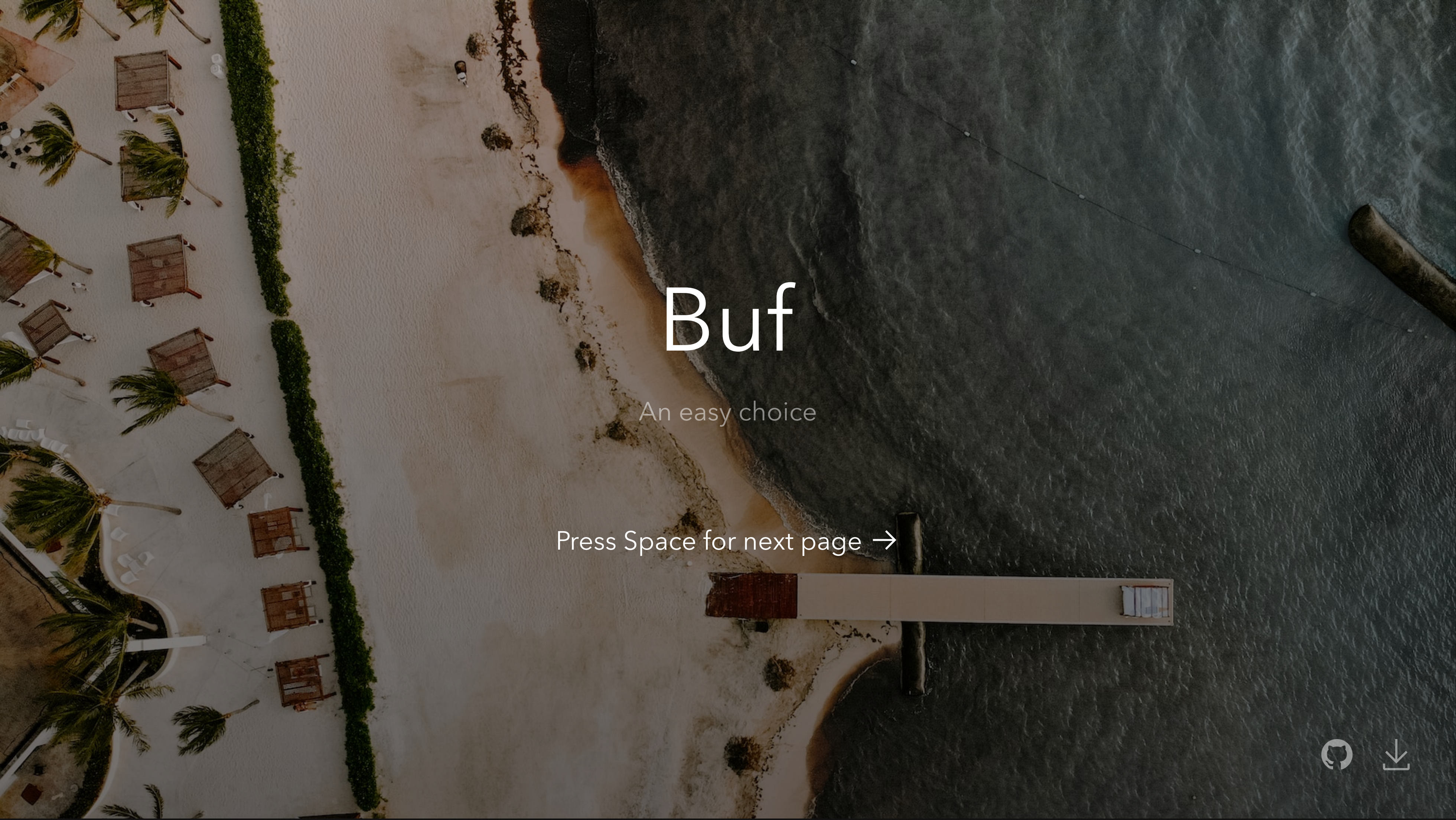

Buf

An easy choice

Press Space for next page →

Kevin Chen

Software Engineer at Teaching Strategies.

Formerly at The Drivers Coop, Care.com, and IrisVR.

Fan of chess, racket sports, skiing, guitar.

kevinmichaelchen

kevinmchen

kevin.chen@teachingstrategies.com

Table of Content

What is a Buf?

Buf’s tagline is:

"Building a better way to work with Protocol Buffers"

What is a Protocol Buffer?

Protocol buffers (protobufs) are Google's

- language-neutral,
- platform-neutral,
- extensible mechanism

for **serializing structured data**.

- Think XML, but smaller, faster, and simpler.
- You **define** how you want your data to be structured **once**, then you can use special **generated source code** to **easily write** and read your structured data to and from a variety of data streams and using a **variety of languages**.

The world in 2019

2019 - protoc is hard

- protoc is the Protobuf compilation tool.
- It looks easy to work with.

```
$ brew install protobuf
$ protoc --version  # Ensure compiler version is 3+
$ protoc -I=$SRC_DIR --go_out=$DST_DIR $SRC_DIR/addressbook.proto
```

2019 - protoc is hard

- protoc is the Protobuf compilation tool.
- It looks easy to work with.

```
$ brew install protobuf
$ protoc --version  # Ensure compiler version is 3+
$ protoc -I=$SRC_DIR --go_out=$DST_DIR $SRC_DIR/addressbook.proto
```

- But is it?
  - Lots of flags: unclear what they do; easy to make mistakes
  - Lots of plugins: steep learning curve
  - What about more complex directory structures?
  - Bash scripts would start to accumulate
  - It soon becomes arcane

2019 - no good way of sharing

- There is no registry for protobufs in 2019.
    - If Service A and Service B want to use a common protobuf…
    - They have to manually copy/paste and maintain 2 separate versions. 👎
    - Imagine trying to code in JS/TS without NPM, or in Golang without Go modules. 🤯

2019 - no good way of sharing

- There is no registry for protobufs in 2019.
    - If Service A and Service B want to use a common protobuf…
    - They have to manually copy/paste and maintain 2 separate versions. 👎
    - Imagine trying to code in JS/TS without NPM, or in Golang without Go modules. 🤯
- Stub distribution
    - Once you auto-generate code from protobufs, how do you distribute/share those stubs?
    - Do you centralize all your protos in a monorepo?
    - Do you have all your clients/microservices using ``protoc``?
    - Do you have to worry about publishing to package registries (Maven/Gradle, NPM, Go modules, etc)?

2019 - other tooling is missing

- no linter
- no formatter
- no build-time compilation checkers
- no breaking-change detection
    - outages abound
- no Postman-esque tool
- not debuggable in the browser

2020 - Enter Buf

Problems solved

- No more fiddling with protoc. We have the CLI now.
- The Buf Schema Registry (BSR) is a powerful tool.
  - No more copy/pasting protobufs. We can now pull dependencies.
  - We can also generate stubs remotely.
  - When you push to the BSR, you can immediately download private auto-generated stubs for Go or Typescript.

```
$ go get go.buf.build/grpc/go/orgname/licenseapis
```

- We have linting, formatting, and breaking change detection.
- Auto-generated docs via BSR.
- We have a Postman-esque tool via Buf Studio.
- We have a much better in-the-browser dev experience thanks to Connect.

finito