Developer Log: Design Decisions rationale and justification, design questions you may not be able to answer them in the log that is ok, technical debt we all do bad things sometimes this just calls them out for all to see and notes our understanding.

These are all the notes I wrote on my phone. The command input reader monogame project has a more in-depth devlog, but this is where I took all my notes after I started work on the game proper

gp2 fighting game  
  
proposal: fighting game  
  
proof of concept: input reader  
  
could write the input reader as a POCO class. week 1 put it in monogame and use the console to display inputs and alert when a quarter circle / dp is done  
  
use the command pattern to fire off commands for buttons or command inputs  
  
  
controller inputs come in ->  
package into an input class that holds all the buttons from that frame, and the time/frame it was ->  
put that input class into a list / stack / queue ->  
read that list, check for more complicated commands before simpler ones  
⁃ full circle  
⁃ pendulum  
⁃ half circle  
⁃ double quarter circle  
⁃ DP  
⁃ quarter circle  
⁃ charge???  
⁃ command normals  
⁃ normals  
  
structure:  
IInputHost: needs to be written per engine, turn raw inputs into readable input data  
  
InputHandler: takes data from input host, reads it, returns output  
has a priority queue of IGestures that it reads through.  
will return a priority queue of all igestures that are applicable to the current set of inputs  
  
IGesture: 1 command input. had a command like Check() or something. that checks the inputs and decides if it qualifies or not. also has a priority value  
IGestureOutput?? or maybe the same gesture. gets put out in the end  
  
InputReceiver / playercontroller / whatever: takes the priority queue of applicable inputs, and then goes through and does whichever the highest priority action that the player character has a move for is. ex: they don’t have a double quarter circle forward, but they do have a DP so do the DP  
  
ideally this whole system is as button agnostic as possible. want it to take in any set of buttons and it can read them, no matter if it’s a 2 button fighter or 6. or what the buttons are called.  
the direction inputs could be a built in enum since there will only ever be 9, but i’ll need some way to get the buttons setup  
  
buttons could have their own unrelated priority queue, unrelated to the gestures. and then each IInputResult would have a queue of command inputs and a queue of the buttons ?  
would not allow for button-direction commands. like you couldn’t do skugs taunts like that  
  
⁃ make gestures get read via the same input loop, instead of each doing its own loop over the inputs?? probably unnecessary optimization, until it’s not.  
  
  
okay we need a new output class.  
IMove  
IMove contains a gesture type, and a button  
could still have a list or priority queue of buttons, to allow for 2 button input moves like EX moves or supers  
  
the readpackage now has the most recent hostpacjage, and a priority queue of Moves  
since gestures only matter to the game if they have a button attached. doing quarter circle nothing is nothing. so that means we also only need to read gestures if there is a button we can use.  
can also implement an input buffer this way, for if you press the button a little before you finish the gesture. we only read gestures if there was a button pressed in the last [buffer time] frames  
  
IMove needs to be simple equatable and have a hash code that’s the same between moves with the same gestures and buttons.  
  
because eventually, the Fighter Class will have a dictionary of <IMove, Attack> or something to check which attack should come out  
  
maybe first we have a list of every possible Attack our fighter can do.  
  
then when we have input.  
we cut that list down to just every attack that theoretically can execute (all the aerial attacks when we are in the air, for example.)  
  
then we check if any of those want any of our gestures and button(s?), in priority order  
  
attacks could use the decorator pattern to have a list of conditions  
  
  
character select and player input:  
okay so what if i grab the device off the player configuration playerinput. and then get rid of that playerinput.  
and then when i make my player, i unpair it with any other device and i pair it with that device  
  
  
  
animations checklist  
movement  
✓ idle  
✓ walk  
◦ dash - could be a slide like slayers 2d  
◦ back dash  
✓ jump  
✓ air  
✓ crouch  
  
block/hit  
✓ stand block  
✓ crouch block  
✓ air block  
✓ stand hit  
✓ crouch hit  
✓ air hit  
✓ stand/stagger - can just be in hitstun for a long time  
✓ flying through the air before kd - can just be air hitstun  
✓ soft knockdown? - can just be hkd for less time  
✓ hard knockdown  
✓ hkd stand up  
  
buttons  
✓ 5A - goldlewis jab  
✓ 2A - crouch jab  
✓ j.A - beo jLK  
✓ 5B - gio 6P  
✓ 2B - crouch kick  
✓ j.B - normal jumping kick  
✓ 5C - beo 5HK? or! black dahlia overhead?  
✓ 2C - uppercut launcher  
✓ j.C -  
✓ A+B? or 4B/6B - grab - pot air grab bonk. or headbutt  
✓ 236A - hadouken - stomp fireball effect gunflame. goldlewis 5K with gunflame projectile. stomp could be a low. shoots a wave like fish guy fantasy strike  
✓ maybe more like annie H crescent cut? like you shoot it out and run behind it  
✓ 623B - DP. water volcanic viper??  
◦ 236B - dire eclat / stun dipper? some sort of hkd special combo ender  
◦ stun dipper but the two parts are split. kinda like elk hunt  
◦ slide sweep that causes hkd. but is minus on block and leaves you in their face  
◦ follow up slash that has high knock back, causes soft knockdown on hit. or at least hkd at a far range for worse oki  
◦ sweep is laying / sliding on a surf board, the follow up is popping up on the board. or maybe popping up and hitting you with the board  
◦ 214A - aerial special command air dash  
◦ sort of like gios flip kick but in a horizontal air dash form. probably a 214 input? but maybe 236  
◦ so it’s got a hit box right at the start that only hits behind you  
◦ and then it just air dashes forward  
◦ the cross up hitbox would probably give a soft knockdown like gios cross up flip kick does?  
◦ or maybe the crossup drags the enemy with you, and you can combo from it with air hits  
◦ the air dash would have a pretty short recovery and you would be actionable after  
◦ animation is like breast stroke :) and your frog kick is the backwards hitbox  
◦ super: 641236 (forward, half circle forward)  
◦ ????  
◦ maybe a big projectile ?  
◦ or just a big hit like +r tyrant rave  
◦ a fast, far reaching reversal shoulder charge, functions like beo gigantic arm. but not as tall  
  
pressure based around meaty gunflame like ky CSE pressure  
  
grappler character- Captain Blackhand  
⁃ pirate  
⁃ barnacle billy  
⁃ captain will “barnacle bill” roddy  
⁃ captain jensen e jones (davy jones brother)  
⁃  
⁃ sea green, off limbs are pure black, special color is brown. or maybe white?  
⁃ command grab where he grabs you and then pulls out a cannon that shoots you at point blank range. and it shoots you and launches you full screen. maybe wall bounces? or maybe just leaves you cornered.  
⁃ long range anti air grab with a blind spot up close. like heat knuckle but not as high / close. designed mostly to catch backdashes and some jumps  
⁃ slow dash, can’t do hop normals  
⁃ can make a wall like crusader dnf to force strike throw corner pressure anywhere  
⁃ has a parry? to hard call out mashing. flies a white flag, turns to black when hit, then ??????  
⁃ or maybe it’s an invincible/armored run/ dash where you surf a wave  
⁃ anti air move where he summons a shark to jump straight up right in front of him. good for catching jumps and also as a pseudo launcher like beo’s chairless 2hp. launches on air hit, pulls downwards on air block  
⁃ maybe some sort of divekick?  
⁃ what if he set up a cannon on the ground that he could later fire remotely  
⁃ he makes a wave, kind of like argagarg in fantasy strike. but the wave goes backwards, instead of coming from him and going forwards, it goes from the edge of the screen towards him. so it’ll push enemies towards him (so he can grab them)  
⁃ maybe on block it has some vacuum effect, and on hit it has more vacuum and causes a soft knockdown  
⁃ one move to help punish someone for jumping to avoid his grab (the shark)  
⁃ move where he sits on the cannon and fires it backwards and rides it as it goes forward  
⁃ peg leg forward kick  
⁃  
  
  
game is train themed  
the map is loosely based the California station cta  
  
title:  
✓ Traintracks Combat  
◦ Train-atTracks  
  
  
round start call:  
  
“all aboard!”  
“tickets please!”  
“chugga chugga choo choo!”  
“i’ve got a schedule to keep!”  
“you’ve got a one way ticket to hell!”  
  
“Go!”  
“Lay on!”  
“Fight!”  
“Duel!”  
“Brawl!”  
“Fisticuffs!”  
“Bash!”  
“Quarrel!”  
  
round end:  
choo choo!!!  
“that’s the ticket!”  
“end of the line..”  
“this is your stop”  
“good show!”  
“Heh. That tracks..”  
“Please don’t lean against the doors.”  
  
  
important things to add after playtesting  
✓ stop standing on peoples heads  
✓ juggle combos / air hit properties  
✓ tech throws  
✓ dashing, backdash  
✓ fireball  
✓ lower health amount  
✓ more recovery on fireball  
✓ fireball shoot sound  
✓ fix wallbounce bug?  
✓ rounds  
✓ better juggle scaling, make mid screen combos possible but infinites worse  
✓ scale Y knock back to be less, but X knock back to be more. so you’ll juggle less high and further away with longer combos  
✓ probably adjust or maybe remove the current momentum scaling  
◦ maybe cave and add some sort of IPS  
✓ stop flipping on backthrow tech  
✓ second character  
✓ character select screen  
◦ rebindable controls  
✓ DP landing recovery  
✓ throws deal more damage  
✓ air grab  
✓ make character select screen better  
✓ shark should have a rising hitbox  
✓ more colors  
✓ color / portrait on healthbar  
◦ rematch and character select button on round end  
✓ 2C should have less horizontal knockback  
✓ improve wave  
✓ spawn a set distance from the wall, not from the player  
✓ come on screen sooner but move slower  
◦ maybe work out a way for self damage to work but not suck  
✓ background  
✓ train  
◦ character specific round start intro animation  
◦ make command grab grab invuln

Standard Documentation:

Please include the following documentation.

Game Name: Traintracks Combat

Game Description Short: A fighting game that takes place at a train station

Game Description Long: A momentum-based fighting game featuring two characters, special moves, and unique movement mechanics.

Credits:

Graphics, animations, programming, design by me

Some sounds from freesound.org

Some sounds originally created for my previous game Balance Block Battle by Eli Armero

Genre: Fighting Game

Postmortem

Class UML diagram with descriptions of class responsibilities.

Command input reader UML:

A screenshot of a computer screen

Description automatically generated with low confidence

Game UML

A screenshot of a chat

Description automatically generated with low confidence

What patterns did you implement in your game. How did the help manage the structure of your game classes and their dependencies.

FighterStates: Subclass Sandbox & Strategy pattern: allows for many different implementations easily

GameAttackConditions: Decorator pattern, easily allows attacks to be created and changed with different execution requirements

PlayerConfigurationManager: singleton. Only have one, in charge of the player configurations

What challenges did you have making your game components? What techniques did you use to overcome these challenges.

Creating the command input reader- a whole lot of thinking, and a lot of reading about systems in other fighting games

Deciding which attack to execute, since multiple attacks have the same button press- used the decorator pattern to make a series of conditions, described in UML above

Creating a second playable character- learned to use scriptable objects, and moved many of the fighter’s stats to the CharacterModule scriptable object

Hitboxes and hurtboxes- read an article about one way to do them in unity, and adapted it for my own game

Character select- watched a tutorial and adapted/expanded upon it for my own game

Rate how reusable the game codebase is? What percentage is specific to the game? What percentage is specific to the platform (monogame/unity)?

The command input reader is platform independent (made in monogame, used in unity)

The game code has some strong ties to unity, but could be ported without an insane amount of hassle. Just have to adapt to another game engine’s physics systems and such. And I think all the code for GameAttacks (and their conditions and properties) is platform independent as well.

Probably around 50% independent, 25% slightly dependent, and 25% very dependent

How maintainable is your code? Is it easy to add more enemies, pickups, weapons, levels etc?

Quite easy and maintainable. Adding new states is easy. Adding new attacks is a dream. Even adding new characters is not hard, mostly just setting up the new stats and attacks.

Did you finish your game? Is there any outstanding technical debt?

It is much more finished than I scoped for in my proposal. It is pretty close to what I’ve been striving for the past few weeks. Maybe 1-2 more weeks and I’ll release it online as a portfolio piece. To be more than a portfolio piece, it would need more characters, which does take more time due to design and animation.

Technical debt: the input reader still has some issues. It behaves oddly when your character turns around between when an input is stored in the buffer and when it is executed. It is not yet built to handle different types of button press (hold down the button, release the button, or press two buttons at once)

The game has some technical debt too, and I think a lot of it will be difficult to fix while using just plain unity. The animations sometimes loop and show the wrong frame before transitioning. I haven’t figured out a good solution to not be able to stand on your opponent’s head. And you can walk into each other and push each other around too much or too little depending on character. Some timing things are weird because all the attack speeds are measured in frames but much of the other timing is done using fractions of a second which aren’t necessarily the same. Also the game behaves inconsistently if the two players hit each other at the same time, favoring one player over the other.