BUOYANT

The creators of LINKERD

Writing Service Mesh Controllers in Rust

Eliza Weisman, Linkerd maintainer

Eliza Weisman

Software Engineer, Buoyant

@mycoliza

@hawkw

@eliza

BUOYANT

Rust is fast and memory safe, all without garbage collection

BUOYANT

LINKERD

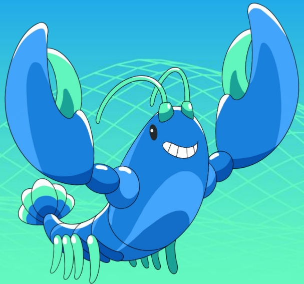
Ultralight, ultrafast, security-first **service mesh** for Kubernetes.

★ Created by **Buoyant**

★ **Goals:** secure, efficient, fast

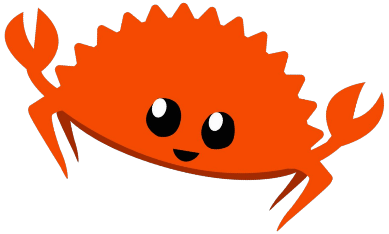
★ **Data plane:** proxies application traffic

★ **Control plane:** tells the proxies what to do



Linkerd + Rust

- **Pure Rust data plane** since the release of Linkerd 2
- First **Rust control plane component** released in Linkerd 2.11



BUOYANT

Rust was **the only choice** for the **proxy**.

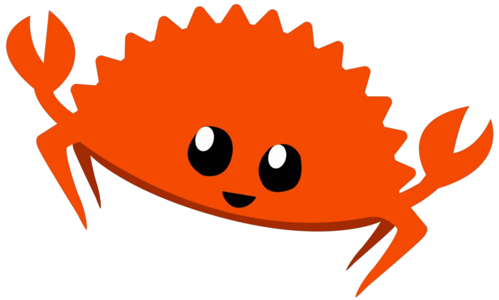
We **didn't want GC pauses**…

…but we **didn't want C++ CVEs**, either.

BUOYANT

Rust isn't the only choice for the control plane, but it is a good one (and we like it).

Rust has great language features
dev tools
libraries



BUOYANT

Linkerd's control plane watches Kubernetes resources and serves gRPC APIs for the proxies

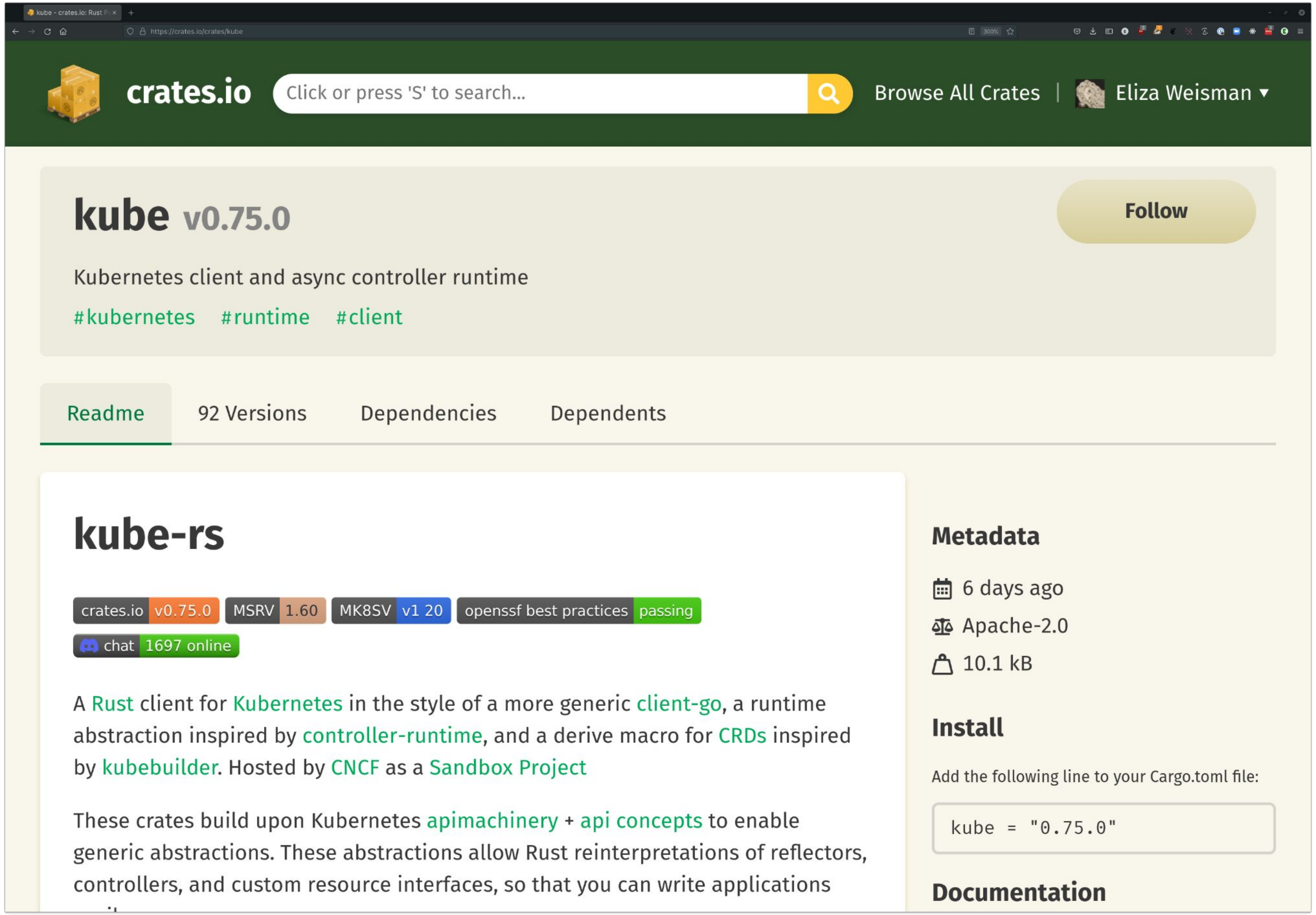
BUOYANT

The policy controller indexes policy resources and associates them with ports on pods.

BUOYANT

Writing controllers in Rust means talking to Kubernetes, which means bindings for the Kubernetes API.

BUOYANT



kube v0.75.0

Kubernetes client and async controller runtime

#kubernetes #runtime #client

kube-rs

A Rust client for Kubernetes in the style of a more generic client-go, a runtime abstraction inspired by controller-runtime, and a derive macro for CRDs inspired by kubebuilder. Hosted by CNCF as a Sandbox Project

These crates build upon Kubernetes apimachinery + api concepts to enable generic abstractions. These abstractions allow Rust reinterpretations of reflectors, controllers, and custom resource interfaces, so that you can write applications

Metadata

- 6 days ago
- Apache-2.0
- 10.1 kB

Install

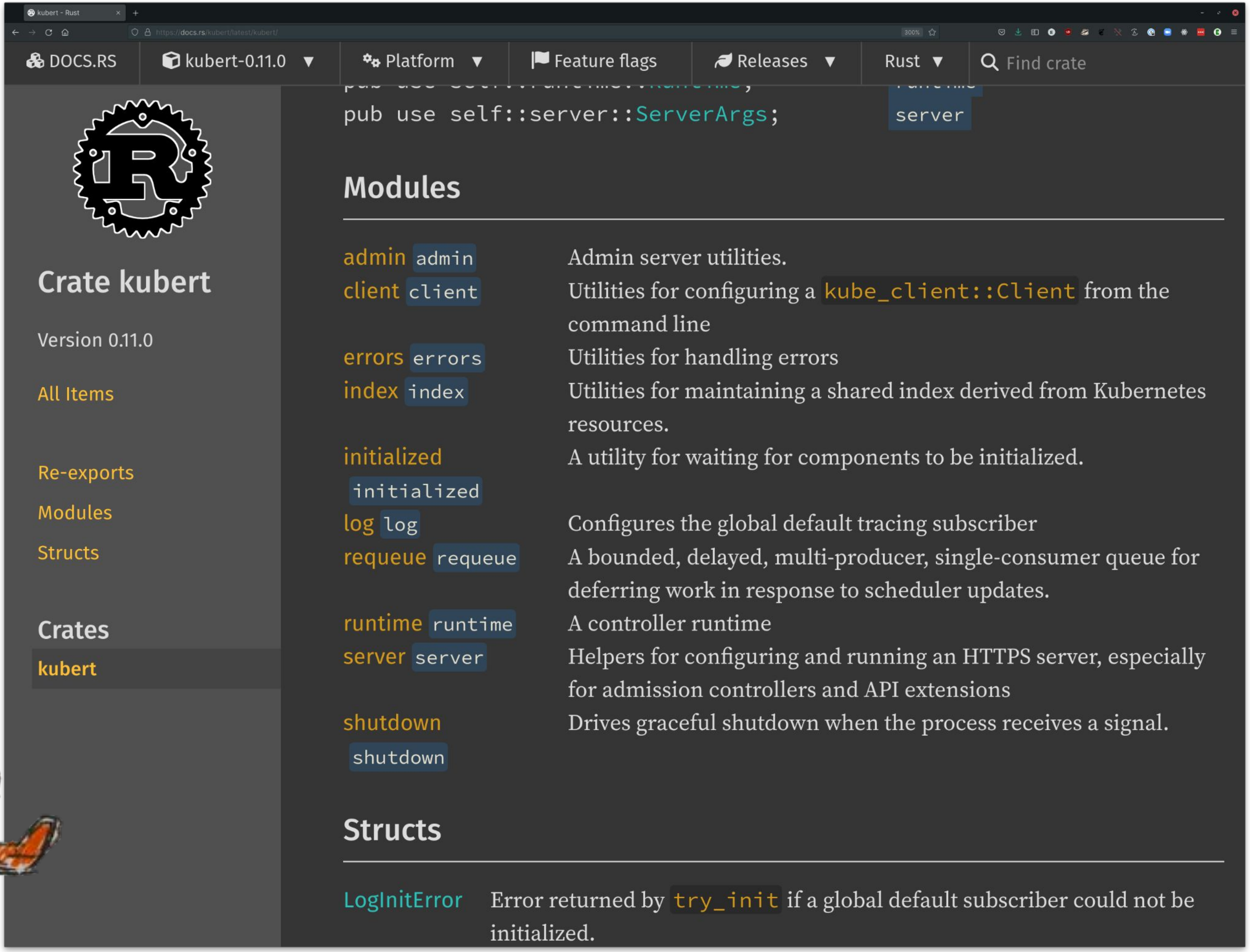
Add the following line to your Cargo.toml file:

```
kube = "0.75.0"
```

Documentation

BUOYANT

kube + rt = kubert



Crate kubert

Version 0.11.0

All Items

Re-exports

Modules

Structs

Crates

kubert

```
pub use self::server::ServerArgs;
```

Modules

admin `admin`	Admin server utilities.
client `client`	Utilities for configuring a `kube_client::Client` from the command line
errors `errors`	Utilities for handling errors
index `index`	Utilities for maintaining a shared index derived from Kubernetes resources.
initialized `initialized`	A utility for waiting for components to be initialized.
log `log`	Configures the global default tracing subscriber
requeue `requeue`	A bounded, delayed, multi-producer, single-consumer queue for deferring work in response to scheduler updates.
runtime `runtime`	A controller runtime
server `server`	Helpers for configuring and running an HTTPS server, especially for admission controllers and API extensions
shutdown `shutdown`	Drives graceful shutdown when the process receives a signal.

Structs

LogInitError	Error returned by `try_init` if a global default subscriber could not be initialized.

BUOYANT

```rust
pub trait IndexNamespacedResource<T> {
    fn apply(&mut self, resource: T);
    fn delete(&mut self, namespace: String, name: String);
    fn reset(
        &mut self,
        resources: Vec<T>,
        removed: HashMap<String, HashSet<String>>,
    );
}
```

BUOYANT

```yaml
apiVersion: policy.linkerd.io/v1beta1
kind: Server
metadata:
  namespace: linkerd-viz
  name: admin
spec:
  podSelector:
    matchLabels:
      linkerd.io/extension: viz
  port: admin-http
  proxyProtocol: HTTP/1
```

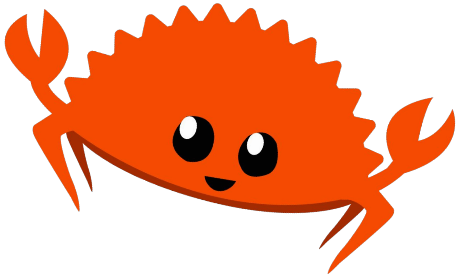
```rust
#[derive(Clone, Debug, PartialEq, Eq)]
#[derive(kube::CustomResource)]
#[derive(Deserialize, Serialize, JsonSchema)]
#[kube(
    group = "policy.linkerd.io",
    version = "v1beta1",
    kind = "Server",
    namespaced
)]
#[serde(rename_all = "camelCase")]
pub struct ServerSpec {
    pub pod_selector: labels::Selector,
    pub port: Port,
    pub proxy_protocol: Option<ProxyProtocol>,
}
```

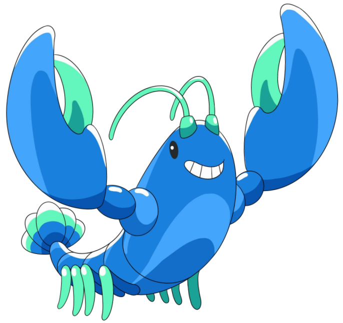
BUOYANT

```rust
#[tokio::main]
async fn main() -> Result<(), Box<dyn Error>> {
    // Build the Kubert runtime (Kubert also has arg-parsing helpers)
    let Args { client } = Args::parse();
    let runtime = kubert::Runtime::builder()
        .with_client(client)
        .build()
        .await?;

    // Start indexing the Server CRD
    let index = Index::default();
    let servers = runtime.watch_all::<Server>(ListParams::default());
    tokio::spawn(kubert::index::namespaced(index.clone(), servers));

    // Do more stuff, like spawning gRPC servers…

    runtime.run().await.map_err(Into::into)
}
```

BUOYANT

https://github.com/linkerd/linkerd2/tree/main/policy-controller





BUOYANT

BUOYANT'S SERVICEMESH ACADEMY

The creators of LINKERD

Monthly hands-on, engineer-focused training from the creators of the service mesh

Nov 11-17: What really happens at startup: a deep dive into Linkerd, init containers, CNI plugins, and more

SIGN UP TODAY!

buoyant.io/sma

Fully managed LINKERD on any Kubernetes cluster

Buoyant Cloud automated upgrades, data plane version tracking, mesh health alerts, and much, much more.

BOOK A DEMO
buoyant.io/demo

BUOYANT

The creators of LINKERD

Thanks so much!

Eliza Weisman
Software Engineer, Buoyant
eliza@buoyant.io

@BuoyantIO

buoyant.io