Maybe *OCaml* Was the Friends We Made Along the Way

Maybe *OCaml* Was the Friends We Made Along the Way

A Journey of Growth in Software Engineering

This is a talk about

This is a talk about *professional growth*

This is a talk about *learning in public*

This is a talk about *mentorship*

This is a talk about *gratitude*

This is a talk about *becoming a better engineer*

This is a talk about *community*

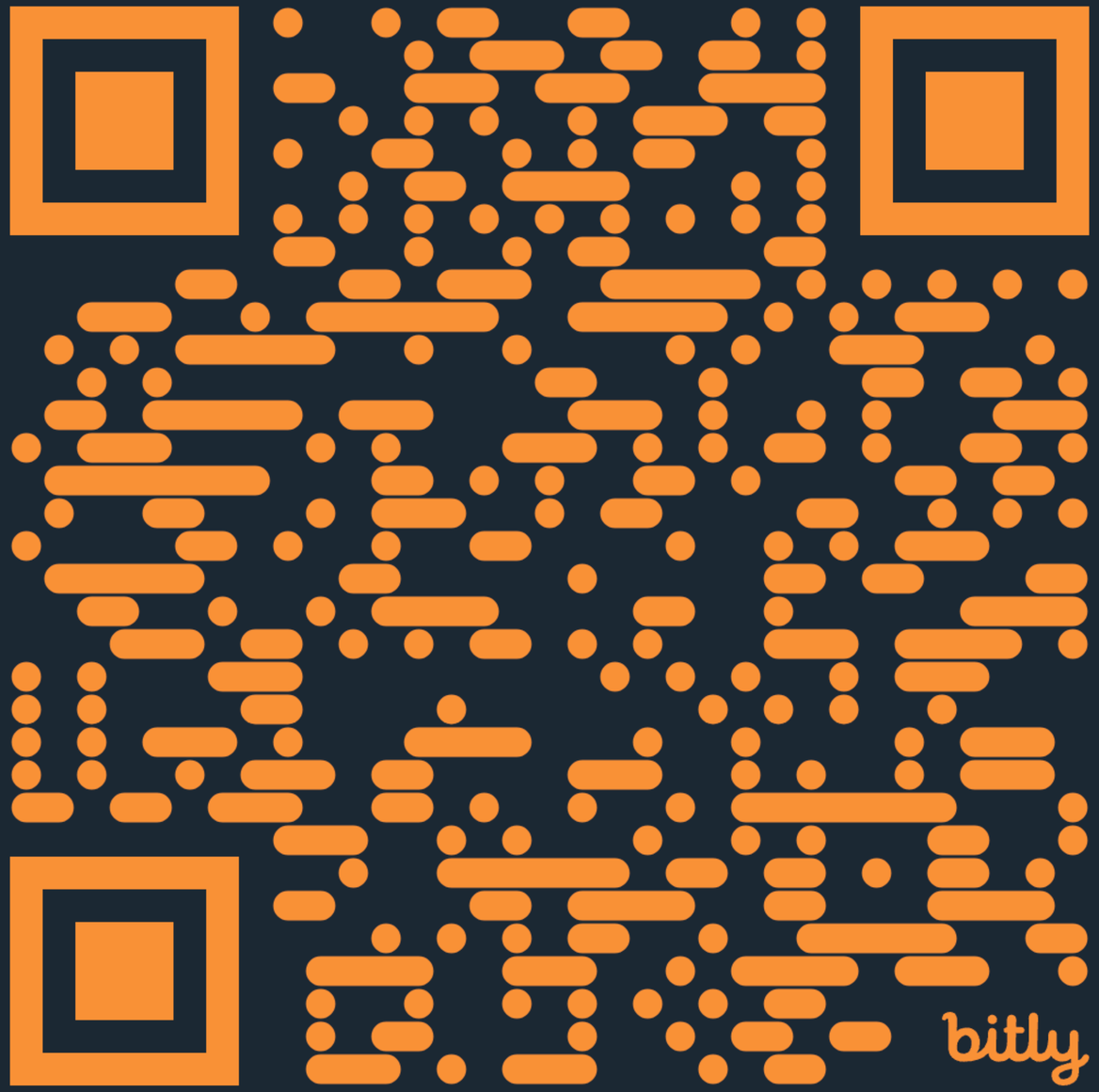
This is a talk about

OCaml

Dillon Mulroy

Software Engineer @ Vercel ▲

- twitch.tv/dmmulroy
- x.com/dillon_mulroy
- github.com/dmmulroy
- dillonis.online


bitly

My Journey to *OCaml*

Job #1

Job #1
JavaScript

Job #2

Job #2
JavaScript

Job #3

You guessed it

Job #3
JavaScript

Job #3

~~JavaScript~~

Job #3
TypeScript

Job #4


ANOTHER ONE

Job #4
TypeScript

Job #5

Job #5

OCaml

Job #6

Job #6
TypeScript

Job #6
*TypeScript**

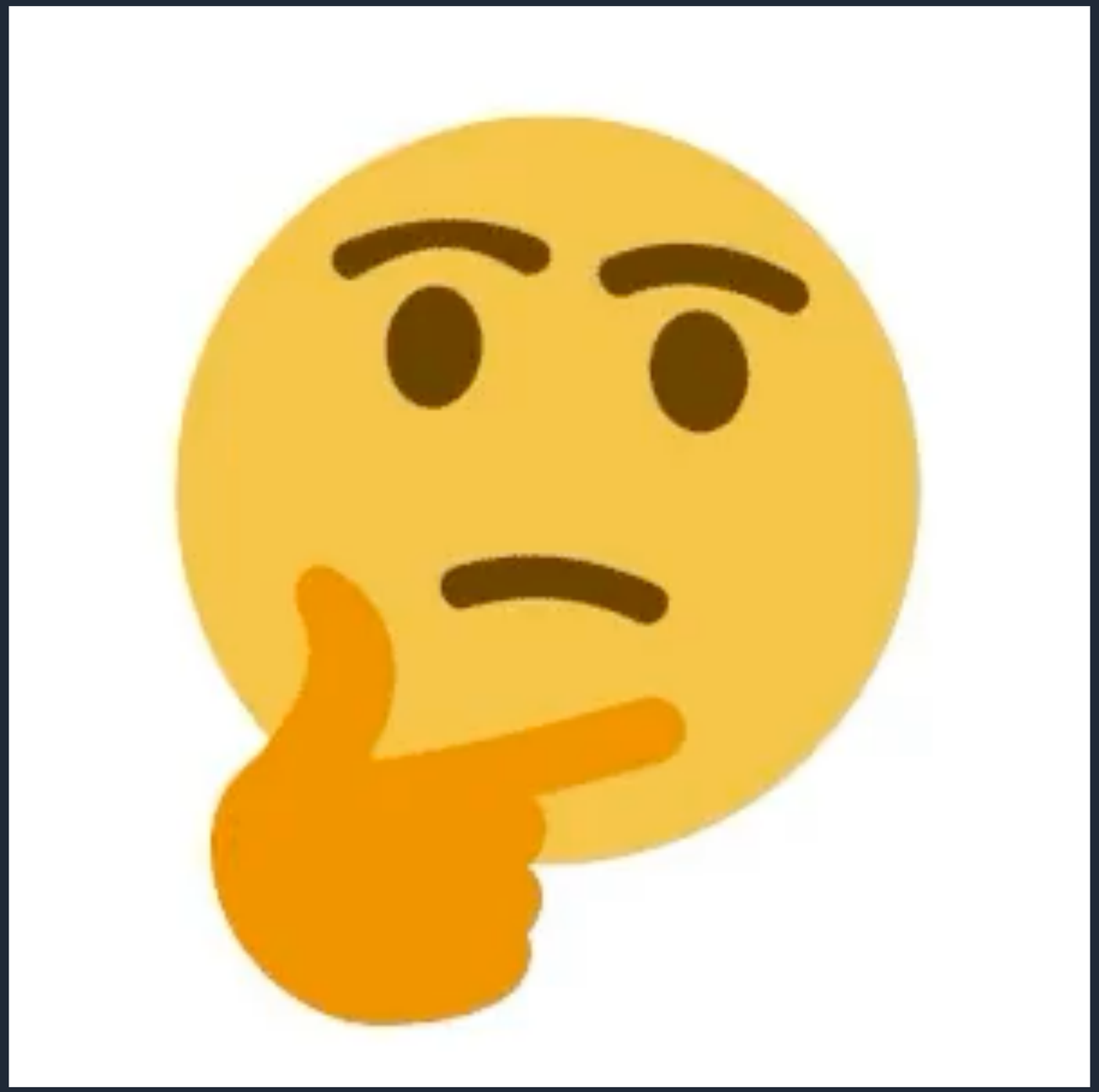
*for now 😈

My experience learning *OCaml*

Day 1


WHERE THE HELL AM I?

Day 2



Day 3

Maybe diving straight into the company codebase wasn’t the right approach 😅

OCaml Programming

Correct + Efficient + Beautiful

OCaml Programming: Correct + Efficient + Beautiful

A textbook on functional programming and data structures in OCaml, with an emphasis on semantics and software engineering. This book is the textbook for CS 3110 Data Structures and Functional Programming at Cornell University. A past title of this book was "Functional Programming in OCaml".

Fall 2024 Edition.

Videos. There are over 200 YouTube videos embedded in this book. They can be watched independently of reading the book. Start with this YouTube playlist.

Authors. This book is based on courses taught by Michael R. Clarkson, Robert L. Constable, Nate Foster, Michael D. George, Dan Grossman, Justin Hsu, Daniel P. Huttenlocher, Dexter Kozen, Greg Morrisett, Andrew C. Myers, Radu Rugina, and Ramin Zabih. Together they have created over 20 years worth of course notes and intellectual contributions. Teasing out who contributed what is, by now, not an easy task. The primary compiler and author of this work in its form as a unified textbook is Michael R. Clarkson, who as of the Fall 2021 edition was the author of about 40% of the words and code tokens.

Next

About This Book ›

By Michael R. Clarkson et al.

Day 4

After binging *CS-3110* I decide it’s time to actually build something

Enter
ocaml-playlist-transfer

Day 5

It *happens*

My 💡 moment

All at once things *click* and start falling into place

The Catalyst?

```
module Lwt = sig
  type 'a t

  val bind : 'a t -> ('a -> 'b t) -> 'b t

  val ( let* ) : 'a t -> ('a -> 'b t) -> 'b t
end
```

```
module Lwt = sig
  type 'a t

  val bind : 'a t -> ('a -> 'b t) -> 'b t

  val ( let* ) : 'a t -> ('a -> 'b t) -> 'b t
end


let* value = async_fn input
```

```
module Lwt = sig
  type 'a t

  val bind : 'a t -> ('a -> 'b t) -> 'b t

  val ( let* ) : 'a t -> ('a -> 'b t) -> 'b t
end
```

```
let* value = async_fn input
```

Wait a second…

I know this

```
module Lwt = sig
  type 'a t

  val bind : 'a t -> ('a -> 'b t) -> 'b t

  val ( let* ) : 'a t -> ('a -> 'b t) -> 'b t
end
```

```
let* value = async_fn input
```

```
let* value = async_fn input
```

```
let* value = async_fn input
```

```
let value = await async_fn(input)
```

OCaml *fundamentally* changed the way I think about & approach building software

OCaml changed the way I build software

(for the better)

- Type Driven Development (make illegal states unrepresentable)
- Errors as *values*
- “Modular” programming
- Simple abstractions focused on *one type*
- Immutability by default

An example in TypeScript

todo.ts

```ts
export interface Todo {
  readonly id: string;
  readonly name: string;
  readonly description: string;
  readonly status: "new" | "in-progress" | "completed";
  readonly createdAt: Date;
  readonly updatedAt: Date;
  readonly completedAt?: Date;
}
```

todo.ts

```ts
export const Todo = {
  make: (name: string, description: string): Todo => {
    const now = new Date();
    return {
      id: ulid(),
      name,
      description,
      status: "new",
      createdAt: now,
      updatedAt: now,
    };
  },

  start: (todo: Todo): Todo => ({ ...todo, status: "in-progress" }),

  complete: (todo: Todo): Todo => ({ ...todo, status: "completed" }),
};
```

todo.ts

```ts
export namespace Todo {
  export type t = Readonly<{
    id: string;
    name: string;
    description: string;
    status: 'new' | 'in-progress' | 'completed';
    createdAt: Date;
    updatedAt: Date;
    completedAt?: Date;
  }>;

  // ...excluded for brevity...

  export const complete = (todo: t): t => ({
    ...todo,
    status: 'completed',
  });
}

const todo: Todo.t = Todo.make('finish slides', 'stop procrastinating');
```

app.ts

```ts
const completedTodo = pipe(
  Todo.make("finish my slides", "im procrastinating"),
  Todo.start,
  Todo.complete,
);
```

Okay.
Back to my learning experience

During this time period of learning, I was seeking help and asking questions in/at places like

During this time period of learning, I was seeking help and asking questions in/at places like

The *ReasonML* & *OCaml* *Discord* Servers

During this time period of learning, I was seeking help and asking questions in/at places like

The *OCaml* Discuss Forums

During this time period of learning, I was seeking help and asking questions in/at places like

During this time period of learning, I was seeking help and asking questions in/at places like

 (the everything app)

During this time period of learning, I was seeking help and asking questions in/at places like

 (the everything app™)

During this time period of learning, I was seeking help and asking questions in/at places like

 ~~(the everything app™)~~

Twitter

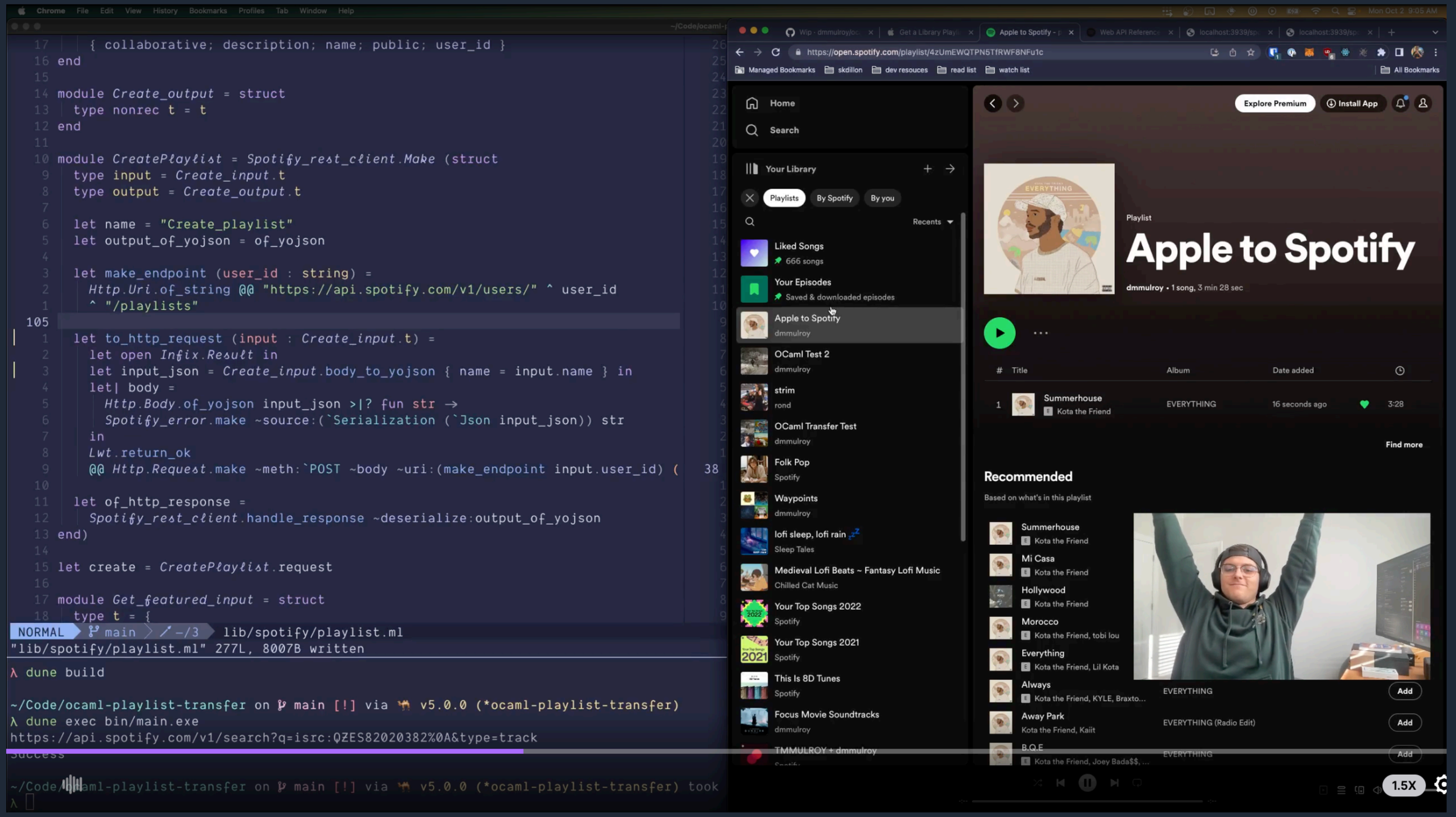
During this time period of learning, I was seeking help and asking questions in/at places like

Twitch

Something *stood out* to me

Every single person who helped me went so far *above & beyond* any help that I had received from other tech communities, and they were *eager* to do so

Also around this same time I started streaming on *Twitch*


{ collaborative; description; name; public; user_id }
end
module Create_output = struct
type nonrec t = t
end
module CreatePlaylist = Spotify_rest_client.Make (struct
type input = Create_input.t
type output = Create_output.t
let name = "Create_playlist"
let output_of_yojson = of_yojson
let make_endpoint (user_id : string) =
Http.Uri.of_string @@ "https://api.spotify.com/v1/users/" ^ user_id
^ "/playlists"
let to_http_request (input : Create_input.t) =
let open Infix.Result in
let input_json = Create_input.body_to_yojson { name = input.name } in
let| body =
Http.Body.of_yojson input_json >|? fun str ->
Spotify_error.make ~source:(`Serialization (`Json input_json)) str
in
Lwt.return_ok
@@ Http.Request.make ~meth:`POST ~body ~uri:(make_endpoint input.user_id) (
let of_http_response =
Spotify_rest_client.handle_response ~deserialize:output_of_yojson
end)
let create = CreatePlaylist.request
module Get_featured_input = struct
type t = {
NORMAL main -/3 lib/spotify/playlist.ml
"lib/spotify/playlist.ml" 277L, 8007B written
λ dune build
~/Code/ocaml-playlist-transfer on main [!] via v5.0.0 (*ocaml-playlist-transfer)
λ dune exec bin/main.exe
https://api.spotify.com/v1/search?q=isrc:QZES82020382%0A&type=track
Success
~/Code/ocaml-playlist-transfer on main [!] via v5.0.0 (*ocaml-playlist-transfer) took
λ
https://open.spotify.com/playlist/4zUmEWQTPN5TfRWF8NFu1c
Home
Search
Your Library
Playlists
By Spotify
By you
Recents
Liked Songs
666 songs
Your Episodes
Saved & downloaded episodes
Apple to Spotify
dmmulroy
OCaml Test 2
dmmulroy
strim
rond
OCaml Transfer Test
dmmulroy
Folk Pop
Spotify
Waypoints
dmmulroy
lofi sleep, lofi rain
Sleep Tales
Medieval Lofi Beats ~ Fantasy Lofi Music
Chilled Cat Music
Your Top Songs 2022
Spotify
Your Top Songs 2021
Spotify
This Is 8D Tunes
Spotify
Focus Movie Soundtracks
dmmulroy
TMMULROY + dmmulroy
Explore Premium
Install App
Playlist
Apple to Spotify
dmmulroy • 1 song, 3 min 28 sec
Title
Album
Date added
1 Summerhouse
Kota the Friend
EVERYTHING
16 seconds ago
3:28
Find more
Recommended
Based on what's in this playlist
Summerhouse
Kota the Friend
Mi Casa
Kota the Friend
Hollywood
Kota the Friend
Morocco
Kota the Friend, tobi lou
Everything
Kota the Friend, Lil Kota
Always
Kota the Friend, KYLE, Braxto...
EVERYTHING
Add
Away Park
Kota the Friend, Kaiit
EVERYTHING (Radio Edit)
Add
B.Q.E
Kota the Friend, Joey Bada$$, ...
EVERYTHING
Add
1.5X

Streaming on *Twitch* has allowed me to share my interests & meet amazing *friends*


SOUTHWEST
TOKYO


SOUTHWEST
TOKYO












Dillon Mulroy

Fun OCaml 2024









Dillon Mulroy

Fun OCaml 2024

Dillon Mulroy

Fun OCaml 2024

Dillon Mulroy

Dillon Mulroy

Streaming *OCaml*

Learning in public

- My first stream I *learned* something from chat
- It has helped grow my network and meet new *friends*
- I can share my interests & passions with liked minded people
- Best of all, I have *inspired others*

TL;DR
Learn in *public*

Looking to the *future*…

I am *very* excited about the work that is happening & where *OCaml* is going

I am *very* excited about the work that is happening & where *OCaml* is ~~going~~

I am *very* excited about the work that is happening & where *OCaml* is ~~going~~ *growing*

Thank you