RDC 18

ROBLOX DEVELOPER CONFERENCE

RDC 18
ROBLOX DEVELOPER CONFERENCE

Building Amazing GUIs with Roact and Rodux

Lucien Greathouse

Who I Am

Lucien Greathouse

LPGhatguy

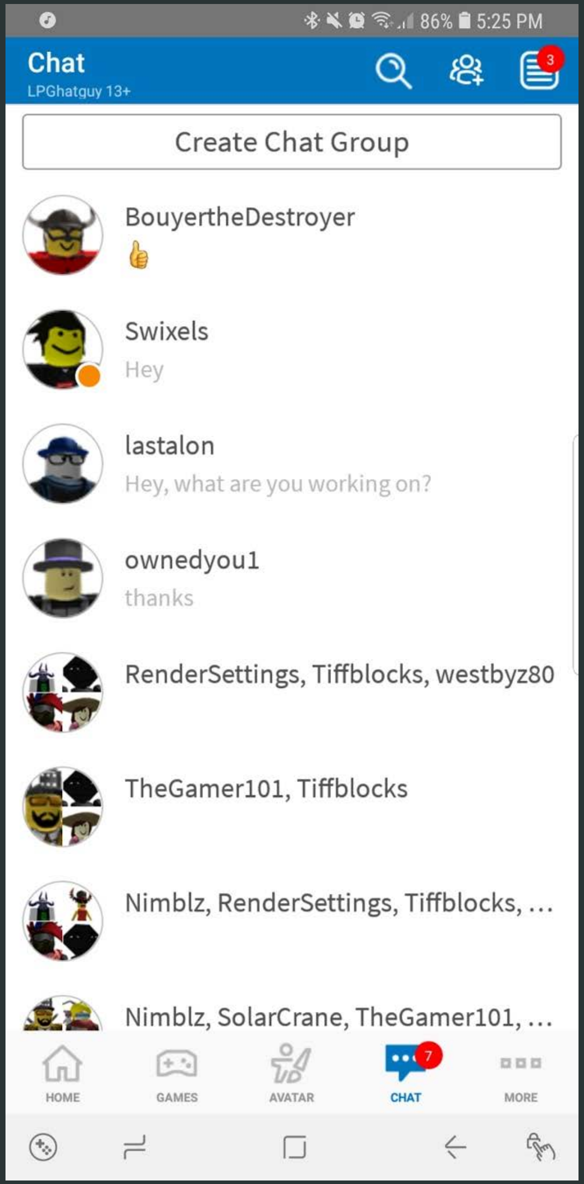
Agenda

- Lua Mobile Chat
- 3 Big Problems
- Solutions
- Demo
- Q&A

Lua Mobile Chat

Mobile Lua Chat Product Goals

- Rebuild mobile chat using Roblox game engine
- Based on success of Xbox and the avatar editor



BouyertheDestroyer
👍

Swixels
Hey

lastalon
Hey, what are you working on?

ownedyou1
thanks

RenderSettings, Tiffblocks, westbyz80

TheGamer101, Tiffblocks

Nimblz, RenderSettings, Tiffblocks, ...

Nimblz, SolarCrane, TheGamer101, ...

Mobile Lua Chat Engineering Goals

- Automated testing for Lua
- Improve quality of Lua code company-wide
- Open source!

Key Challenges

Automated Testing

- Automated testing is easy, you just have to do it!
- No automated testing for Lua code at Roblox 😱
- Roblox has TestService, but it’s not very ergonomic

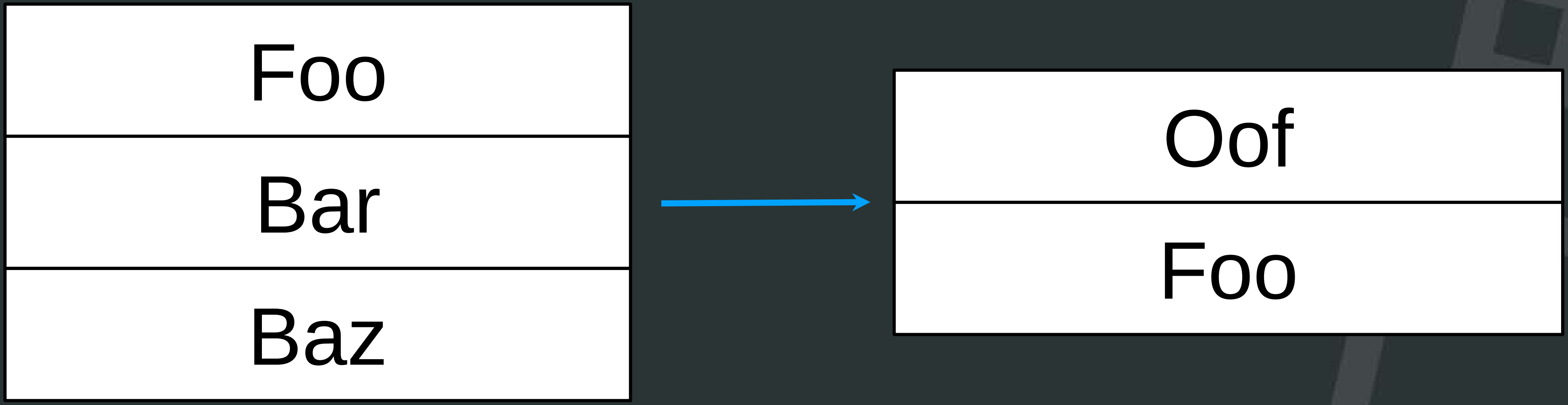
- Current state of the art outside Roblox:
  - Busted (Lua)
  - Mocha/Chai (JavaScript)
  - Ginkgo (Go)
  - Cargo Test (Rust)

State Management

- Lots of code wants to read/write data
  - Networking
  - User interaction
- Everything needs to agree on what that data is!
  - We use the term “state ownership” to describe this idea
- Popular solutions outside Roblox:
  - Redux, MobX
  - Angular, WPF

Dynamic UI

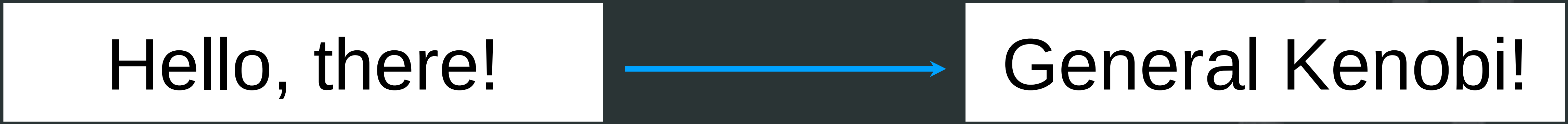
- It’s hard to keep data in sync with UI!
- Simple values, like currency, are easy
- Lists and grids of items are hard



Major UI Paradigms

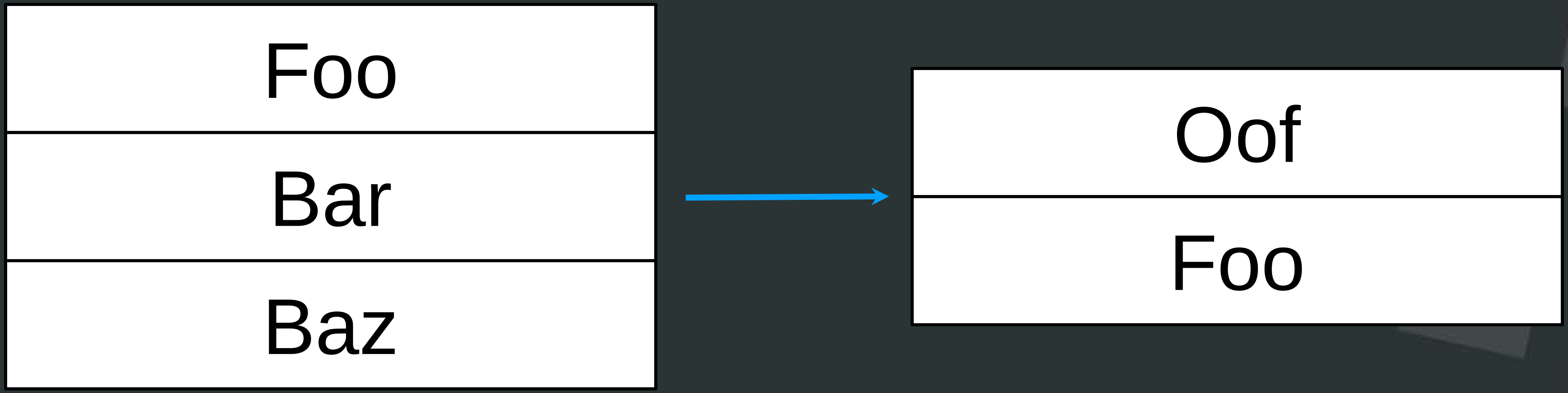
Retained Mode UI

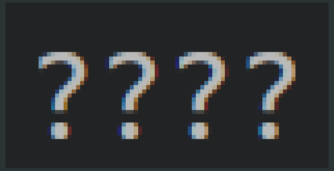
- UI represented by persistent objects
- Changes are performed by setting properties



```
textLabel.Text = "General Kenobi!"
```

Retained Mode UI





Immediate Mode UI

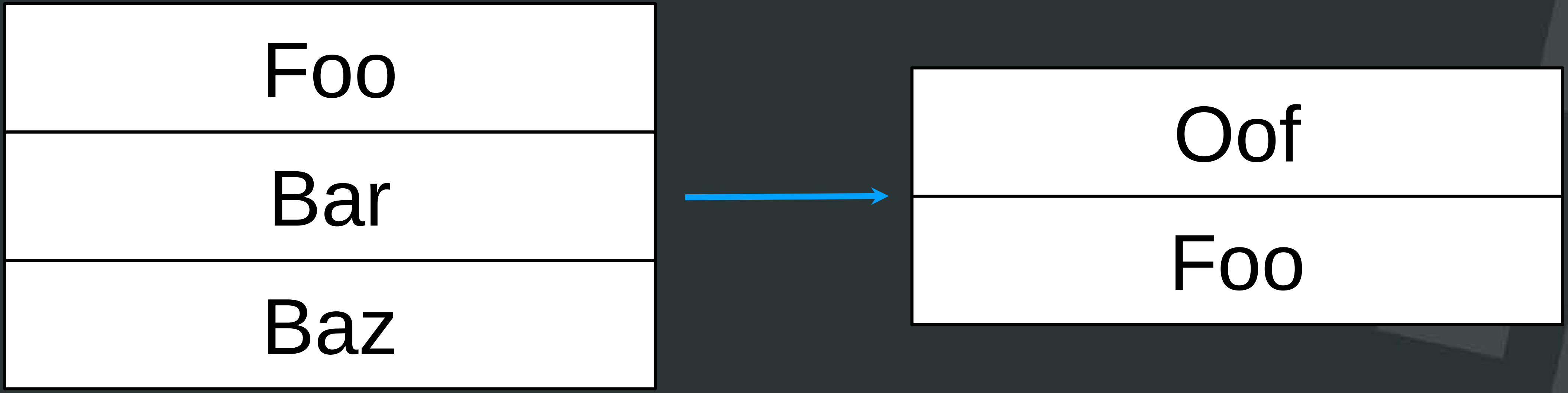
- UI represented by code called every frame
- Immediate mode UI is the *gold standard*
- It can have performance problems!

Hello, there! → General Kenobi!

```
text("Hello, there!")
```

```
text("General Kenobi!")
```

Immediate Mode UI



```
list("Foo", "Bar", "Baz")
```

```
list("Oof", "Foo")
```

Techniques for UI Outside Roblox

- Scaleform
  - Retained-mode Flash UI framework by Autodesk
  - Used in Grand Theft Auto V
- React
  - Declarative JavaScript UI framework by Facebook
  - Used in Battlefield 1

Solutions

- Automated Testing → TestEZ
- State Management → Rodux
- Dynamic UI → Roact

Unit Testing: TestEZ

- Behavior-Driven Development testing framework
- Runs inside Roblox via normal and core scripts
- Also runs inside Lemur, which we use on Travis-CI

```lua
describe("DateTime", function()
    describe("new()", function()
        it("should construct a DateTime object", function()
            expect(DateTime.new()).to.be.ok()
        end)
    end)

    describe("format()", function()
        it("should format dates correctly", function()
            local party = DateTime.new(1999, 12, 31)

            local formatted = party:format("YYYY-MM-DD")
            expect(formatted).to.equal("1999-12-31")
        end)
    end)
end)
```

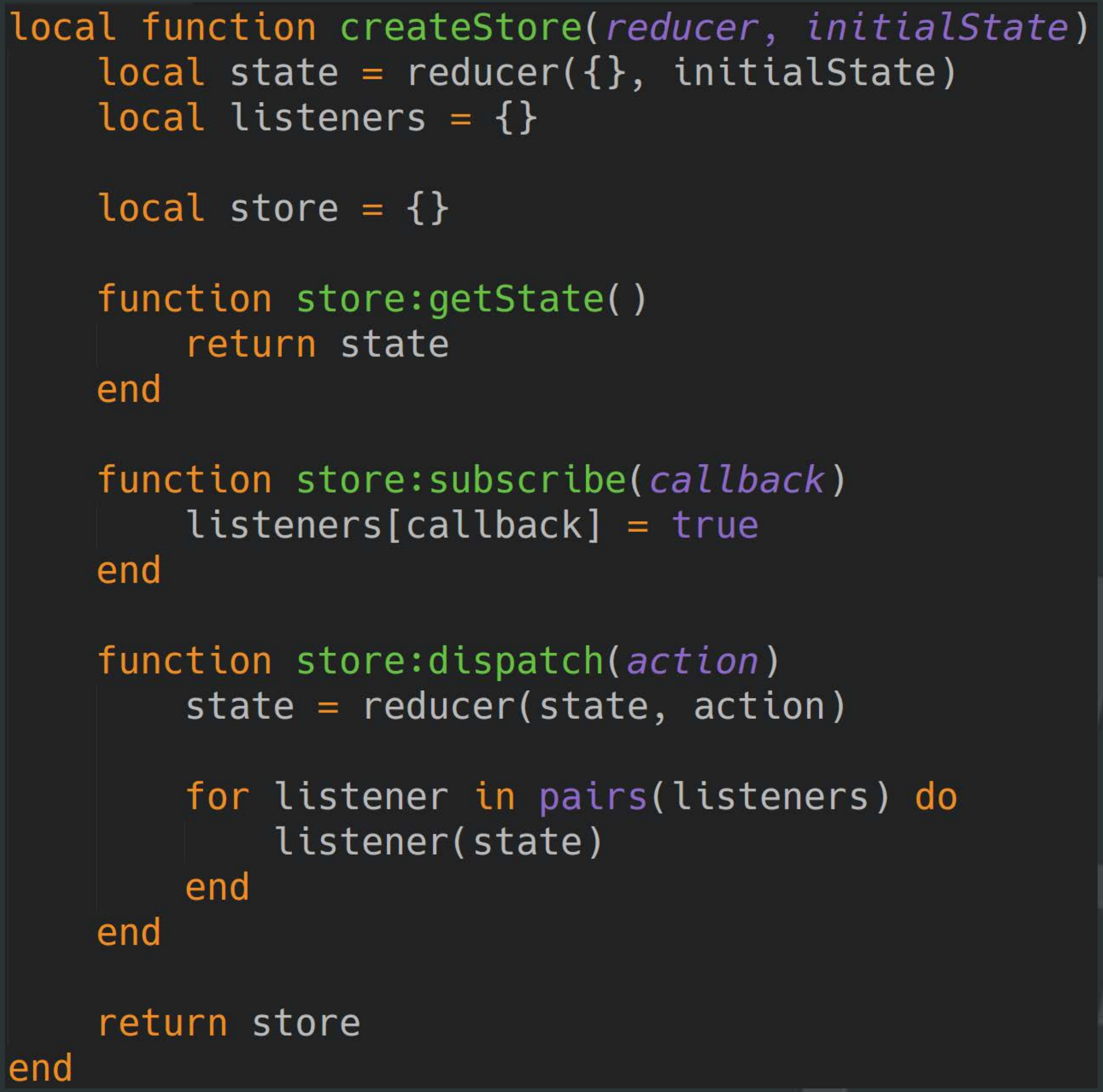
```
$ lua spec.lua
Test results:
[+] DateTime
   [+] new()
      [+] should construct a DateTime object
   [+] format()
      [+] should format dates correctly
2 passed, 0 failed, 0 skipped
```

State Management: Rodux

- Based on Redux, created by Dan Abramov
- Three principles:
  - Single source of truth for all state
  - State is read-only
  - State is defined by pure functions, known as reducers

- Can be implemented in only ~20 lines of Lua!

Redux in 18 lines



```lua
local function createStore(reducer, initialState)
    local state = reducer({}, initialState)
    local listeners = {}

    local store = {}

    function store:getState()
        return state
    end

    function store:subscribe(callback)
        listeners[callback] = true
    end

    function store:dispatch(action)
        state = reducer(state, action)

        for listener in pairs(listeners) do
            listener(state)
        end
    end

    return store
end
```

Create a reducer:

```lua
local function reducer(state, action)
    state = state or 0

    if action.type == "increment" then
        return state + 1
    end

    return state
end
```

Create a store:

```lua
local store = Store.new(reducer)
```

Subscribe to state changes:

```lua
store:subscribe(function(count)
    textButton.Text = count
end)
```

Dispatch actions:

```lua
textButton.Activated:Connect(function()
    store:dispatch({ type = "increment" })
end)
```

Dynamic UI: Roact

- Create components to represent pieces of UI
- Components receive state and return description of UI
- Roact actually updates your UI objects!

Roact tries to emulate immediate mode *ergonomics* without giving up retained mode *performance*.

Hello, Roact!

Define a handy alias:

```lua
local e = Roact.createElement
```

Describe our UI:

```lua
local hello = e("ScreenGui", nil, {
    Label = e("TextLabel", {
        Text = "Hello, RDC!"
    })
})
```

Make our UI real:

```lua
Roact.mount(hello, LocalPlayer.PlayerGui)
```

Alias (surprise!):

```lua
local e = Roact.createElement
```

Component:

```lua
local function Inventory(props)
    local items = props.items

    local children = {}

    children.Layout = e("UIListLayout", {
        SortOrder = Enum.SortOrder.LayoutOrder
    })
```

Create children:

```lua
    for index, item in ipairs(items) do
        children[index] = e("TextLabel", {
            LayoutOrder = index,
            Text = item.name,
            Size = UDim2.new(1, 0, 0, 30)
        })
    end
```

Combine everything:

```lua
    return e("Frame", {
        Size = UDim2.new(0, 400, 0, 300)
    }, children)
end
```

Describe data:

```lua
local items = {
    { name = "Katana of Doom" },
    { name = "Health Potion" }
}
```

Create UI:

```lua
local ui = e(Inventory, { items = items })
local handle = Roact.mount(ui, PlayerGui)
```

Update data and UI:

```lua
table.insert(items, { name = "Super Health Potion" })

ui = e(Inventory, { items = items })
Roact.reconcile(handle, ui)
```

Resources

https://github.com/Roblox/roact

https://github.com/Roblox/rodux

https://github.com/Roblox/testez

https://github.com/LPGhatguy/rdc-project

DevForum: LPGhatguy

Twitter: @LPGhatguy

RDC 18

ROBLOX DEVELOPER CONFERENCE

Q & A