Card Battle

By Jacob Hummer, Tanner Brown, Alex Phelan, Solomon Stevens, and Tye Hamilton

What we did

Tye

We made a card game!

Each deck has 20 regular cards and four landscape cards

The game starts with each player drawing 5 cards

Both players start with 25 health and start their turn by drawing one card

They are two phases (Play and Attack)

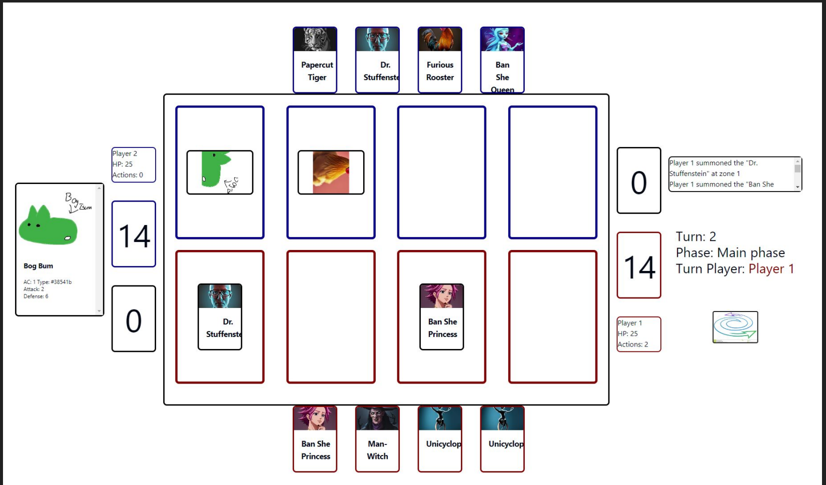
During the Play phase player’s get 2 actions

Actions can be used to summon or draw cards

Every card has an attack, defense, and an action cost

The board is set up with four lanes on each side facing each other

During the attack phase the creatures on both sides attack each other



Bog Bum

AC: 1 Type: #38541b
Attack: 2
Defense: 6

Player 2
HP: 25
Actions: 0

Player 1 summoned the "Dr. Stuffenstein" at zone 1
Player 1 summoned the "Ban She

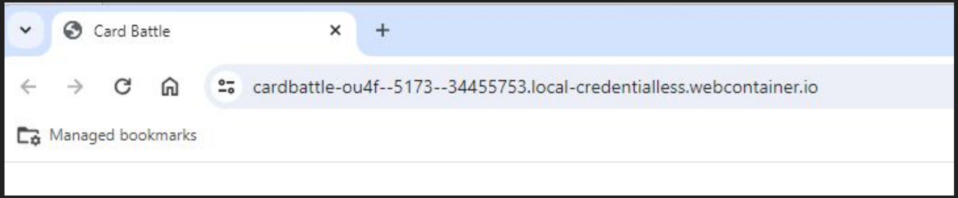
Turn: 2
Phase: Main phase
Turn Player: Player 1

Player 1
HP: 25
Actions: 2

Jacob

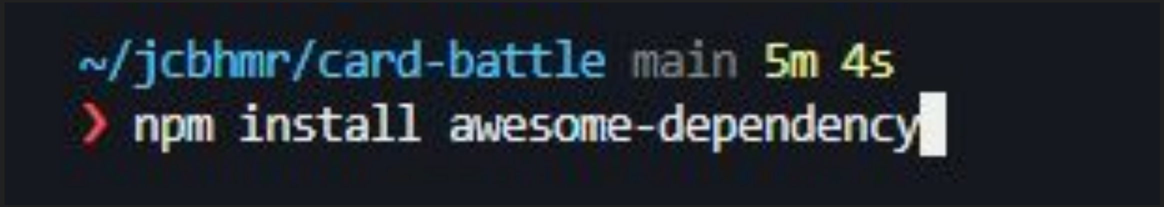
JavaScript & HTML

- Node.js
- npm
- **Runs in a browser!**



Card Battle

cardbattle-ou4f--5173--34455753.local-credentialless.webcontainer.io



```
~/jcbhmr/card-battle main 5m 4s
❯ npm install awesome-dependency
```

https://jcbhmr.me/card-battle/ ← You’ll see this in 3 minutes

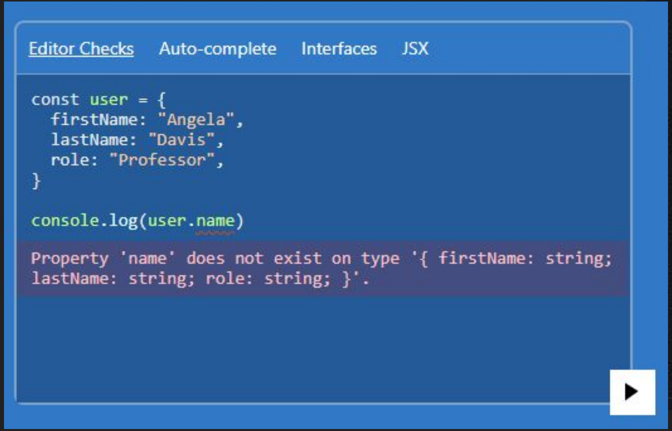
Tanner

React & TypeScript

HTML templates

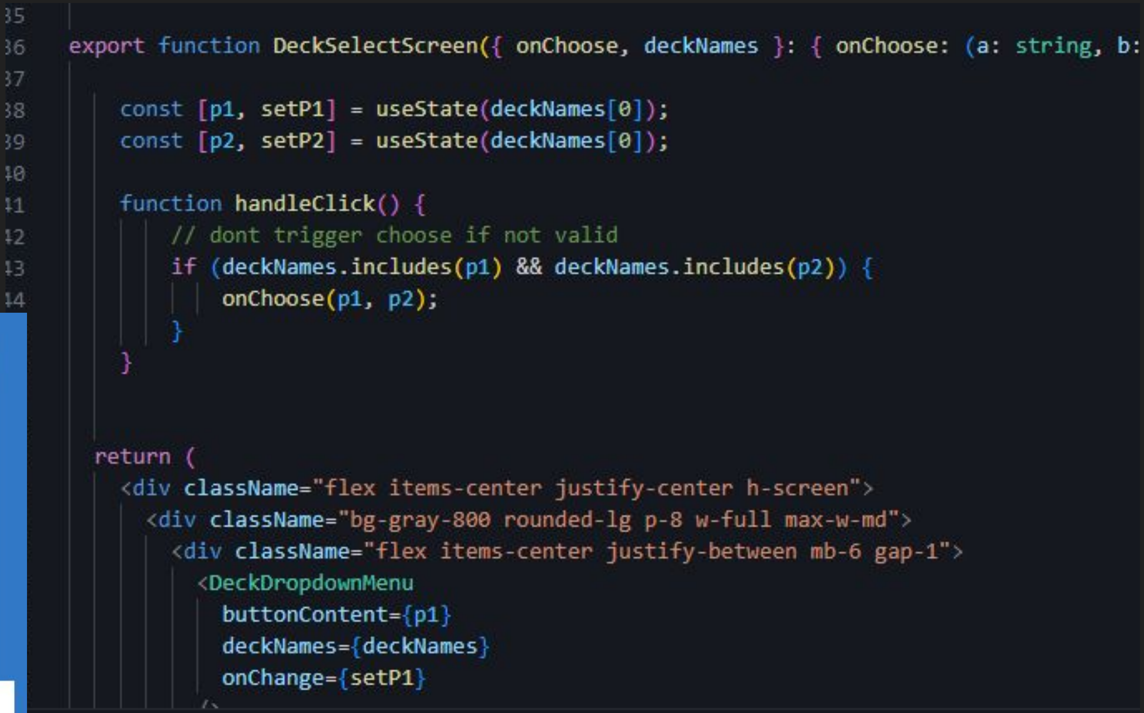
Tailwindcss

Types!



```
const user = {
  firstName: "Angela",
  lastName: "Davis",
  role: "Professor",
}

console.log(user.name)
```

```
Property 'name' does not exist on type '{ firstName: string;
lastName: string; role: string; }'.
```



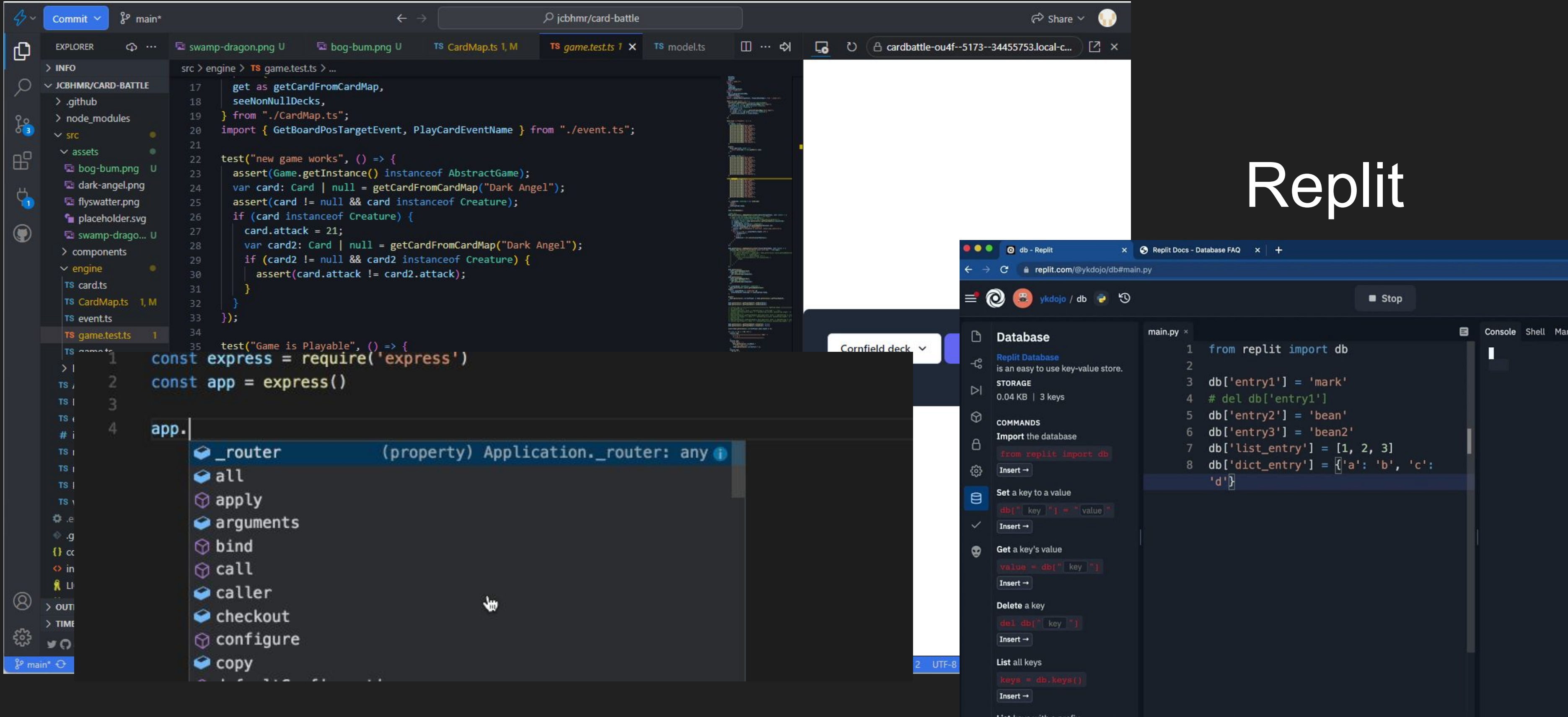
```

export function DeckSelectScreen({ onChoose, deckNames }: { onChoose: (a: string, b:

    const [p1, setP1] = useState(deckNames[0]);
    const [p2, setP2] = useState(deckNames[0]);

    function handleClick() {
        // dont trigger choose if not valid
        if (deckNames.includes(p1) && deckNames.includes(p2)) {
            onChoose(p1, p2);
        }
    }


  return (
    <div className="flex items-center justify-center h-screen">
      <div className="bg-gray-800 rounded-lg p-8 w-full max-w-md">
        <div className="flex items-center justify-between mb-6 gap-1">
          <DeckDropdownMenu
            buttonContent={p1}
            deckNames={deckNames}
            onChange={setP1}
          />
```

Alex

Online IDE

StackBlitz

Replit



```
  get as getCardFromCardMap,
  seeNonNullDecks,
} from "./CardMap.ts";
import { GetBoardPosTargetEvent, PlayCardEventName } from "./event.ts";

test("new game works", () => {
  assert(Game.getInstance() instanceof AbstractGame);
  var card: Card | null = getCardFromCardMap("Dark Angel");
  assert(card != null && card instanceof Creature);
  if (card instanceof Creature) {
    card.attack = 21;
    var card2: Card | null = getCardFromCardMap("Dark Angel");
    if (card2 != null && card2 instanceof Creature) {
      assert(card.attack != card2.attack);
    }
  }
});

test("Game is Playable", () => {
```

```
const express = require('express')
const app = express()

app.
```

```
_router        (property) Application._router: any
all
apply
arguments
bind
call
caller
checkout
configure
copy
```

```
from replit import db

db['entry1'] = 'mark'
# del db['entry1']
db['entry2'] = 'bean'
db['entry3'] = 'bean2'
db['list_entry'] = [1, 2, 3]
db['dict_entry'] = {'a': 'b', 'c':
'd'}
```

Database

Replit Database is an easy to use key-value store.

STORAGE

0.04 KB | 3 keys

COMMANDS

Import the database

`from replit import db`

Set a key to a value

`db[" key "] = " value "`

Get a key's value

`value = db[" key "]`

Delete a key

`del db[" key "]`

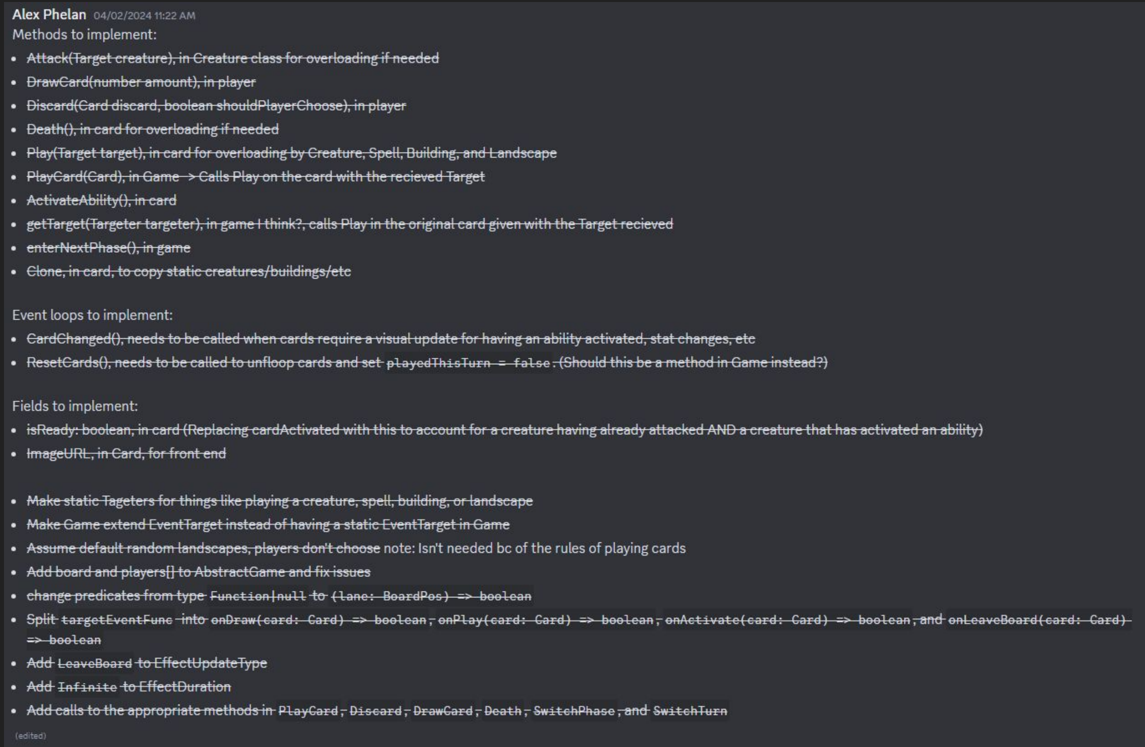
List all keys

`keys = db.keys()`

Alex

Process

- Meet every class day in library
- Group Split
  - Backend
  - Frontend
- Discord
- Use library computers



Alex Phelan 04/02/2024 11:22 AM

Methods to implement:

- ~~Attack(Target creature), in Creature class for overloading if needed~~
- ~~DrawCard(number amount), in player~~
- ~~Discard(Card discard, boolean shouldPlayerChoose), in player~~
- ~~Death(), in card for overloading if needed~~
- ~~Play(Target target), in card for overloading by Creature, Spell, Building, and Landscape~~
- ~~PlayCard(Card), in Game -> Calls Play on the card with the recieved Target~~
- ~~ActivateAbility(), in card~~
- ~~getTarget(Targeter targeter), in game I think?, calls Play in the original card given with the Target recieved~~
- ~~enterNextPhase(), in game~~
- ~~Clone, in card, to copy static creatures/buildings/etc~~

Event loops to implement:

- ~~CardChanged(), needs to be called when cards require a visual update for having an ability activated, stat changes, etc~~
- ~~ResetCards(), needs to be called to unfloop cards and set `playedThisTurn = false`. (Should this be a method in Game instead?)~~

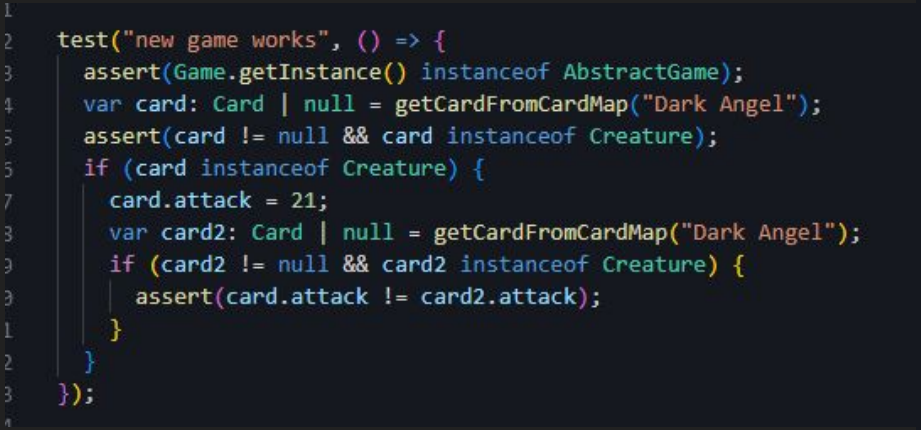
Fields to implement:

- ~~isReady: boolean, in card (Replacing cardActivated with this to account for a creature having already attacked AND a creature that has activated an ability)~~
- ~~ImageURL, in Card, for front end~~

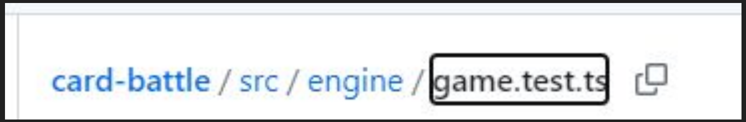
- ~~Make static Tageters for things like playing a creature, spell, building, or landscape~~
- ~~Make Game extend EventTarget instead of having a static EventTarget in Game~~
- ~~Assume default random landscapes, players don't choose~~ note: Isn't needed bc of the rules of playing cards
- ~~Add board and players[] to AbstractGame and fix issues~~
- ~~change predicates from type `Function|null` to `(lane: BoardPos) => boolean`~~
- ~~Split `targetEventFunc` into `onDraw(card: Card) => boolean`, `onPlay(card: Card) => boolean`, `onActivate(card: Card) => boolean`, and `onLeaveBoard(card: Card) => boolean`~~
- ~~Add `LeaveBoard` to EffectUpdateType~~
- ~~Add `Infinite` to EffectDuration~~
- ~~Add calls to the appropriate methods in `PlayCard`, `Discard`, `DrawCard`, `Death`, `SwitchPhase`, and `SwitchTurn`~~

(edited)

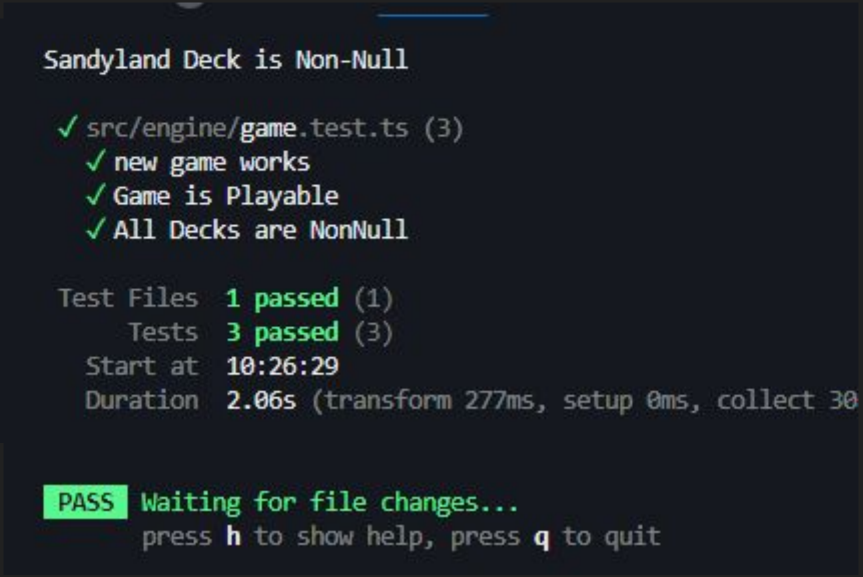
Vitest! Testing is good!



```
test("new game works", () => {
  assert(Game.getInstance() instanceof AbstractGame);
  var card: Card | null = getCardFromCardMap("Dark Angel");
  assert(card != null && card instanceof Creature);
  if (card instanceof Creature) {
    card.attack = 21;
    var card2: Card | null = getCardFromCardMap("Dark Angel");
    if (card2 != null && card2 instanceof Creature) {
      assert(card.attack != card2.attack);
    }
  }
});
```



card-battle / src / engine / game.test.ts



```
Sandyland Deck is Non-Null

 ✓ src/engine/game.test.ts (3)
   ✓ new game works
   ✓ Game is Playable
   ✓ All Decks are NonNull

 Test Files  1 passed (1)
      Tests  3 passed (3)
   Start at  10:26:29
   Duration  2.06s (transform 277ms, setup 0ms, collect 30


PASS  Waiting for file changes...
      press h to show help, press q to quit
```

Demo time!

https://jcbhmr.me/card-battle/

What we would do different

Tanner

Start earlier!

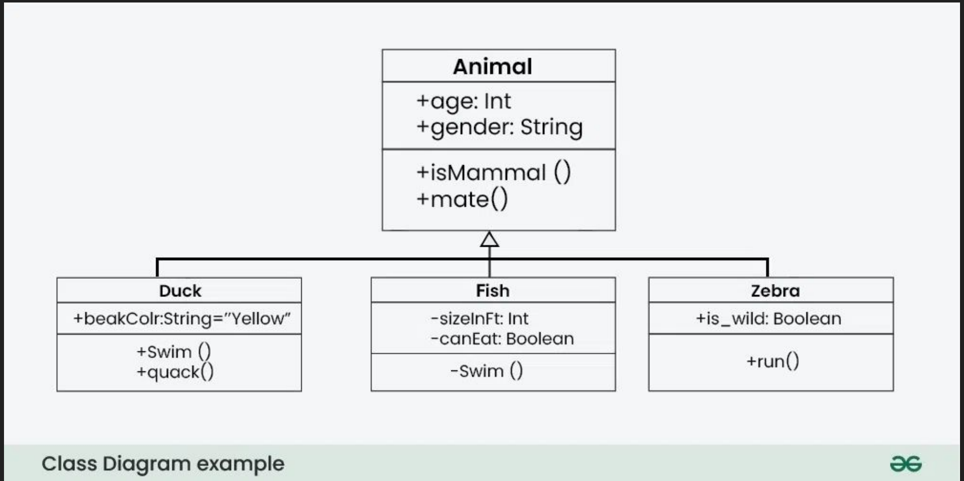
- Started programming a few weeks into the project
- Spent 1 meeting choose game
- Spent 1 meeting choose tech
- Spent 1 meeting bumbling around with throwaway code

Alex

Plan the code.

~~Abilities~~

- What classes?
- What responsibilities?
- What methods call what?
- **How do dynamic abilities work?**



Animal
+age: Int
+gender: String
+isMammal ()
+mate()

Duck
+beakColr:String="Yellow"
+Swim ()
+quack()

Fish
-sizeInFt: Int
-canEat: Boolean
-Swim ()

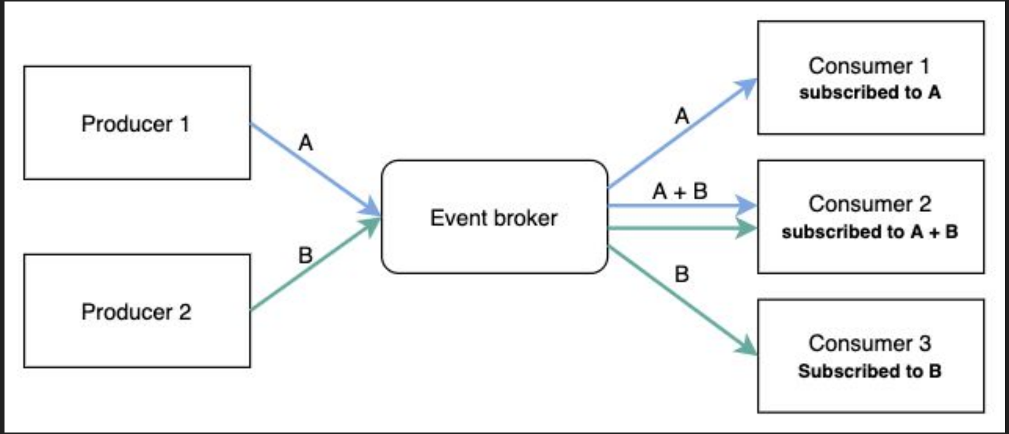
Zebra
+is_wild: Boolean
+run()

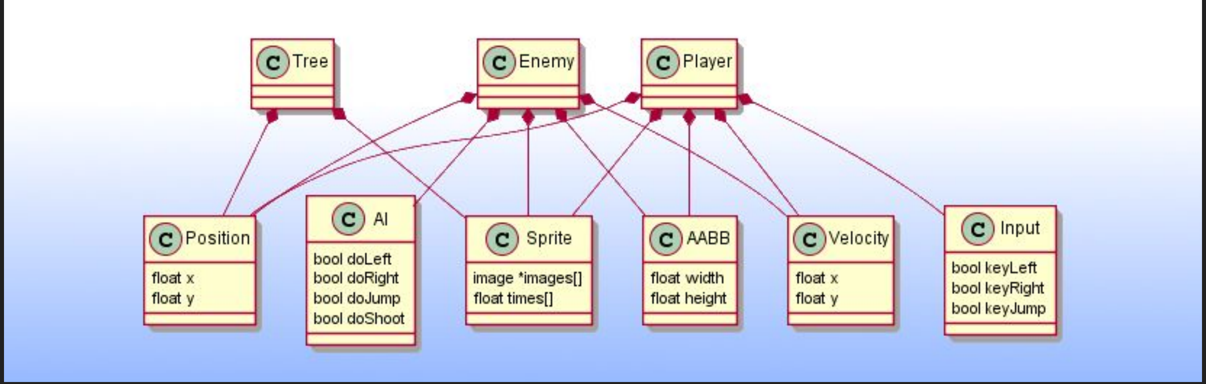
Class Diagram example

Solomon

Event based structure

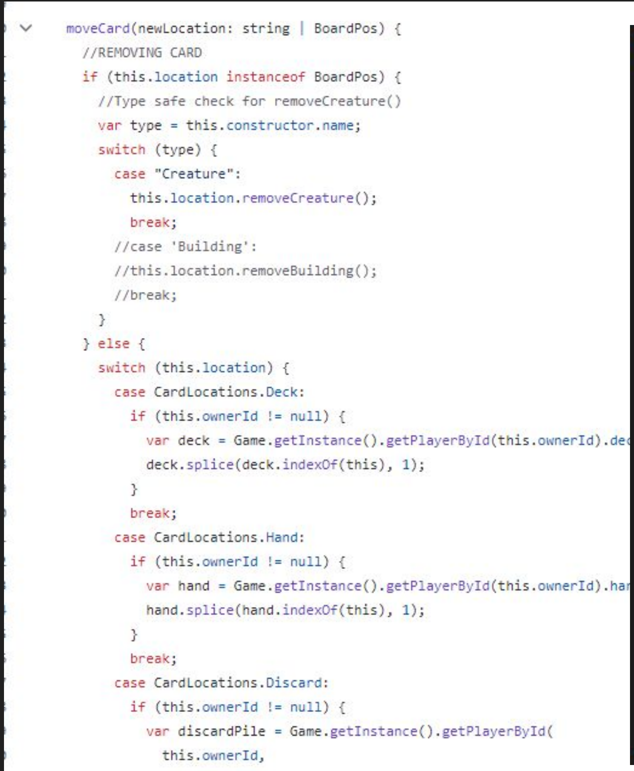
- Entity component system?
- Dynamic abilities!
- Events instead of methods



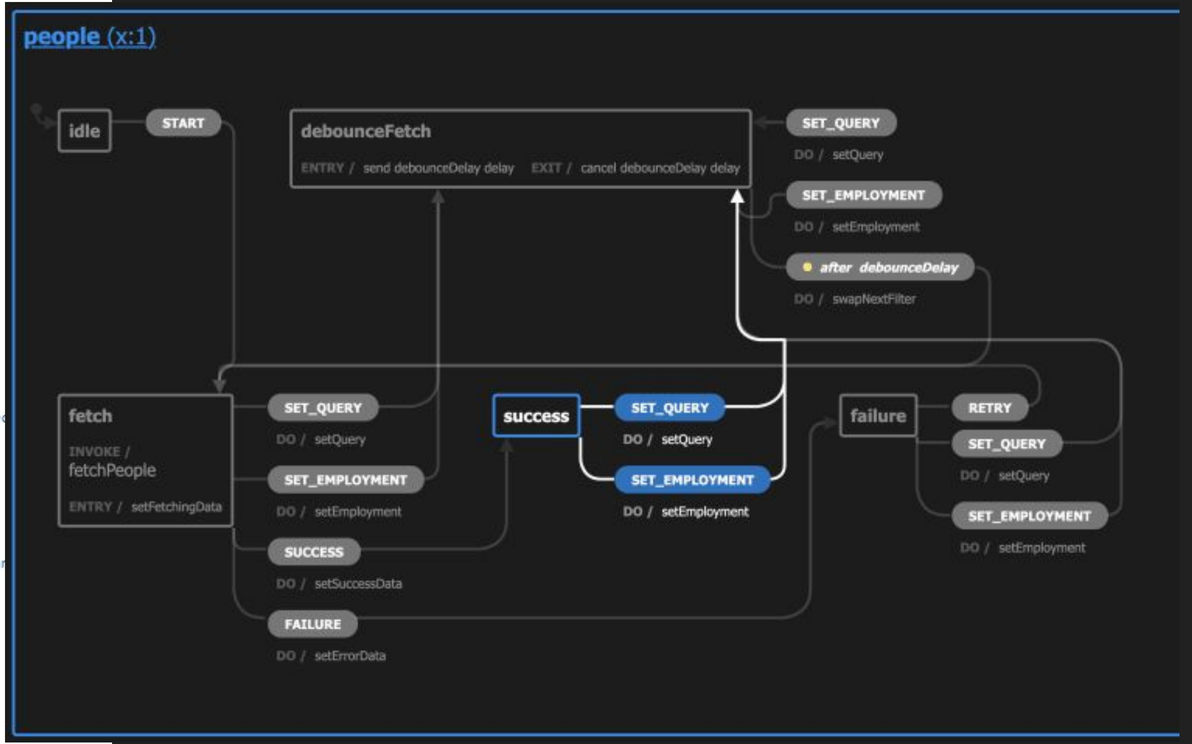


Jacob

Use a library for state machine magic



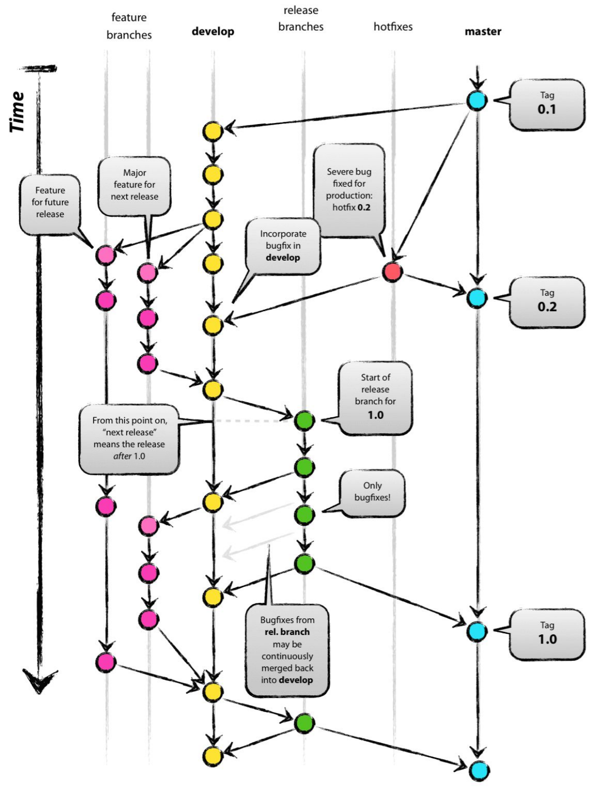
```
moveCard(newLocation: string | BoardPos) {
  //REMOVING CARD
  if (this.location instanceof BoardPos) {
    //Type safe check for removeCreature()
    var type = this.constructor.name;
    switch (type) {
      case "Creature":
        this.location.removeCreature();
        break;
      //case 'Building':
      //this.location.removeBuilding();
      //break;
    }
  } else {
    switch (this.location) {
      case CardLocations.Deck:
        if (this.ownerId != null) {
          var deck = Game.getInstance().getPlayerById(this.ownerId).dec
          deck.splice(deck.indexOf(this), 1);
        }
        break;
      case CardLocations.Hand:
        if (this.ownerId != null) {
          var hand = Game.getInstance().getPlayerById(this.ownerId).har
          hand.splice(hand.indexOf(this), 1);
        }
        break;
      case CardLocations.Discard:
        if (this.ownerId != null) {
          var discardPile = Game.getInstance().getPlayerById(
            this.ownerId,
```



Tye

People & branching

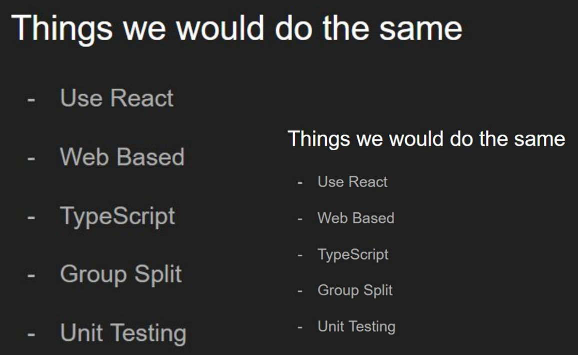
- Use branches!
- Assign specific tasks to people
- Get them done
- StackBlitz with branching


feature branches
develop
release branches
hotfixes
master
Time
Tag 0.1
Feature for future release
Major feature for next release
Severe bug fixed for production: hotfix 0.2
Incorporate bugfix in develop
Tag 0.2
Start of release branch for 1.0
From this point on, "next release" means the release after 1.0
Only bugfixes!
Bugfixes from rel. branch may be continuously merged back into develop
Tag 1.0

everyone

Things we would do the same

- Use React
- Web Based
- TypeScript
- Group Split
- Unit Testing



Things we would do the same

- Use React
- Web Based
- TypeScript
- Group Split
- Unit Testing

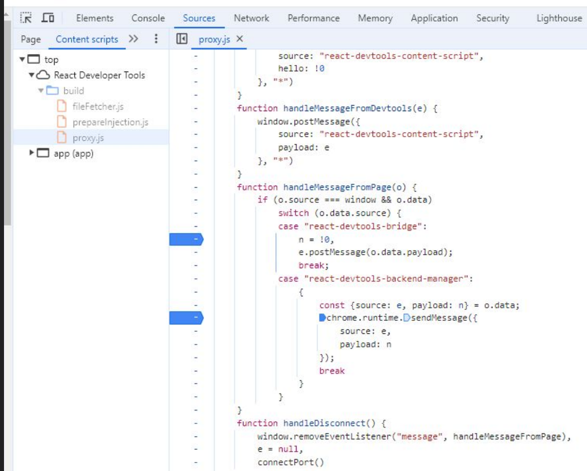
Things we would do the same

- Use React
- Web Based
- TypeScript
- Group Split
- Unit Testing

everyone

Recap: skills acquired

- JavaScript syntax
- How to write TypeScript
- Node.js environment vs browser environment
- How to setup a modern JS project (Vite, npm)
- How linking works (static vs runtime)
- How to unit test (it’s easy!)
- React
- Writing Tailwind Css
- How to debug JavaScript (breakpoints, browser console)
- Use AI tools like v0 and ChatGPT to generate code



```
            source: "react-devtools-content-script",
            hello: !0
        }, "*")
    }
    function handleMessageFromDevtools(e) {
        window.postMessage({
            source: "react-devtools-content-script",
            payload: e
        }, "*")
    }
    function handleMessageFromPage(o) {
        if (o.source === window && o.data)
            switch (o.data.source) {
            case "react-devtools-bridge":
                n = !0,
                e.postMessage(o.data.payload);
                break;
            case "react-devtools-backend-manager":
                {
                    const {source: e, payload: n} = o.data;
                    chrome.runtime.sendMessage({
                        source: e,
                        payload: n
                    });
                    break
                }
            }
    }
    function handleDisconnect() {
        window.removeEventListener("message", handleMessageFromPage),
        e = null,
        connectPort()
```