HALF A DECADE OF RUST IN MOBILE GAMES

BY STEPHAN DILLY

WHO AM I?

- Stephan Dilly
- Worked for Ubisoft, InnoGames, Kolibri, GitButler & Kraken
- Founded Gameroasters (sold to CRATR.games)
- Founded Rustunit.com
- Created GitUI
- Organiser of the Bevy Meetup

GOALS FOR TODAY

- How did using Rust changed over time
- Some war stories
- Where are we today?
- Whats does the future hold?

OUTLINE

- Why Rust?
- Shared Rust Core Library
- Pure Rust Backends
- Hybrid Clients 2.0
- Modular Gamebackend
- Fullstack Rust Mobile Games

WHY RUST?

DREAM OF FULLSTACK

- Share critical logic with the backend
- Performance critical AI calculation in the Client
- Easily exposable via c-ffi
- No GC
- “If it builds it works”
- Dare to dream of Fullstack Mobile Dev

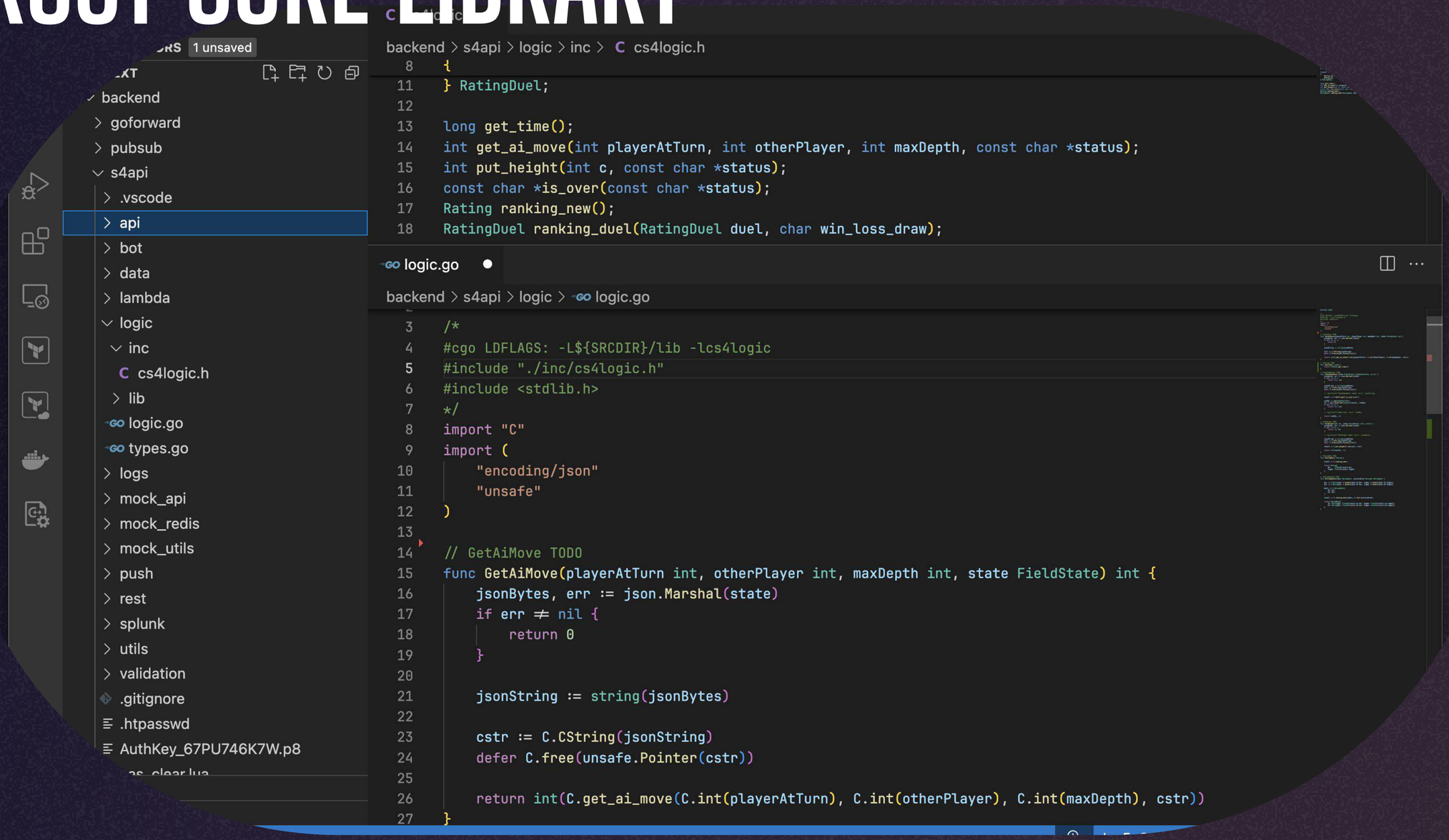
SHARED RUST CORE LIBRARY

GO BACKEND MEETS RUST

- calling C from go is horrible
- stringified json c-ffi

SHARED RUST CORE LIBRARY

GO C-FFI



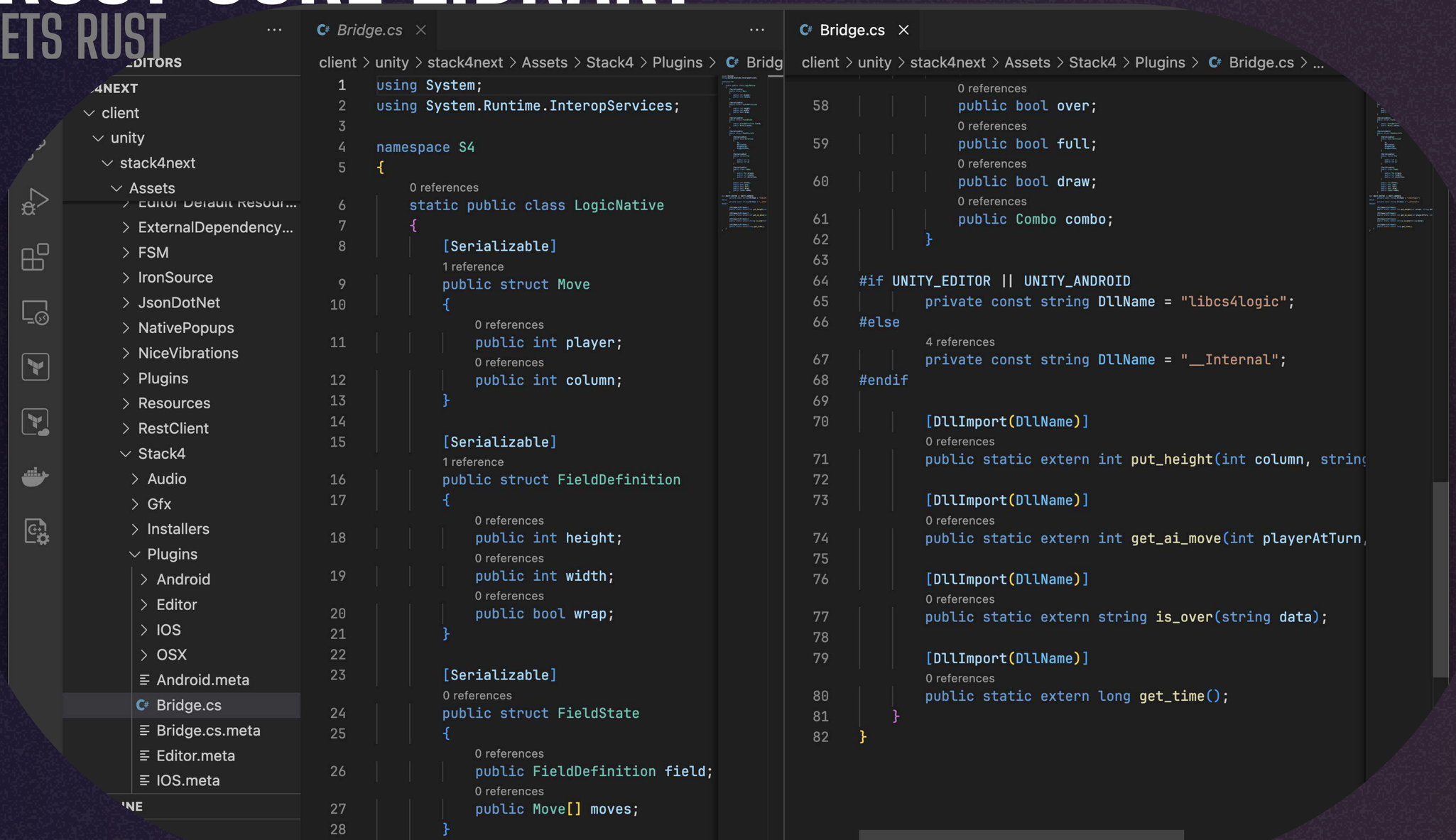
SHARED RUST CORE LIBRARY

UNITY CLIENT MEETS RUST

- c-ffi is not too bad from C#
- cross compilation for iOS & Android rocks
- iteration is slow in Unity (reopen editor)
- stringified json c-ffi

SHARED RUST CORE LIBRARY

UNITY CLIENT MEETS RUST



SHARED RUST CORE LIBRARY

CONCLUSION

- verbose and slow data passing across boundaries
- go c-ffi is slow
- hard to maintain
- go in the backend viable but adds another tech
- go error handling and package manager sucked at the time
- Slow Unity / C# json parsing

PURE RUST BACKEND

WHY NOW?

- Async Await stabilised
- More and more driver support for Rust (MongoDB)
- Client Project Opportunity to use Rust
- Warp + CosmosDB (via MongoDB)

PURE RUST BACKEND

WHAT COULD GO WRONG?

- Server hit by 1k players per second
- Crashed every few minutes
- Official MongoDB driver bug
- Connection Pool handling was broken

PURE RUST BACKEND

MONGODB DRIVER BUG

Decrement connection count when connection is dropped due to...

nished operation

main

v2.8.2 ... v1.--ignore

Stephan Dilly authored and **saghm** committed on Aug 19, 2020 — 1 parent ffa0b66 commit 4d4e007

Showing **2 changed files** with **33 additions** and **21 deletions**.

Whitespace | Ignore whitespace | Split | Unified

Filter changed files

- src/cmap
  - conn
    - mod.rs
  - mod.rs

46 src/cmap/conn/mod.rs

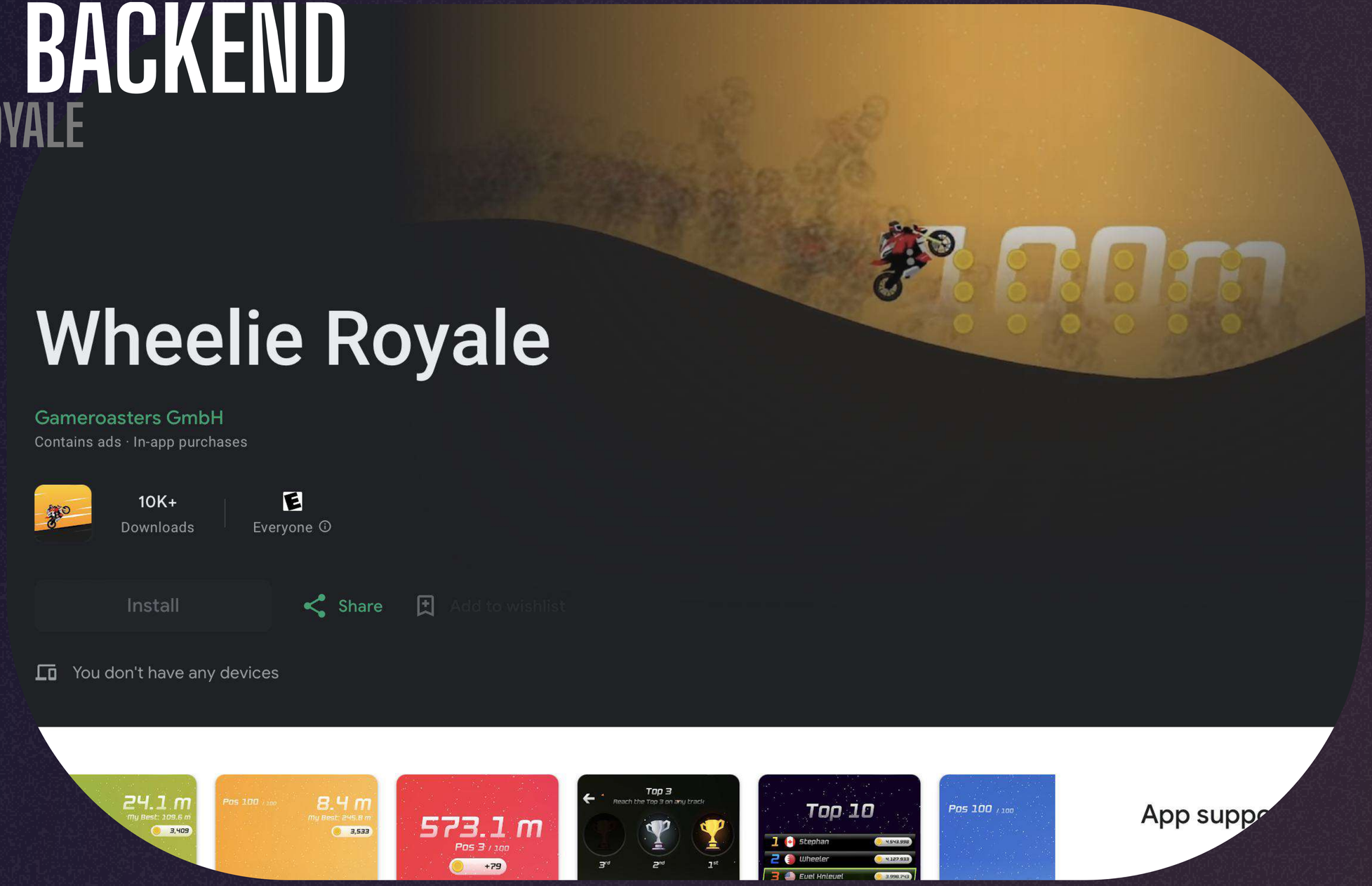
8 src/cmap/mod.rs

```
@@ -300,6 +300,14 @@ impl ConnectionPoolInner {
300 300         }
301 301     }
302 302
    303 +   async fn dropped(&self, conn: Connection) {
    304 +       let mut connection_manager = self.connection_manager.lock().await;
    305 +
    306 +       connection_manager.close_connection(conn, ConnectionClosedReason::Dropped);
    307 +
    308 +       self.wait_queue.wake_front();
    309 +   }
    310 +
303 311     async fn check_in(&self, mut conn: Connection) {
304 312         self.emit_event(|handler| {
305 313             handler.handle_connection_checked_in_event(conn.checked_in_event());
```

HTTPS://GITHUB.COM/MONGODB/MONGO-RUST-DRIVER/PULL/234

PURE RUST BACKEND

USAGE IN WHEELIE ROYALE



HYBRID CLIENTS 2.0

PROTOBUF TO THE RESCUE

- c-ffi down to a single function
- protobuf instead instead of json
- code generation for Rust and C#
- single memory buffer, no more GC hick-ups
- simple code sharing with backend

HYBRID CLIENTS 2.0

CODE #1

ity > stack4 > Assets > _stack4 > Scripts > Native > Bridge.cs > ...

```
using GR.S4.Controller.Backend;
using UnityEngine;

namespace GR.S4.Native
{
    2 references
    static class Native
    {
#if UNITY_EDITOR || UNITY_ANDROID
        private const string DllName = "libs4clogic";
#else
        2 references
        private const string DllName = "__Internal";
#endif

        [DllImport(DllName)]
        1 reference
        public static extern long get_time();

        [DllImport(DllName)]
        1 reference
        public static extern bool logic_call(
            byte[] buffer, long bufferSize, long inputSize, ref long outputSize);
    }


    1 reference
```

HYBRID CLIENTS 2.0

CODE #2

```
option csharp_namespace = "GR.S4.Controller.Backe

message LogicCalls {
  message LogicGame {
    Game.Field field = 1;
    GameState state = 2;
  }

  message GameOver {}

  message AiMove {
    GameState.PlayerIndex player = 1;
    uint32 depth = 2;
  }

  message TurnInfo {
    message Move {
      GameState.PlayerIndex player = 1;
      UtcTime time = 2;
    }

    UtcTime now = 1;
    UtcTime startTime = 2;
    GameState.PlayerIndex startPlayer = 3;
    Move lastMove = 4;
    Duration moveDuration = 5;
  }

  message ComboQuery {
    uint32 column = 1;
    uint32 height = 2;
  }

  oneof calls {
    LogicGame setState = 1;
    GameState.Move canMove = 2;
    AiMove aiMove = 3;
```

```
}

message GameOver {
  message Draw {}

  message WinnerCombos {
    repeated CubePos positions = 1;
  }

  oneof kind {
    Draw draw = 1;
    WinnerCombos combos = 2;
  }
}

message TurnInfo {
  GameState.PlayerIndex player = 1;
  UtcTime start = 2;
}

message ComboQuery {
  message ComboPos {
    CubePos position = 1;
    bool missing = 2;
  }

  message Combo {
    repeated ComboPos positions = 1;
  }

  repeated Combo combos = 1;
}

oneof calls {
  GetCubePlacement getCubePlacement = 1;
  CanMove canMove = 2;
  AiMove aiMove = 3;
```

HYBRID CLIENTS 2.0

SHOWCASE: TOWERRANGERS

Tower Rangers

Idle tower defense meets real-time duels

MODULAR BACKEND

ATLAS SERVER COMPONENTS

- Authentication, SSO (FB, SIWA, Google)
- IAP Validation
- Push Notifications
- PubSub Realtime matches
- AB Testing
- Friends Lists, Leaderboards
- Tracking

MODULAR BACKEND

ATLAS STACK

- WARP
- API crates for 3rd parties (FB,SIWA)
- reusable, composable WARP layer
- static typed DI (using HList)
- find opensource at:
  https://github.com/gameroasters/atlasserver

MODULAR BACKEND

EXAMPLE: HLIST

```
    fn sculpt<Ts, Indices>(                                                    so
    self
) -> (Ts, <HCons<Head, Tail> as Sculptor<Ts, Indices>>::Remainder)
where
    HCons<Head, Tail>: Sculptor<Ts, Indices>,
```

Consume the current HList and return an HList with the requested shape.

`sculpt` allows us to extract/reshape/sculpt the current HList into another shape, provided that the requested shape's types are are contained within the current HList.

The `Indices` type parameter allows the compiler to figure out that `Ts` and `Self` can be morphed into each other.

Examples

```
use frunk_core::{hlist, HList};

let h = hlist![9000, "joe", 41f32, true];
let (reshaped, remainder): (HList![f32, i32, &str], _) = h.sculpt();
assert_eq!(reshaped, hlist![41f32, 9000, "joe"]);
assert_eq!(remainder, hlist![true]);
```

```
entation
struct S4Module {}

impl CustomModule for S4Module {
    type Resources = Hlist![
        Arc<PlayerStateResource>,
        Arc<TrackingResource>,
        Arc<UserLoginResource>,
        Arc<RankingResource>,
        Arc<PushNotificationResource>,
        Arc<S4FcmSendResource>,
        Arc<SsoResource>,
        Arc<WofResource>,
        Arc<RealtimeSenderArenaResource>,
        Arc<FriendsResource>,
        Arc<SpectateResource>,
        Arc<AdsResource>,
        Arc<LobbyResource>,
        Arc<GameSpawnerResource>,
        Arc<CheatsResource>,
        Arc<S4BroadcastResource>,
        Arc<ShopResource>
    ];
```

MODULAR BACKEND

EXAMPLE: REALITY

```
server.get_server_resources().sculpt();
let (
    playerstateres: Arc<PlayerStateResource>,
    (
        tracking_resource: Arc<TrackingResource>,
        (
            user_resource: Arc<UserLoginResource>,
            (
                ranking_resource: Arc<RankingResource>,
                (
                    _,
                    (
                        s4_push_res: Arc<S4FcmSendResource>,
                        (
                            sso_res: Arc<SsoResource>,
                            (
                                wof_res: Arc<WofResource>,
                                (
                                    realtime: Arc<RealtimeSenderArenaResource>,
                                    (
                                        friends: Arc<FriendsResource>,
                                        (
                                            spectate: Arc<SpectateResource>,
                                            (
                                                ads: Arc<AdsResource>,
                                                (
                                                    lobby_res: Arc<LobbyResource>,
                                                    (
                                                        game_spawner: Arc<GameSpawnerResource>,
                                                        (
                                                            cheats: Arc<CheatsResource>,
                                                            (
                                                                broadcast: Arc<BroadcastResource<BroadcastMessage>>,
                                                                shop: Arc<ShopResource>,
                                                            ),
                                                        ),
                                                    ),
                                                ),
                                            ),
                                        ),
                                    ),
                                ),
                            ),
                        ),
                    ),
                ),
            ),
        ),
    ),
) =
```

MODULAR BACKEND

EXAMPLE: WARP TYPE MADNESS

```
- CustomModule for S4Module {
 fn create_filter<S: atlasserver::ModuleResources<Self>>(
                >()) And<And<And<And<And<And<And<…, …>, …>, …>, …
                .and(friends) And<And<And<And<And<And<And<…, …>, …>, …>, …
                .and_then(fun: facebook_friends_filter_fn);

331
332         let spectate_filter: AndThen<And<And<And<And<And<…, …>, …>, …>, …>…> = warp::path!("s4" / "spectate")
333             .and(userlogin::session_filter(user_resource.clone())) And<And<And<And<impl Filter<Extract = …, Er…
334             .and(warp::post()) And<And<And<And<And<impl Filter<Extract = ……
335             .and(pbwarp::protobuf_body::<schema::SpectateRequest>()) And<And<And<And<And<And<impl Filter<Extract…
336             .and(spectate) And<And<And<And<And<And<And<…, …>, …>, …>, …
337             .and_then(fun: spectate_filter_fn);
338
339         let wof: BoxedFilter<(Box<dyn Reply>,)> = create_filters_wof(wof_res, &user_resource);
340         let lobby: BoxedFilter<(Box<dyn Reply>,)> = create_filters_lobby(lobby_res, &user_resource);
341         let siwa: BoxedFilter<(Box<dyn Reply>,)> = create_filters_siwa(&user_resource, sso_res);
342
343         let tracking_filter: AndThen<And<And<And<And<And<…, …>, …>, …>, …>…> = warp::path!("tracking")
344             .and(userlogin::session_filter(user_resource)) And<And<And<impl Filter<Extract = …, Error …
345             .and(warp::post()) And<And<And<And<impl Filter<Extract = …, Er…
346             .and(pbwarp::protobuf_body::<schema::TrackingRequest>()) And<And<And<And<And<impl Filter<Extract = ……
347             .and(warp::any().map(fun: move || tracking_resource.clone())) And<And<And<And<And<And<impl Filter<Ext…
348             .and_then(fun: tracking_filter_fn);
349
350         let time_filter: Map<And<And<impl Filter<Extract = …, Erro… = warp::path!("time").map(time_filter_fn);
351
352         upgrade_item_filter AndThen<And<And<And<And<And<…, …>, …>, …>, …
353             .or(lobby) Or<AndThen<And<And<And<And<…, …>, …>, …>, ……
354             .or(wof) Or<Or<AndThen<And<And<And<…, …>, …>, …>, …>…
355             .or(siwa) Or<Or<Or<AndThen<And<And<…, …>, …>, …>, …>,…
356             .or(get_kotd_filter) Or<Or<Or<Or<AndThen<And<And<…, …>, …>, …>, …
357             .or(ranking_filter) Or<Or<Or<Or<Or<AndThen<And<…, …>, …>, …>, ……
358             .or(select_cube_filter) Or<Or<Or<Or<Or<Or<AndThen<…, …>, …>, …>, …>…
359             .or(matchmaking_filter) Or<Or<Or<Or<Or<Or<Or<AndThen<…, …>, …>, …>,…
360             .or(get_state_filter) Or<Or<Or<Or<Or<Or<Or<Or<AndThen<…, …>, …>, …
361             .or(game_get_filter) Or<Or<Or<Or<Or<Or<Or<Or<Or<…, …>, …>, …>, ……
362             .or(game_resign_filter) Or<Or<Or<Or<Or<Or<Or<Or<Or<…, …>, …>, …>, ……
'63             .or(game_leave_filter) Or<Or<Or<Or<Or<Or<Or<Or<Or<…, …>, …>, …>, ……
                .or(game_move_filter) Or<Or<Or<Or<Or<Or<Or<Or<Or<…, …>, …>, …>, ……
                .or(game_rematch_filter) Or<Or<Or<Or<Or<Or<Or<Or<Or<…, …>, …>, …>, ……
                .or(claim_reward_filter) Or<Or<Or<Or<Or<Or<Or<Or<Or<…, …>, …>, …>, ……
                .or(get_state_other_filter) Or<Or<Or<Or<Or<Or<Or<Or<Or<…, …>, …>, …>, ……
```

MODULAR BACKEND

SHOWCASE: STACK4

FULLSTACK RUST MOBILE GAMES

HELLO BEVY

- ECS-first, multithreaded, fast
- Modular using Plugins
- Cross Platform: win,mac,linux,ios,android,web...
- Free and OpenSource
- Huge community (19k on Discord)

FULLSTACK RUST MOBILE GAMES

ZOOLITAIRE

FULLSTACK RUST MOBILE GAMES

BACKEND NOW USING AXIUM

- Axium is more ergonomic
- Easier to modularise using Tower
- HLIST-free DI using Extractors
- Reuses components of Atlas

FULLSTACK RUST MOBILE GAMES

BACKEND NOW USING AXIUM

```rust
    Path((id: String, secret: String, question_id: i64)): Path<(String, String, i64)>,
    State(app: Arc<App>): State<SharedApp>,
) -> std::result::Result<impl IntoResponse, InternalError> {
    tracing::info!("mod_get_question");

    Ok(Json(app.get_question(id, Some(secret), question_id).await?))
}

#[instrument(skip(app))]
pub async fn get_question(
    Path((id: String, question_id: i64)): Path<(String, i64)>,
    State(app: Arc<App>): State<SharedApp>,
) -> std::result::Result<impl IntoResponse, InternalError> {
    tracing::info!("get_question");

    Ok(Json(app.get_question(id, None, question_id).await?))
}

#[instrument(skip(app))]
pub async fn mod_edit_question(
    Path((id: String, secret: String, question_id: i64)): Path<(String, String, i64)>,
    State(app: Arc<App>): State<SharedApp>,
    Json(payload: ModQuestion): Json<shared::ModQuestion>,
) -> std::result::Result<impl IntoResponse, InternalError> {
    tracing::info!("mod_edit_question");

    Ok(Json(
        app.mod_edit_question(id, secret, question_id, payload)
            .await?,
    ))
```

FULLSTACK RUST MOBILE GAMES

BACKEND NOW USING AXIUM

```
275
276     let event_routes = Router::new()
277         .route("/:id", get(handle::getevent_handler))
278         .route("/:id/pwd", post(handle::set_event_password))
279         .route("/add", post(handle::addevent_handler))
280         .route("/editlike/:id", post(handle::editlike_handler))
281         .route("/addquestion/:id", post(handle::addquestion_handler))
282         .route("/question/:id/:question_id", get(handle::get_question));
283
284     #[rustfmt::skip]
285     let mod_routes = Router::new()
286         .route("/:id/:secret", get(handle::mod_get_event))
287         .route("/upgrade/:id/:secret", get(handle::mod_premium_upgrade))
288         .route("/capture/:id/:order", get(handle::mod_premium_capture))
289         .route("/delete/:id/:secret", get(handle::mod_delete_event))
290         .route("/question/:id/:secret/:question_id", get(handle::mod_get_question))
291         .route("/questionmod/:id/:secret/:question_id", post(handle::mod_edit_question))
292         .route("/:id/:secret", post(handle::mod_edit_event));
293
294     #[rustfmt::skip]
295     let router = Router::new()
296         .route("/api/ping", get(handle::ping_handler))
297         .route("/api/version", get(handle::version_handler))
298         .route("/api/error", get(handle::error_handler))
299         .route("/api/payment/stripe/webhook", post(stripe_webhooks::handle_webhook))
300         .route("/push/:id", get(push_handler))
301         .nest("/api/event", event_routes)
302         .nest("/api/mod/event", mod_routes)
303         .nest("/api/admin", admin_routes)
304         .layer(auth_layer)
305         .layer(session_layer)
306         .layer(SetSensitiveRequestHeadersLayer::new(once(header::COOKIE)))
307         .layer(SentryHttpLayer::with_transaction())
308         .layer(NewSentryLayer::new_from_top())
309         .layer(TraceLayer::new_for_http())
310         .layer(setup_cors())
311         .with_state(Arc::clone(&app));
```

FULLSTACK RUST MOBILE GAMES

ZOOLITAIRE: OPENSOURCE

- bevy_ios_impact
- bevy_ios_review
- bevy_ios_alerts
- bevy_ios_notifications (soon)

SEE HTTPS://GITHUB.COM/RUSTUNIT

THANK YOU

- Join the bevy meetups: bevygamedev.com
- Chat with us on discord: rustunit.com
- We are open for Internships, get in touch
- Meet us at Oxidize Conf in Berlin

GET ZOOLITAIRE NOW:
HTTPS://ZOOLITAIRE.COM