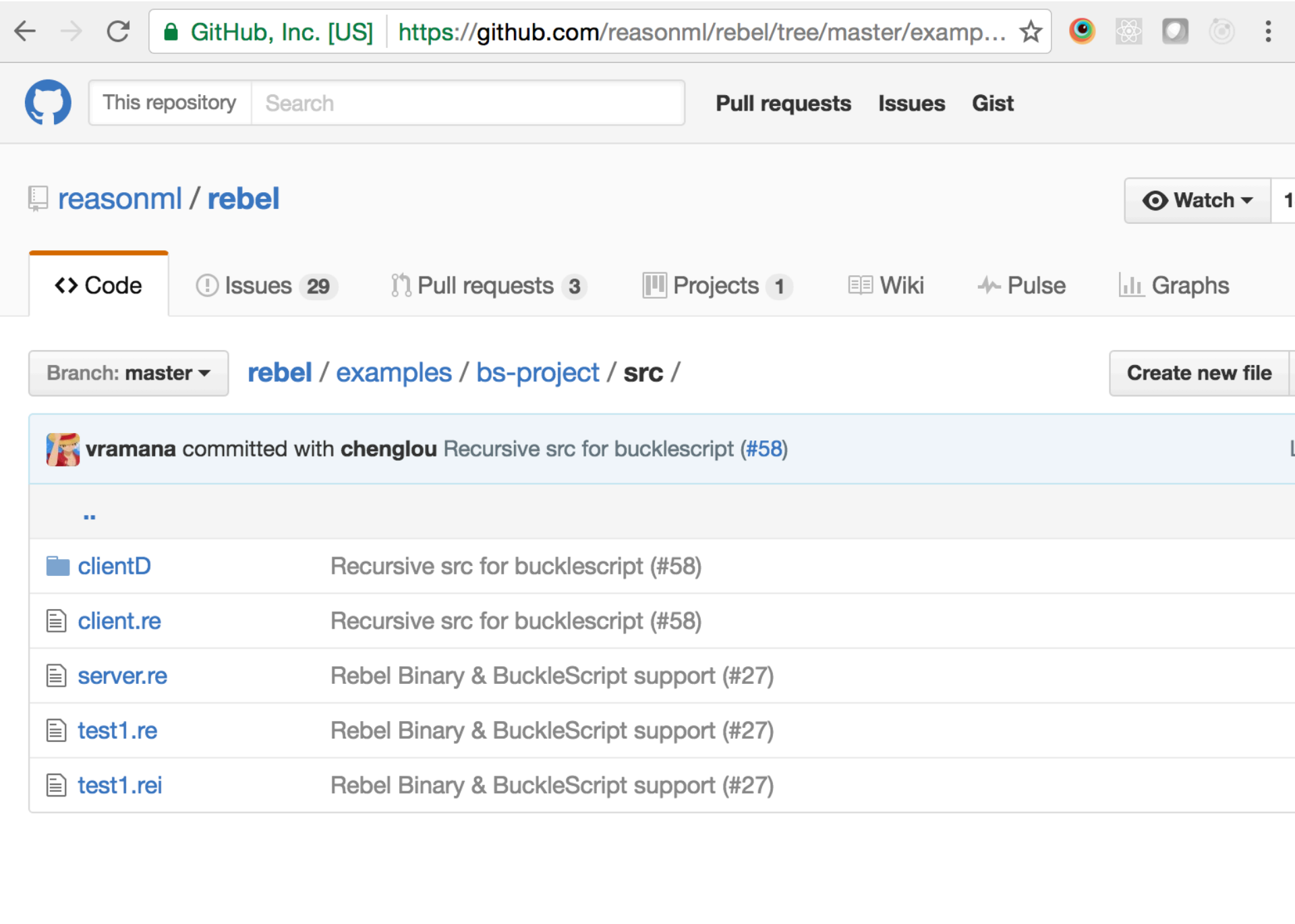
- Lightning build speed: fast startup, incremental, parallel, etc.
- `rebel` to build everything. No need for sub-build commands since only changed things are rebuilt.
- Works with npm-style `node_modules` third-party dependencies that follows the spec (see next section).
- Works with OPAM/ocamlfind dependencies: put them in the `rebel.ocamlfindDependencies` field, like so.
- Works with Reason and vanilla OCaml syntax, out of the box.
- Generates JavaScript output through js_of_ocaml or bucklescript.
- Peace-of-mind "wipe the whole world" : just remove `_build/`.
- Share your work through the normal npm workflow: `npm publish`.
- Integrated BetterErrors.

Coming soon:

- Generates the correct .merlin files for Merlin, for editor assistance.
- Generates utop/rtop bootstrap file.
- Generates documentation based on `mli` / `rei` interface files.
- Easier installation: precompiled binary, etc.

```
  "targets": [
    {
      "target": "native",
      "engine": "native",
      "entry": "src/a.re"
    },
    {
      "target": "byte",
      "engine": "byte",
      "entry": "src/a.re"
    },
    {
      "target": "js-nice",
      "engine": "bucklescript",
      "entry": "src/a.re"
    },
    {
      "target": "js-ugly",
      "engine": "jsoo",
      "entry": "src/a.re"
    }
```


A Reason To Witness


GitHub, Inc. [US]
https://github.com/reasonml/rebel/tree/master/examp...
This repository
Search
Pull requests
Issues
Gist
reasonml / rebel
Watch
Code
Issues 29
Pull requests 3
Projects 1
Wiki
Pulse
Graphs
Branch: master
rebel / examples / bs-project / src /
Create new file
vramana committed with chenglou Recursive src for bucklescript (#58)
..
clientD
Recursive src for bucklescript (#58)
client.re
Recursive src for bucklescript (#58)
server.re
Rebel Binary & BuckleScript support (#27)
test1.re
Rebel Binary & BuckleScript support (#27)
test1.rei
Rebel Binary & BuckleScript support (#27)

Branch: master ▾ **rebel** / examples / bs-project / src / **server.re**

20 lines (12 sloc) | 410 Bytes

```
/*
 * vim: set ft=rust:
 * vim: set ft=reason:
 */
let f () => "Hello World, Compiled from REBEL!!";

let g () => Test1.my_secret;

open Rexpress.Express;

let __dirname: Js.undefined string = [%bs.node __dirname];

let app = Express.express ();

Express.get app "/" (fun req res => Response.json res [%bs.obj {root: __dirname}]);

Express.use app (Express.static path::"__dirname");

Express.listen app 3000;
```

Branch: master ▾ **Rexpress** / src / **express.re**

24 lines (20 sloc) | 668 Bytes

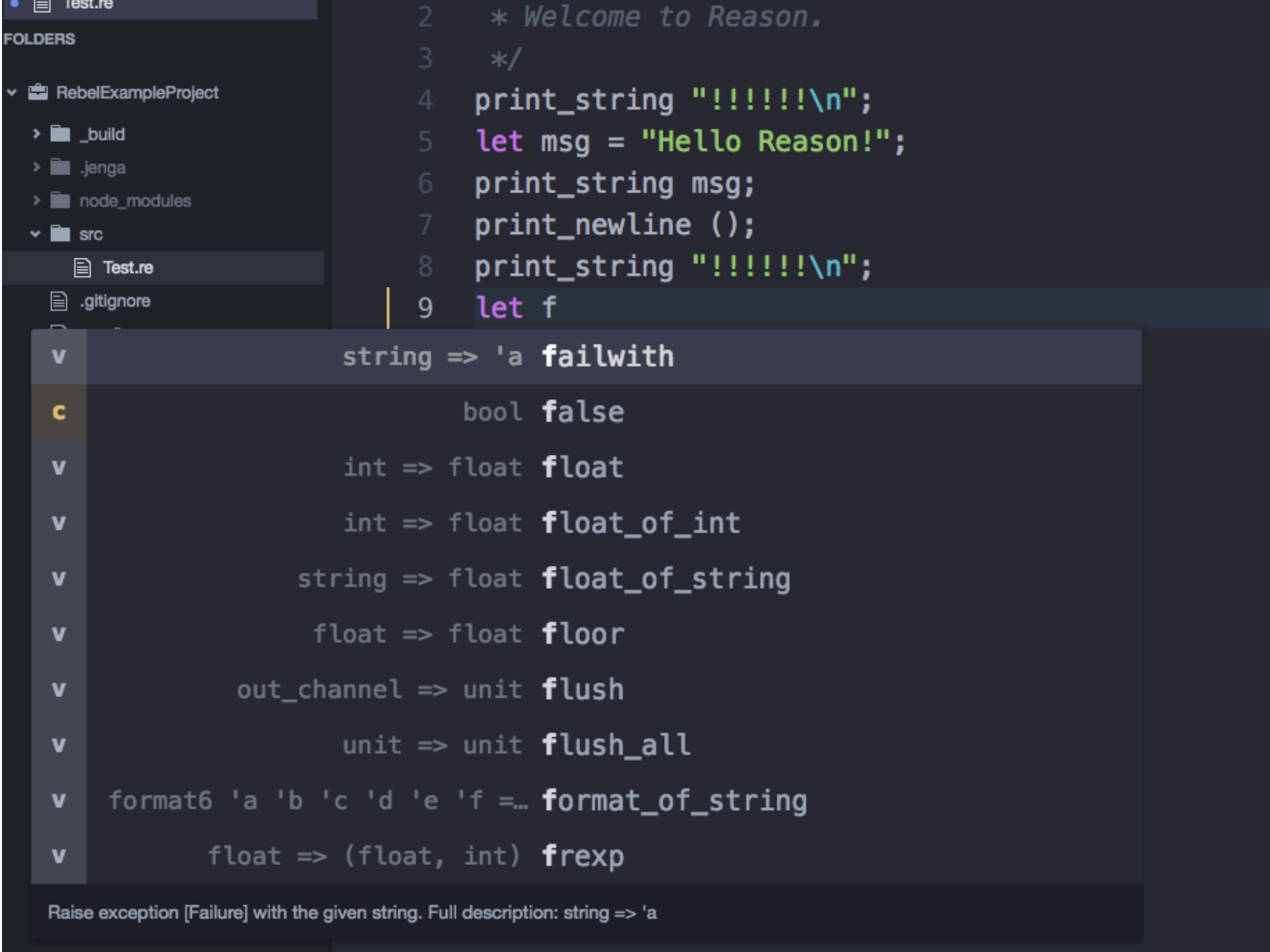
```
let module Next = {
  type t;
};

let module Request = {
  type t;
};

let module Response = {
  type t;
  external sendFile : t => string => 'a => unit = "" [@@bs.send];
  external json: t => 'a => unit = "" [@@bs.send];
};

let module Express = {
  type t;
  type middlewareT = (Request.t => Response.t => Next.t => unit [@bs]);
  external express : unit => t = "" [@@bs.module];
  external use : t => middlewareT => unit = "" [@@bs.send];
  external static : path::string => middlewareT = "" [@@bs.module "express"];
  external get : t => string => ('a => Response.t => unit [@bs]) => unit = "" [@@bs.send];
  external listen: t => int => unit = "" [@@bs.send];
};
```

```
11
12  let t = Obj.magic;
13
14  let module ReactRe = React;
15
16  let comp = React.createClass (
17    {
18      method getInitialState () :state => {"inner": 0};
19      method handleClick _ =>
20        React.setState this (fun (state: state) => {"inner": state##inner + 1});
21      method handleClick2 _ =>
22        React.setState this (fun (state: state) => {"inner": state##inner + 1});
23      method render () => {
24        /* let state: state = React.getState this; */
25        let props: props = React.getProps this;
26        switch (Js.Null_undefined.to_opt props##something) {
27        | None => print_endline "a"
28        | Some b => print_endline b
29        };
30        <div onClick={this##handleClick}> "hi------>" t props##children </div>
31      }
32    }
33    [@bs]
34  );
35
```


Test.re
FOLDERS
RebelExampleProject
_build
.jenga
node_modules
src
Test.re
.gitignore
2 * Welcome to Reason.
3 */
4 print_string "!!!!!!\n";
5 let msg = "Hello Reason!";
6 print_string msg;
7 print_newline ();
8 print_string "!!!!!!\n";
9 let f
v string => 'a failwith
c bool false
v int => float float
v int => float float_of_int
v string => float float_of_string
v float => float floor
v out_channel => unit flush
v unit => unit flush_all
v format6 'a 'b 'c 'd 'e 'f =… format_of_string
v float => (float, int) frexp
Raise exception [Failure] with the given string. Full description: string => 'a