Alex will create a form builder and mocha tests for client side

The way UI particle will work is relaying changes to server and be a websocket

# Work Assignments

Alex: Form Builder

Ryan: Working Client Manager implementation in Rust

Dan: Bidirectional stream... rooms, finish simple message api

Nat: Console App for reading app manifest info

# Console App

Add / Remove Packages

List Human Readable App Name + Signature

# Web sockets and Particles



Each web page(s) is an app.

Apps have UI, probably.

For now, client manager only stores the current stage

## UI setup

Stage

Scene

Component

## New or Reconnecting UI Client

1. Client Requests connection
2. Connection Manager sends client javascript and static assets (“binary”)
3. Connection Manager identifies client, if new then messages START app else resumes LAST app run by client
4. app responds with UI stage
5. Connection Manager messages client with stage change
6. Client renders stage

# Types of Particles

Certain types of particles will have a message type. Some are binary, some are analog

* Switch Type Particle
* Dial

# Form builder



# Store Info

## Package

Manifest : TOML

Name

App Signiture

List Actions Supported (interface)

Permissions Needed

Resources

Binary

Icon

## Action Libs

Action Lib Manifest

Action A...

Action C...

### Action Helper Lib

Action: Implimentation URI

: Local URI

# Structure of App

Apps

App List: TOML

<App Sig>

package

<App Sig>

package

# Types of Actions

UI Action

Open From start

Open specific Scene