Hi!

Welcome to my design folio.

I've been a digital creative since I got my hands on MS Paint and Notepad in 1998. I started my career in web development, took a detour through retail/marketing, and am now finding my place in the games industry.

Most of this folio is about my latest game, Call the Wizards, which I fully designed and coded as a solo developer.

I'd love to bring my skills to your team and continue learning and making awesome games.

Thanks for taking a look!

- Shaun Norton

Call the Wizards!

Other Projects

Call the Wizards!

Game/Grant Pitch

The game was in early development when Meta announced a funding program for up to $25,000 USD to produce an innovative vertical slice for mixed reality.

I seized the opportunity for funding and potential exposure/Meta contacts.

The pitch included core mechanics, scope of development, and how it would utilize MR capabilities.

The pitch was approved within 1 week.

The 7-page pitch is attached separately.

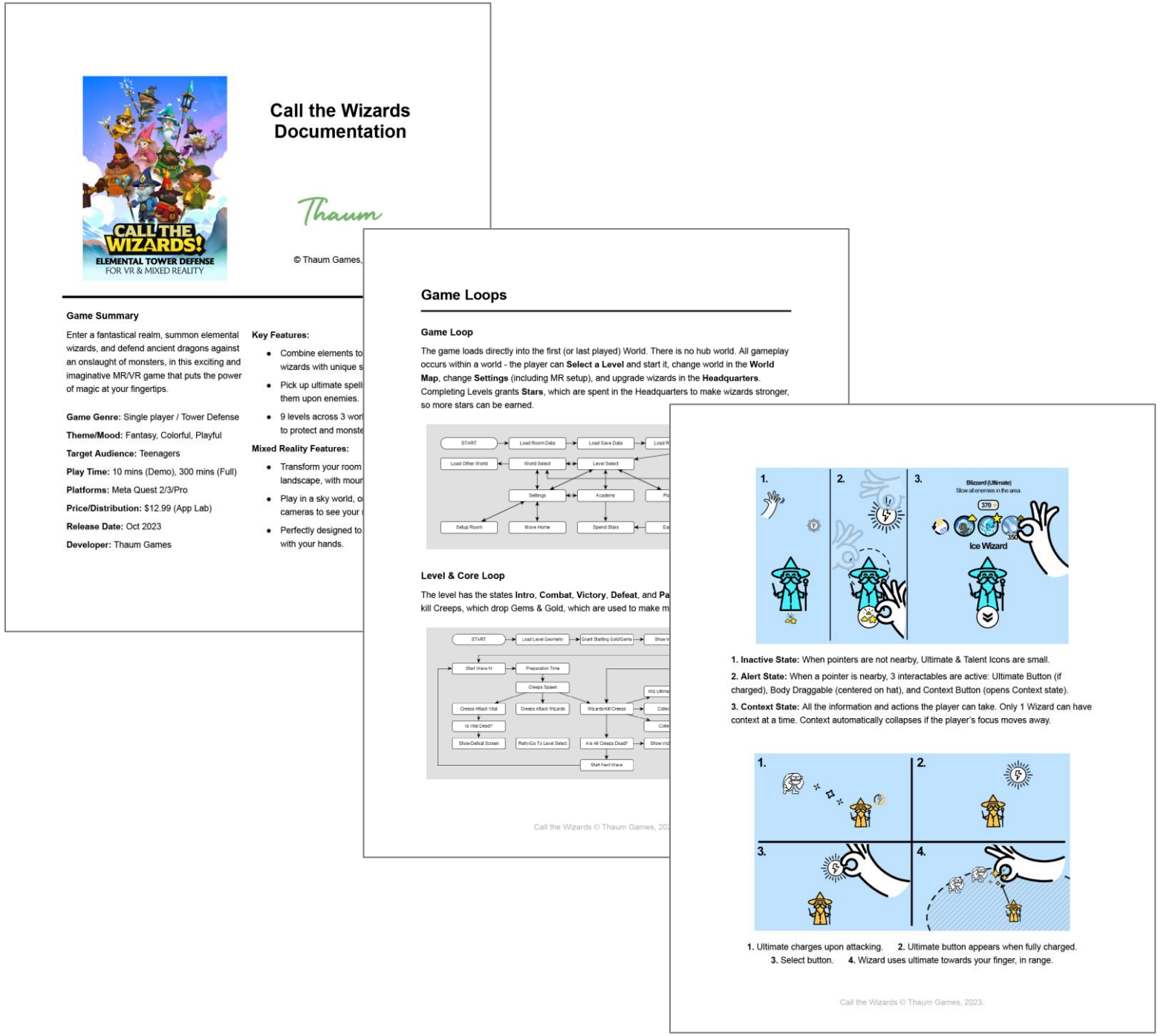
Game Design Doc

The first milestone for the Meta funding program required the delivery of a game design document.

My previous games had evolved through constant iteration, without a tangible design document, so this was an exciting new task for me.

I found that my experience in graphic design and writing well-formatted reports came in handy.

The 20-page GDD is attached separately.

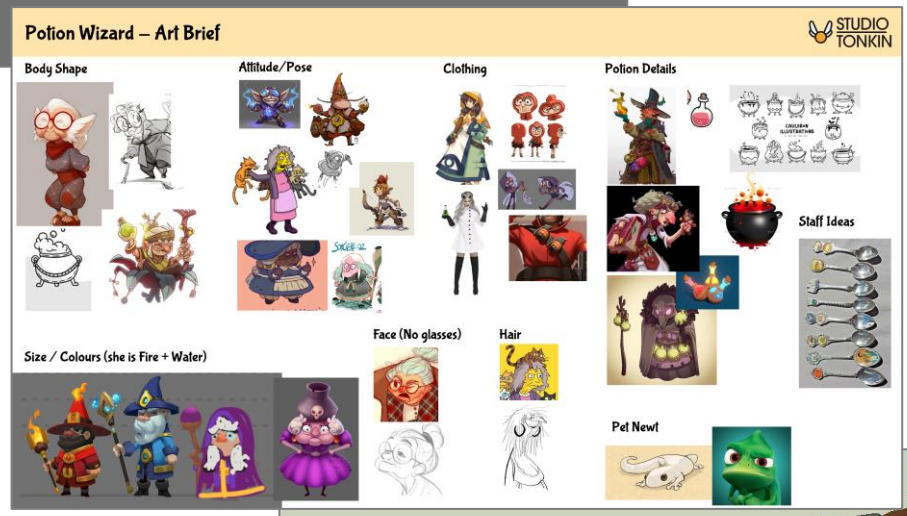


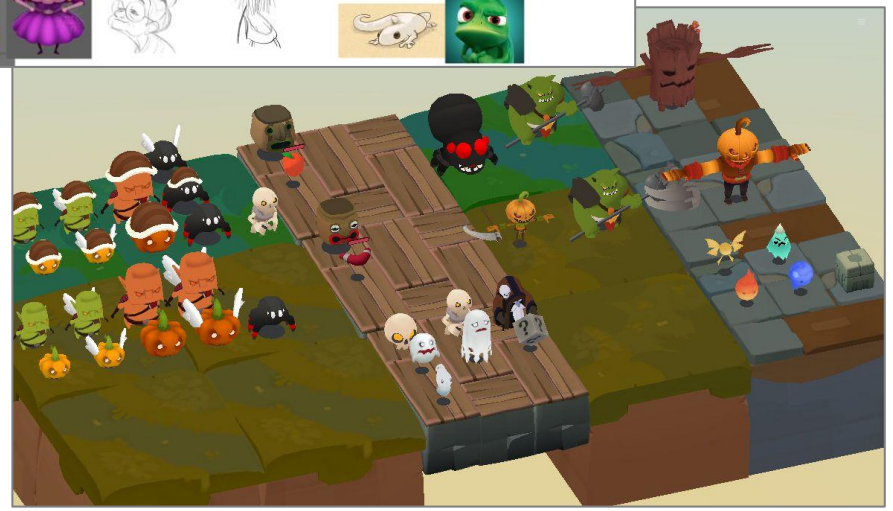
Characters

The player summons and manipulates wizards, so it was important to make them distinct and playful. I used simple models during early prototyping to get a feel for the game.

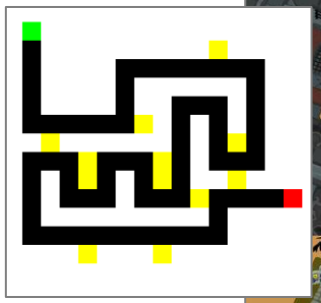
I loved finding the personality of each character and communicating it to the 2D artist via briefs and feedback sessions.

Due to budget, all of the enemies were sourced via the Asset Store and my own modifications. So, where possible, I made variants using size, colour and attachments like wings/armor. The final game had around 50 enemy types.





Map Design



The game shipped with 17 levels.

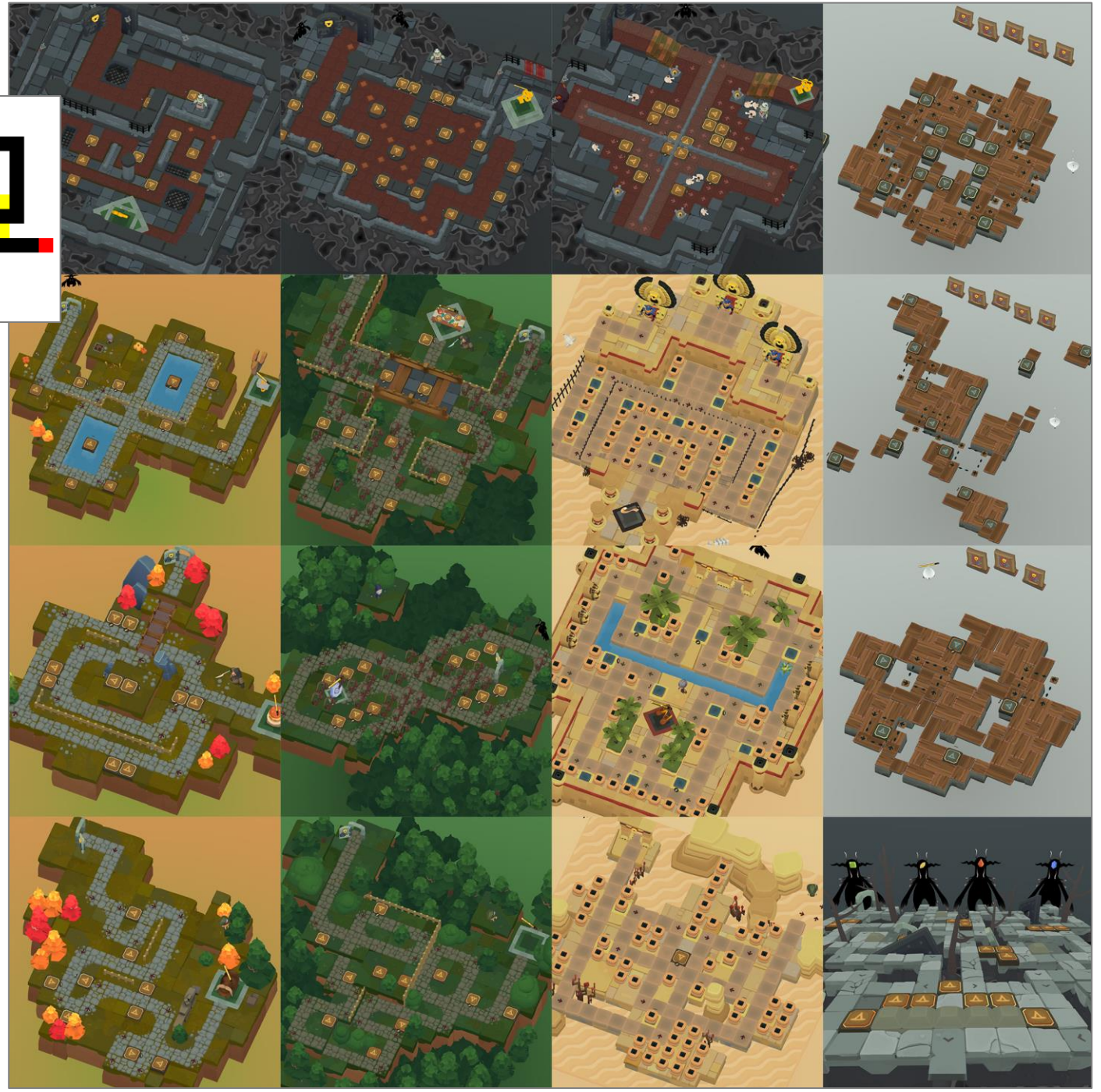
I studied similar games like Kingdom Rush and Bloons TD, to learn how to make better maps, then sketched them in Photoshop.

After I understood what makes a map interesting, I then had the freedom to experiment with a mix of symmetry and randomness.

Some maps were designed with a single optimal strategy, while others had a more aesthetic appeal.

I also created some unique objects per biome, and the ghost levels have a unique objective entirely.

The designs were iterated upon while I was designing the enemy spawning patterns.



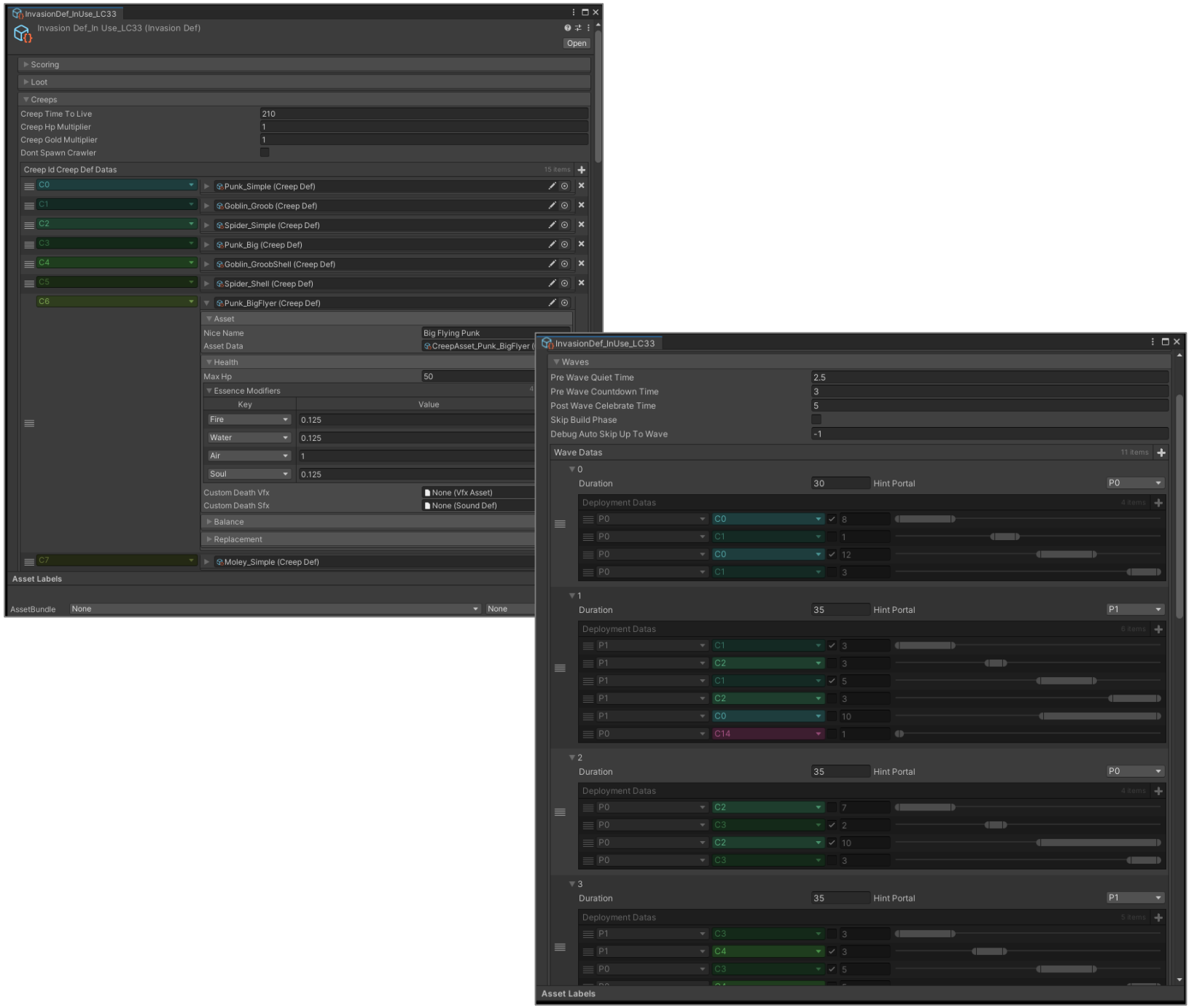
Spawn Data

Each level had different creeps, loot, spawn points and quantity/timing.

I designed my data such that I could copy and paste between levels and make adjustments easily, for example, using a loose relation between a creep enum and the actual creep def that it referred to.

Creep data also included their base stats and elemental immunities.

The sliders on the right helped me visualize the period of time in which the creeps spawn.



Wizards	Spell Design								Ranked Damage						Damage Per Second						Rank Cost				Total Cost				Cost of DPS			
	Period (I	Period (s	Start	Stop	Atk Beat	Atk Area	Attacks	APS	D1	D2	D3	D4	D5		DPS1	DPS2	DPS3	DPS4	DPS5		C2	C3	C4	C5		TC2	TC3	TC4	TC5		E2	E3	E4	E5
Flame	8	3.7	5	1	4	9	36	9.8	3	4	6	7	8		29	39	59	68	78		70	100	150	210		70	170	320	530		1.8	2.9	4.7	6.8
Ocean	8	3.7	0	4	4	14	56	15.2	2	3	4	5	6		30	46	61	76	91		70	100	150	210		70	170	320	530		1.5	2.8	4.2	5.8
Rock	8	3.7	1	4	3	8	24	6.5	5	7	9	12	14		33	46	59	78	91		50	70	100	150		50	120	220	370		1.1	2.1	2.8	4.1
Wind	4	1.8	3	1	2	3	6	3.3	10	14	18	23	28		33	46	59	75	91		70	100	125	150		70	170	295	445		1.5	2.9	3.9	4.9
Lava	4	1.8	1	2	1	10	10	5.4	4	5	7	10	13		22	27	38	54	70		70	100	125	150		70	170	295	445		2.6	4.5	5.4	6.3
Ice	8	3.7	0	2	2	10	20	5.4	4	5	7	10	13		22	27	38	54	70		70	100	150	210		70	170	320	530		2.6	4.5	5.9	7.5
Wood	8	3.7	0	3	3	10	30	8.1	5	7	9	11	13		41	57	73	89	106		50	70	125	150		50	120	245	395		0.9	1.6	2.7	3.7
Storm	4	1.8	2	0	2	6	12	6.5	5	7	10	12	15		33	46	65	78	98		100	210	250	400		100	310	560	960		2.2	4.8	7.2	9.8
Sand	16	7.4	0	5	5	13	65	8.8	5	7	9	11	13		44	62	79	97	114		70	100	125	150		70	170	295	445		1.1	2.1	3.0	3.9
Potion	8	3.7	4	6	2	21	42	11.4	3	4	5	7	8		34	46	57	80	91		100	125	150	210		100	225	375	585		2.2	4.0	4.7	6.4
														Avg	32	44	59	75	90							72	179.5	324.5	523.5		1.8	3.2	4.5	5.9
														Target	30	45	60	75	90												1.5	2.2	2.7	3
																									Avg TC	274.875							

Combat/Economy

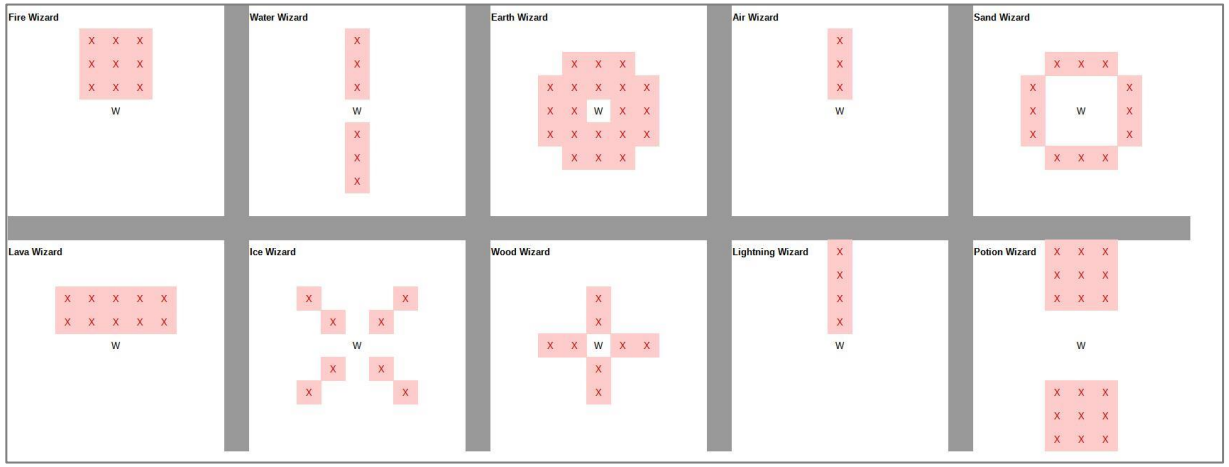
Unlike most tower defense games, the wizards attack on a rhythm, across an area, which is key to making it feel like a cartoon factory.

However, this did complicate the damage/DPS balance a bit. So I used a spreadsheet to keep track of their damage, across multiple ranks.

Then, I had to assign a cost to upgrade, keeping in mind the final “cost of DPS” figure. I used this to influence their cost-power curve. The Flame wizard formed my baseline.

Some wizards were harder to utilize effectively, so they had more damage, or were cheaper to upgrade.

Loot drops were balanced using known good figures and tiny adjustments on a per-level basis.



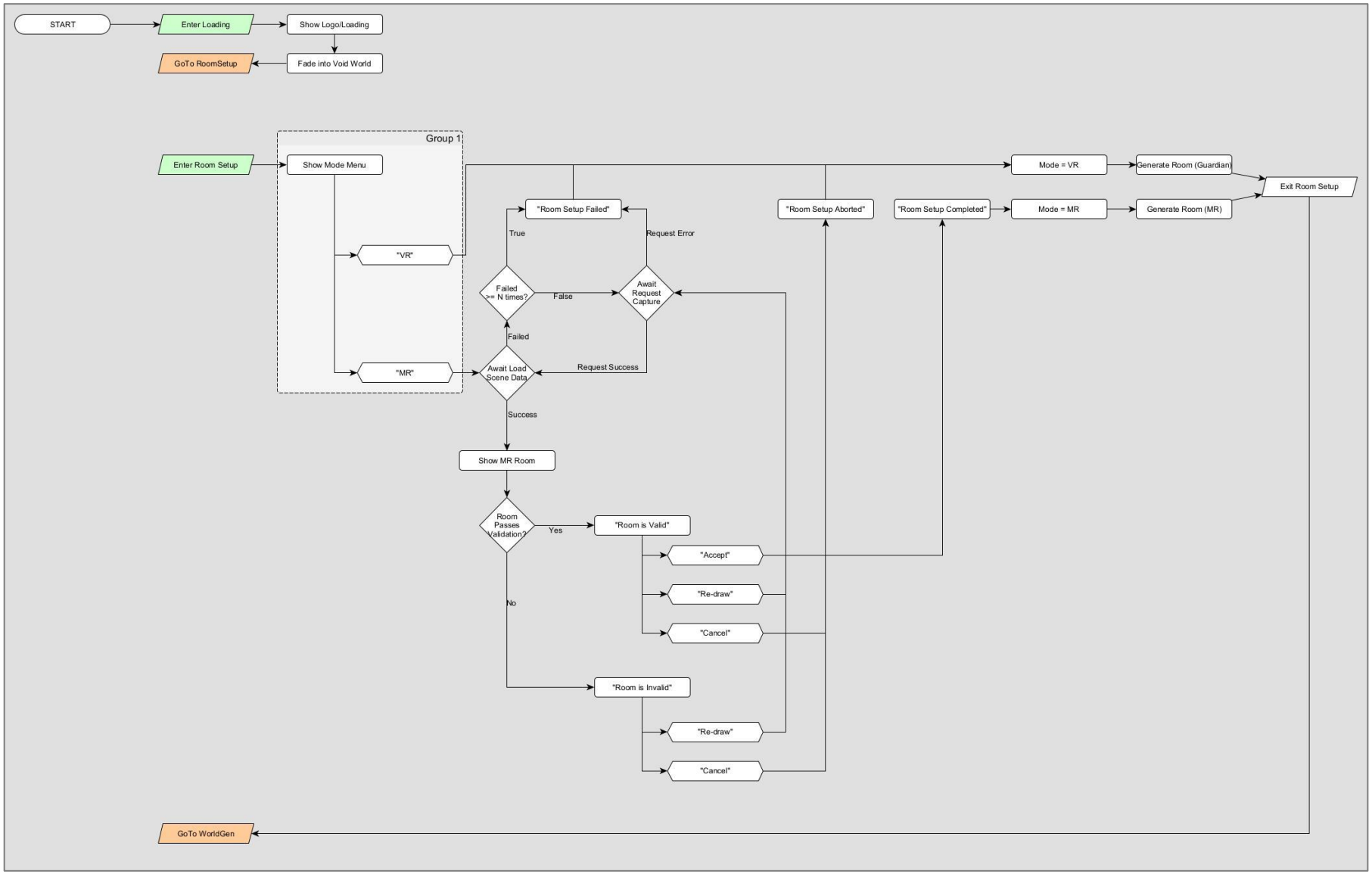
Game State

As a solo developer, I didn't often create flow charts for myself; just preferring to get it into code and work with it.

However, there were occasions when I needed one, such as this.

The Meta Quest Mixed Reality API was fairly difficult to use, as it had asynchronous functions and a variety of fail/error states.

So I used yEdit, my preferred charting tool, to lay it out.



Skill Design

In an early prototype, each wizard had 2 active skills and 2 passive skills.

I used a spreadsheet to design these before implementing them.

Looking back, I can't believe that I had all of these in the game at one point.

It would have been a nightmare to keep balanced and bug-free. Also, it was not so valuable to the player, because the wizards are AI-controlled.

Wizard Skills

Wizard	ID	Skill Name	Standard Description	Upgraded Variant
Flame	Auto	Fireball	Launch a seeking fireball that deals D damage upon hitting an enemy.	
	Ultimate			
	A	Explode	Spells explode on hit, dealing D damage to nearby enemies.	D++
	B	Scorch	Spells scorch the ground for T seconds. Scorch deals D damage to enemies each second.	T++
	C	Meteor	Learn Meteor. Summon a meteor that falls, dealing D damage to enemies in a large area.	Meteor will also stun.
Ocean	Auto	Rain	Launch an arcing ball that deals D damage upon hitting an enemy.	
	Ultimate	Geyser	Stream of water	
	A	Flood	Rain is rapidly cast in bursts of N.	N++
	B	Splash	Spells splash to heal the nearest ally for X% of damage dealt.	H++
	C	Geyser	Learn Geyser. Channel a stream of water that deals D damage for T seconds.	Double duration
Rock	Auto	Earthquake	Shake the ground below an enemy, dealing D damage at that location.	
	Ultimate			
	A	Epicenter	Spells deal more D% more damage when the enemy is nearby.	D++
	B	Mountain	Increase Max Health by H.	H++
	C	Crystal	Learn Crystal. Reflect X% damage back to the attacker for T seconds.	Crystal will also heal.
Wind	Auto	Puff	Launch an air fairy that passes through enemies.	
	Ultimate			
	A	Twist	Spells boomerang back to the caster with X% damage.	T++
	B	Whirl	Spell cooldown %X less.	X++
	C	Tornado	Learn Tornado. Summon a Tornado at location that deals D damage and knocks up enemies.	Tornado knocks back enemies.
Lava	Auto	Smash	Hammer spell that deals D damage to enemies in a cone.	
	Ultimate			
	A	Aftershock	Smash has a C% chance to stun enemies.	C++
	B	Mantle	Smash heals and increases Max Health by H.	H++
	C	Eruption	Learn Eruption. Increase attack speed by X% for T seconds.	Eruption costs less mana.
Ice	Auto	Icicle	Summon ice above an enemy, dropping to deal D damage to it.	
	Ultimate			
	A	Frostbite	Spells chill the enemy on hit, slowing attacks by S% for T seconds.	S++
	B	Freeze	Spells freeze the ground for T seconds. Freeze slows enemy movement by X%.	T++
	C	Blizzard	Learn Blizzard. Slow all enemies in the area by X% for T seconds.	Blizzard slows all living enemies.
Wood	Auto	Apple	Throw an apple at an ally or enemy, healing (% max hp) or damaging them.	
	Ultimate			
	A	Squirrel	Apples hitting a target near a plant has a X% chance to stun (enemy) or grant mana (ally).	X++
	B	Blossom	Apple hits will spawn a flower nearby (max N flowers).	N++
	C	Forest	Learn Forest. Transform into a tree, and throw lots of apples.	Each flower throws an apple.
Storm	Auto	Spark	Instantly deal D damage to an enemy.	
	Ultimate			
	A	Surge	Spells have a C% chance to Spark to nearby enemies (at D% damage)	C++, D++
	B	Shock	Spells have a C% chance to critical hit (+D% damage)	C++, D++
	C	Lightning	Learn Lightning. Deals D damage to an enemy.	Lightning strikes twice.
Sand	Auto	Dune	Create a large sand projectile that moves from the enemy back towards the caster.	
	Ultimate			
	A	Journey	Spells nudge the target a little.	X++
	B	Mirage	Negate X% of incoming damage and blind the attacker.	X++
	C	Ancient	Learn Ancient. Transform into a bright ancient spirit, which explodes to deal 50% damage after T sec	Ancient blinds enemies.
Potion	Auto	Concoction	Throw an exploding flask in an arc at the enemy, dealing D damage to nearby enemies.	
	Ultimate			
	A	Strongarm	Spells have R% longer range and higher arc.	R++
	B	Poison	Spells apply Poison, dealing D damage each second for T seconds.	D++
	C	Waste	Learn Waste. Throw a flask that explodes into an animal unit.	Summons 3 Slimes.

Tutorial Plan

Late in development, I worked on a tutorial level.

It hooked into events and objects in a level, displayed pop-up text, pointed to objects, and restricted some interactions.

My first step involved looking at the game as-is, and what actions or knowledge the player needs to know.

Then I selected a subset of elements, sorted them into groups, and then ordered them as logically as I could.

Finally, I wrote some text to go along with each step.

Tutorial Steps & Writing

Step ID	Text	Button
	Welcome to Training!	
WELCOME	This is the Training Level. You will learn how to play the game. Start by pressing Play.	(CLICK)
	Objective	
OBJ_START	So, here's your mission…	(NEXT)
OBJ_ENEMY	The Xenomancer spawns creeps at Portals. You can see the Wave details here.	(NEXT)
OBJ_ALLY	This guy has hired the Wizards to protect an Objective.	(NEXT)
OBJ_RESULT	If the Objective is destroyed, you lose! If it survives all waves, you win!	(NEXT)
OBJ_STARS	If you do a great job, you'll get a better star rating!	(NEXT)
OBJ_FINISH	Don't mess it up!	(NEXT)
	Calling a Wizard	
CALL_START	Let's call a Wizard!	(NEXT)
CALL_GEM_INFO	Calling a Wizard costs 2 Gems.	(NEXT)
CALL_GEM_LIST	There are 4 types of Gems: Fire, Water, Earth and Air.	(NEXT)
CALL_GEM_PICKUP	Here's a few free gems for you. Click to pick them all up.	(ONCLICK)
CALL_HAT	Now, to call a Wizard, click the Hat.	(ONCLICK)
CALL_ADD_GEM1	Click any gem to add it.	(ONCLICK)
CALL_ADD_GEM2	Next, add another gem.	(ONCLICK)
CALL_COMBO_RESULT	See? Combining X + Y will call the N Wizard!	(NEXT)
CALL_CLEAR	Click this to Clear the recipe.	(ONCLICK)
CALL_COMBO2_WAIT	Try a different gem combination, to call a new Wizard.	(ONCLICK)
CALL_COMBO2_RESULT	That's it! A + B will call the Z Wizard! There are 10 different combinations to try.	(NEXT)
CALL_SPELL	This is where the Wizard will attack.	(NEXT)
CALL_ROTATE	Click to Rotate the Wizard. See how the attack changes?	(ONCLICK)
CALL_PRACTICE	Now, click to Call the Z Wizard.	(ONCLICK)
CALL_FINISH	And that's how you call a Wizard!	(NEXT)
	Inventory / Gems (Advanced)	
INV_START	Here is your Inventory.	(NEXT)
INV_INITIAL	You start the level with some Gems and Gold.	(NEXT)
INV_CREEPS	Creeps always drop gold. Some creeps drop Gems.	(NEXT)
INV_FINISH	Keep a lookout for rare powerups, too!	(NEXT)
	Modify / Upgrading	
MOD_START	Wizards can be upgraded by spending Gold.	(NEXT)
MOD_WIZARD	Click the Wizard to see details and options.	(ONCLICK)
MOD_STATS	Here you can see the Rank and damage.	(NEXT)
MOD_UPGRADE	Click Upgrade now	(ONCLICK)
MOD_ROTATE	Try Rotating the wizard.	(ONCLICK)
MOD_HIDE	Click the Wizard again to cancel.	(ONCLICK)
MOD_OPEN	.. and open it again!	(ONCLICK)
MOD_REFUND	And now Refund the Wizard	(ONCLICK)
MOD_FINISH	Easy, right!	
COMPLETED	Okay! Training is completed! Click to return to the Home Map.	(NEXT)

Hub UI & Progression

Post-release, I overhauled the hub UI and progression between levels.

Although no players reported problems with the Hub, I watched a player get confused and blocked from the first moment, so I knew this was a priority.

I introduced some items that the player could find & use to unlock new worlds.

Overall, the game feels much more like a ‘real game’ and less like a confusing experiment.

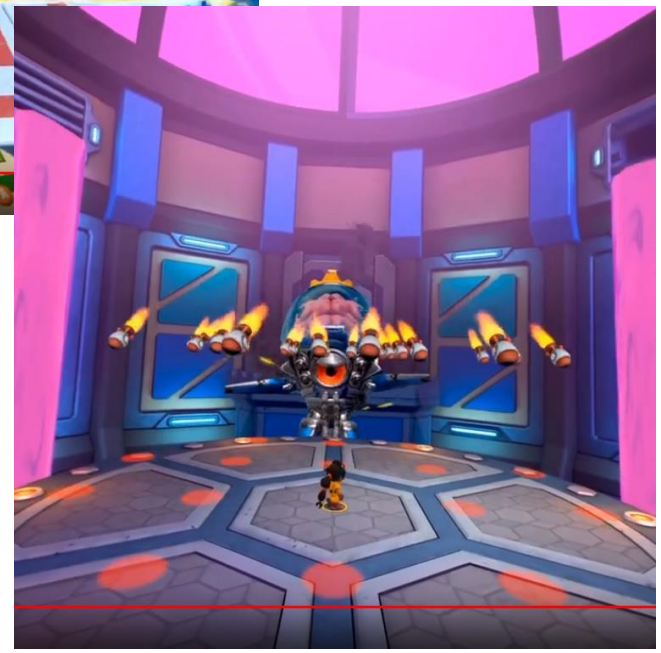
Other Projects

Max Mustard: Behaviour Trees

While working as a Unity programmer at Toast Interactive on the Gold Coast, I took on the early prototyping of the boss fights for the platformer game Max Mustard.

I used Behaviour Trees, scene objects and debug gizmos to design the placement, timing and movement of hazards/platforms.

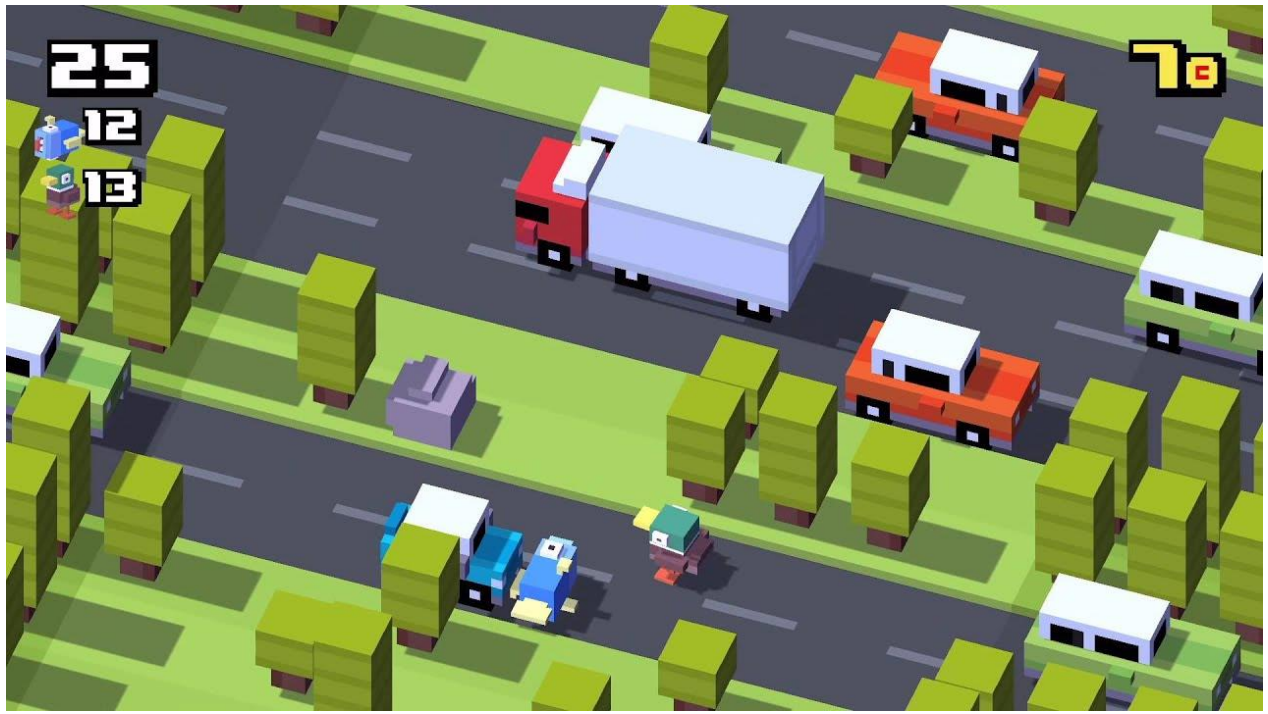
Much of the final gameplay elements emerged from those first few weeks of iteration and internal testing.



Crossy Road : Multiplayer prototypes

While working as a junior programmer at Hipster Whale in Melbourne, I took on a variety of tasks that involved design & implementation:

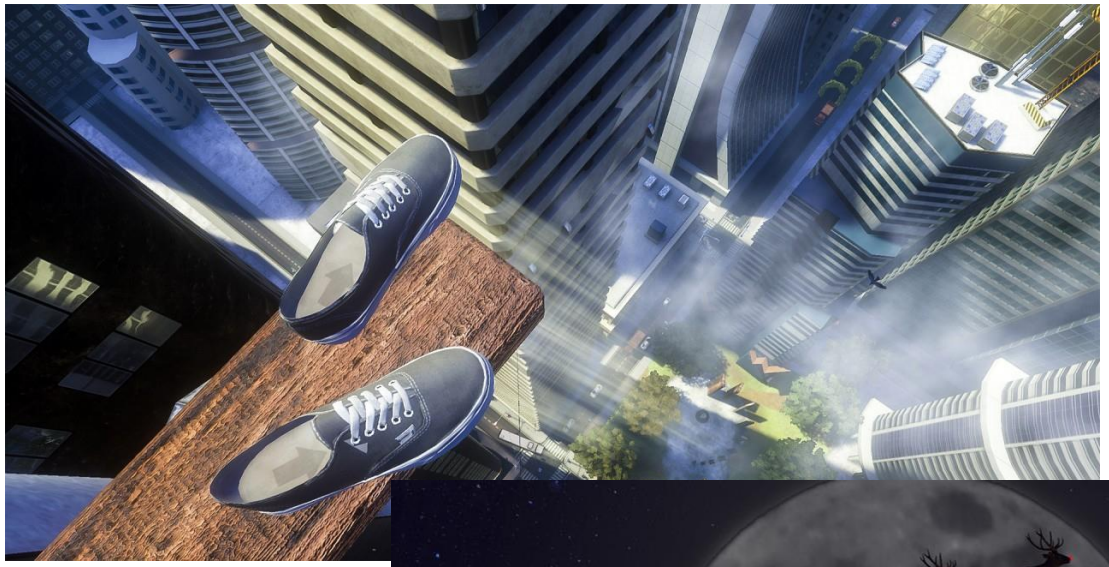
- New experimental multiplayer modes
- Grid-based map & behaviour design
- Cross-platform control UX/UI
- Documentation, cleanup and tutorials for an internal scripting language
- Company-wide game jam
- Multiplayer testing sessions



Richie’s Plank Experience: Scene Design

Ported & redesigned the ‘Christmas’ mode from the PC-VR app into the Quest VR app:

- Player travel path via splines
- Placed objectives
- Balanced scoring
- Placed and moved buildings
- Set-dressing and lighting
- Directed external artist
- Completed within 4 weeks



Unstung:
System illustrations

My bee game, Unstung, had a few unique mechanics, so I made these diagrams as a kind of tutorial.

I can appreciate the value of good graphic design to communicate ideas.

I've been using Photoshop for over 20 years; it's as natural to me as a word processor or web browser.

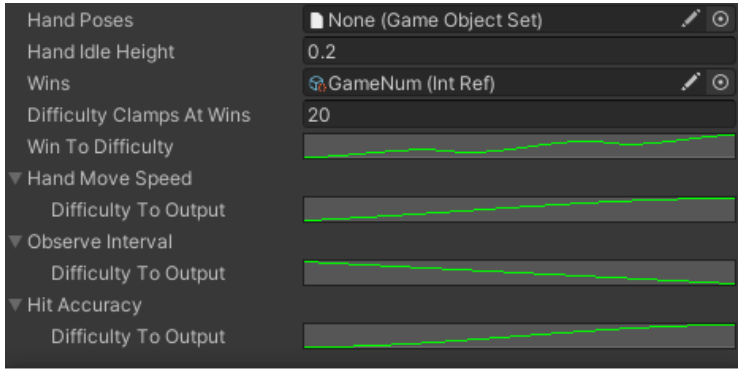
Moon Slime: Difficulty Curves

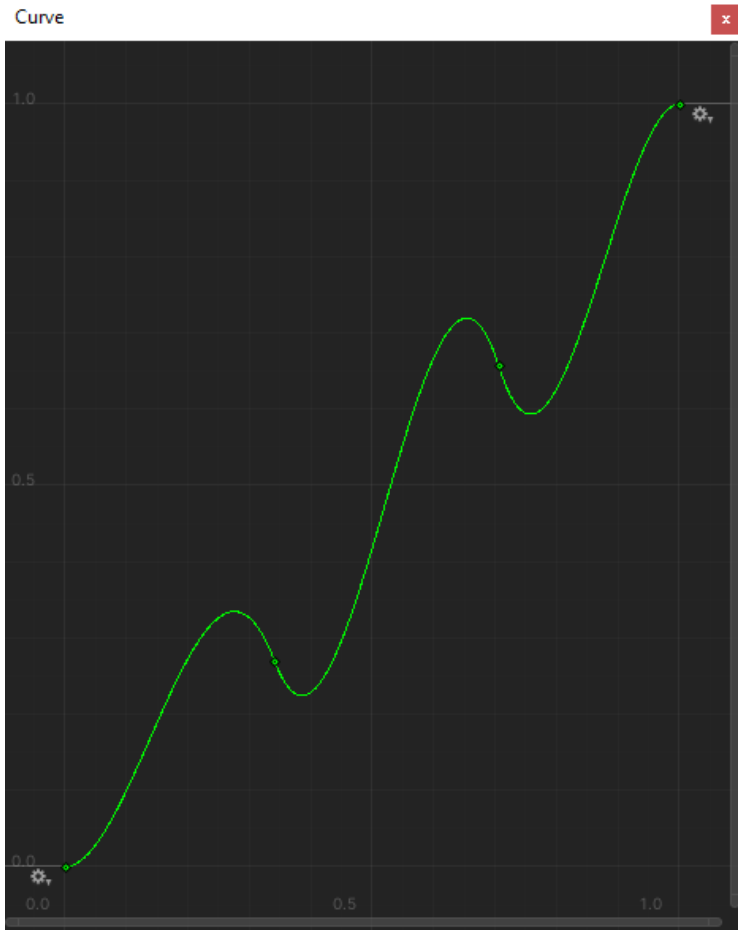
Moon Slime is a Pong-like game with difficulty that increases as the player wins matches.

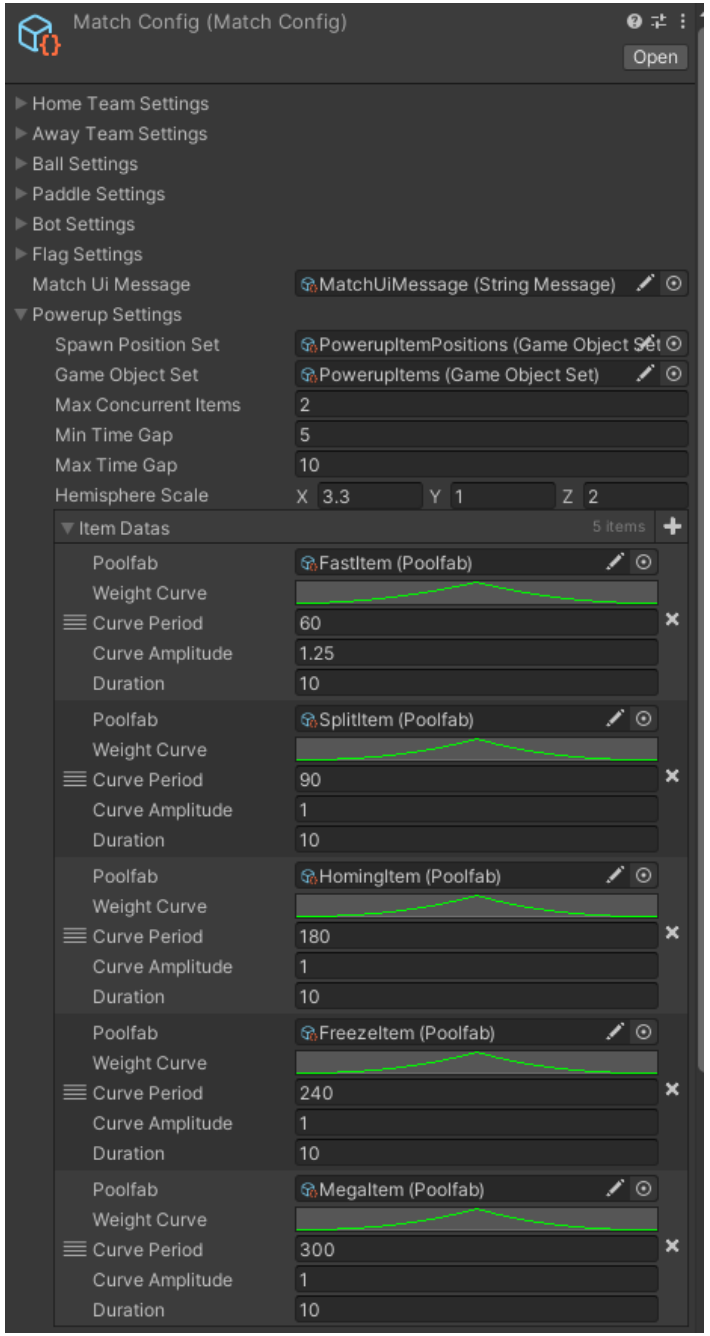
I first learned about curves while studying creative writing, and then found them again when getting into game development.

I took the output of the difficulty curve and fed it into other variables; here you can see how the difficulty value is used to influence the AI agent's speed, reaction time, and accuracy.

I also used curves with varying periods to determine powerup spawns, so the player would experience a sequence of different powerup combinations as they played over about 6 minutes. The peak is a really exciting moment!







Other Game Engines

I've been in Unity/C# land since I got back into coding in 2017. But I have dipped into other engines a tiny bit...

Godot:

I took on the role of Lead, Producer & UI designer for a Godot game jam in April 2024. I got a feel for how it works.

Unreal Engine:

I joined a rev-share project in 2020 for a few weeks, and got a little taste of Unreal 4.