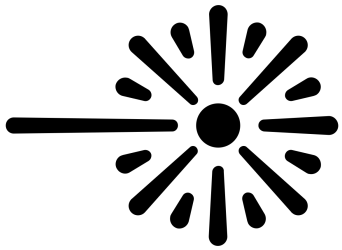
Beamdown Mechanics/Combat Doc



As a general note please refer to spreadsheets for the most updated data. They should be linked to this sheet for immediate reference and can be refreshed for the latest data.

Table of Contents

Core Design

Combat Pillars

- Purposeful Action
- Improvisation
- Tough, but fair
- Strong Feedback
- Meaningful Progression

Player Fantasy

- Player should feel powerful but not invincible. They should feel confident that they can conquer a level, but only through expressing their skill. Warrior-like, the player should be able to prevail despite setbacks.

Player Mechanics

Player Stats

- The player will have the following basic stats.
- These should be public values to be easily modified in editor.

Dev Name	Usage	Notes
Health	Flat value	

Movement Speed	Flat value	How fast the player moves
Critical Strike Chance	Random % chance 0-100	Determines if a critical strike occurs based on a %chance. Per ability
Critical Strike Damage	Flat value	Damage value dealt when a critical strike happens. Per ability

Player Effects

- Player Effects are gameplay features that happen to the player and do not require choice/input. Some of these are simple (taking damage) while future ones may be more complicated (poison, stuns, slows, etc.).

Dev Name	Notes
Stun	Temporarily stuns the player for a limited time. Button mashing reduces the time. Should be using fairly rarely.
Stagger	

Actions

- The player will have various actions they can take, some are offensive, some are defensive, and some are neutral. Certain actions will take priority over others and be able to cancel a current action (see Interrupt Hierarchy)

- Abilities have their own cooldown, any unmarked are 0 cooldown.
    - Cooldowns should be public variables so we can quickly change them in the inspector.

Basic Abilities

- Basic abilities that the player starts with and does not need to unlock. These abilities are non-attacks.

Ability	Dev Description	Narrative Description	Notes
Move	Basic Player Movement should be quick. Relatively low acceleration time. Easy to make turns/change directions.	NOT NEEDED	The player cannot move when falling
Dash	Dash in the target direction. Has invincibility frames and recovery time.		The player can dash when falling. The player can dash off of ground to cross a gap onto a different platform. Dash should reset the state of combos (ie, after doing first 2 hits of a combo if the player dashes and then attacks again they will be back to 1 not do the 3rd). Should be able to interrupt almost any other action.

Basic Attacks

- Basic attacks that the player starts with and does not need to unlock. There will be 1 physical attack combo and 2 energy attacks (that may end up in a combo)

    - Physical attacks use the blade legs while energy attacks use lightning/fire from the arms.
    - Physical attacks will have more difficult animations because of this while energy attacks will have more difficult vfx.
    - Each hit of a multi hit combo should flow together naturally in gameplay and feedback, especially animations.
    - These are the base abilities for dev purposes, the player will only be able to select a variation of these, see Ability Kits

- The Attacks Spreadsheet has example pictures for hitboxes/visuals that do not fit on this page. Player

Ability	Type	Dev Description	Cooldown
Basic Physical Attack Hit 1	**Basic Attack**	First hit of three hit basic physical combo. Stab forward with one blade leg like a karate kick	0
Flamethrower (Basic Energy Combo Hit 1)	**Basic Attack**	First hit of 2 hit basic energy combo. A quick spray of fire in an arc in front of the player	0
Lightning (Basic Energy Combo Hit 2)	**Basic Attack**	Second hit of 2 hit basic energy combo. A blast of lightning in a line in front of the player that has a less wide but further range.	0

Basic Physical Attack Hit 2	**Basic Attack**	Second hit of three hit basic physical combo. Slash with opposite leg from basic physical attack hit 1, does damage in an arc.	0
Basic Physical Attack Hit 3	**Basic Attack**	Third hit of three hit basic physical combo. Flip into a hard landing with blade legs that does damage in an area.	0

Ability Kits

- While on the ship the player will be able to select their abilities from those available.
- Each ability will have a left option, middle option (default), and right option
    - Example: Fire Dash | Basic Dash | Teleport
- The player can only select one option for each ability.
- Example augments not based on our actual abilities:
    - “Your fire breath lasts longer and inflicts a bleed on the enemies.”
    - “Dashing now leaves behind a trail of fire that lasts for 5 seconds and burns enemies.”
    - “Hitting enemies with the third attack of your basic combo stuns them for 1 seconds.”
- See Aiden Xu for more details on upgrade balance/costs and Raphael for details on crafting UI/menus.

	Upgrade Path	Dev Description	Player Feeling
Dash	Right Path Choice	The dash turns into a teleport, happening instantly.	
Dash	Left Path Choice	Dashing leaves behind a trail of fire in its path that lasts for 5 seconds and does damage every second an enemy is in it. This does NOT damage	

		the player if they cross back in.	
3 Hit Combo	Right Path Choice	Life steal for each hit	
3 Hit Combo	Left Path Choice	Every 3 hits of the combo on an enemy does bonus damage	
Lightning	Left Path Choice	The beam does more damage but has a shorter range.	
Lightning	Right Path Choice	A sphere of lightning around the player that lasts 4 seconds, it does light damage every .25 seconds and blocks projectiles.	
Flamethrower	Left Path Choice	3 balls of fire spawn around the player for 5 seconds, rotating in a circle. When an enemy is hit the damage over time is applied. Subsequent hits refresh the time of DOT (don't apply another)	
Flamethrower	Right Path Choice	Enemies take ramping damage (DOT value increases) based on the amount of time that they are in the fire but are slowed by X% while active.	

3 C’s

Camera

- Our camera should make the game feel quick and exciting by carefully framing player actions and combat.

- The player will not be able to move the camera themselves.
- The camera will follow the character and remain at a fixed angle.
- The camera should zoom in slightly when getting closer to enemies and zoom out slightly when moving away from enemies. These values should be clamped to a maximum and minimum.
    - Zoom in is to increase weight and impact of combat. Zoom out is to allow more time for the player to see incoming enemies, and more ability to notice puzzles.
    - Minimum and Maximum values for clamp should be public variables to edit in the inspector.
    - The zooms should be smooth over a period of time.
- The camera should zoom in slightly on hit to emphasize impact
- Any object that comes between the player and the camera and would obscure vision should become translucent such that the player can still be seen.

Character

- Our player character should have the highest visual fidelity of any character models in the game.
- The player character should be clearly recognizable with a silhouette from any view angle.
- Our player character should be visually distinct to other character models and environments.

Controls

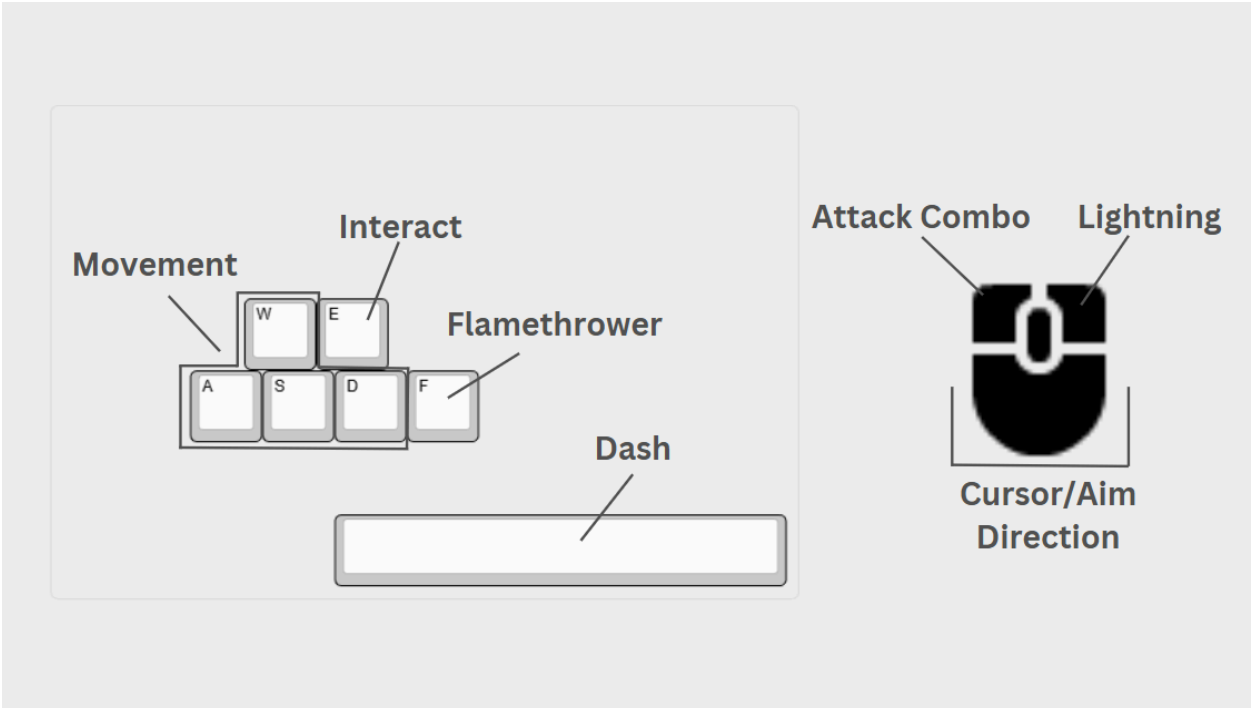
- We will be targeting controller gameplay first but will have keyboard/mouse controls for easier development and testing.
- All actions are mapped to the face buttons.
- Interact is mapped to the right bumper.
- Movement is mapped to the Left Stick.
- Controller Layouts Link
- Old Version

Beamdown Gameplay Controls

-

Beamdown Menu Controls

-



Game Feel

Responsiveness

Action responsiveness is especially important to our game to make the combat feel good. I've split various things we can do to improve the combat experience into their own sections with detailed descriptions. A lot of the stuff in the game feel section is heavily inspired by this GDC talk. When implementing please consider what variables can be public for easy iteration from the inspector. If any of these do not make sense let me (Kyle M) know and I will clarify or add visual examples.

- Interrupt Hierarchy
    - The player should have maximum control over their options and a general expectation for how the game will respond to inputs.
    - To determine what abilities should interrupt others, we will have an interrupt hierarchy from 1-3.
    - Abilities with a lower interrupt value should be able to cancel actions of higher numbers and be performed instead. If an action has the same number nothing happens.
    - We will prioritize player control over defensive responses which will always have maximum interrupt capability (1).

    - Example: Player uses an attack with an input hierarchy of 3 and then pushes the dash button (input hierarchy of 1). The attack will be canceled and the player will dash.
    - Animation blending for this can likely be 0, immediately sending the player into the interrupting action.
    - Input hierarchies can be found on the actions tab here: Player
- Generous Hitboxes
    - Our hitboxes should be fairly generous and prioritize player feel over fidelity and direct match with animation/weapons.
    - Hitboxes should be generous in relation to the player
        - Player hitboxes should be generally larger than expected
        - Enemy hitboxes should be generally smaller than expected
- Aim Assist
    - We should correct for hits at a fairly generous margin, if the player is attacking within a ~45° angle of an enemy we should snap their direction towards the enemy to ensure a hit.
    - This can likely be done with little to no animation blending.
- Magnetism/Hit Impulse (Knockback)
    - If an attack displaces or knockbacks an enemy the player should move the player a similar amount in the same direction. If an attack moves the player in a direction, it should similarly move the enemy in that direction.
    - This is to account for ability stringing/comboing.
    - This does not apply if the enemy is sent further than reasonable for the player to move in an attack (ie. the enemy is punched off screen or the abilitie's intended use is to push enemies away from the player).
- Input Buffering
    - We should buffer player inputs fairly generously to make sure the player does not feel like their inputs are being dropped. These buffered inputs should wait until the system is ready for another action and then perform without additional input.
    - We will need to make sure that button mashing is not over registered to avoid a large queue of inputs.
    - One idea (I want engineer thoughts on best ways to do this) would be to store the first input that occurs during an action and only that one. When the system is ready for another action it automatically occurs and the store is wiped such that another could be stored during that action.
- Input Latency Note:
    - If we notice high input latency we should increase the base parry window to account for it, and potentially speed up the dash/add more invincibility frames.

Feedback

- Animation
    - Animation is one of if not the most important parts of player action feedback so I have given it its own section despite how related it is.
- Hit Reactions & Enemy States
    - As much as possible our enemies should respond to player attacks appropriately. As we continue designing enemies and combat we may define enemy states to directly connect to their animations.
        - Examples would be stunning an enemy or sending them flying back.
    - Enemies should have a state machine and change state upon being attacked.
    - Recovery times for attacks against enemies will come from this and will be important for balancing stunlocking and preventing the player from being too powerful. As much as is practical should be public variables to allow quick iteration.
    - The strength of hit reactions should make sense with the strength and size of the enemy.
        - Harder enemies such as bosses or later more challenging enemies should have less reaction to player hits.
        - If needed we can define enemy classes/weight values to show which ones have stronger reactions.
- Hit Flash
    - A brief flash of the silhouette on the player and the enemy to show that a hit has connected.
    - We should be able to change the color of this fairly easily.
    - Enemies that are weak to physical attacks will flash a specific color to indicate that (likely orange).
    - Enemies that are weak to energy attacks will flash a specific color to indicate that (likely purple).
- Hit Stop
    - A temporary time pause that happens when the player attack connects.
    - It can only be caused by player attacks.
    - Freezes the player and the enemy who is hit.
    - Ideally we would be able to change the amount of hitstop per ability quickly in the inspector.
    - A good base freeze amount for testing is ~115 milliseconds.

    - Note to engineers: If possible when creating this system it would be great if we could extend it beyond just hits and allow us to do more with time such as briefly slowing it for anticipation or in an ability.
- Hit VFX
    - VFX should match the power, type, and style of the move. All moves should have visual indicators that the move is occurring (largely supported by animation) and VFX indicators when a hit happens. These should complement the animation.
    - EX: A sword swing has a trail, when it connects there is a visual indicator on the enemy.
    - Hit VFX should also play when the player is hit, however this VFX should be smaller and of course complement the animation. We want to avoid creating too much visual noise.
- Screenshake
    - Different types and strengths of attacks will have varying screen shakes.
    - We will take great care to differentiate screenshake from enemy attacks and from player attacks.
        - When the player is hit the camera shake should feel more disruptive.
        - When the player is hitting enemies the effect should be far less strong.
    - Ideally this should be changeable in the inspector and on a per ability basis for player attacks and enemy attacks.
- Controller Feedback
    - Almost all instances of screenshake should have controller feedback.
    - Similar to screenshake for player hits vs player getting hit.
    - Also should be changeable in the inspector and on a per ability basis for player attacks and enemy attacks.
- Audio
    - Our audio feedback should be weighty and match the strength, depth, and impact of the attack animation.
    - Audio should match the type of attack well (ie. fire should sound like fire not like a sword swing).
- Enemy Shake
    - Applying a shake on the root or skeleton bone of an enemy to emphasize impact. This should be especially used on enemies that don’t have hit reactions (can’t be stunned, etc.).
    - Applied to yaw and x/y axis.
    - If we have scope concerns about feedback we may be able to replace hit reactions with this and make it more drastic for weaker enemies.

- Damage note
    - Damage values should match the feel and strength of an attack and vice versa. Weightier attacks with more hit stop, feedback, etc. should do more damage.

Animation

- I highly recommend the animation sections of this GDC talk that discusses these in far greater detail.
- Leverage principles of animation.
- General Animation (Especially for player)
    - Anticipation and follow through.
        - Player moves, even when performed quickly, should have clear anticipation and follow through on animations
        - These should generally try to follow semi-realistic physics as much as possible.
        - We should aim to create exaggeration to make the abilities feel believable and sell the feeling of strength on them.
    - Secondary/Overlapping Action
        - Concept that having too many elements perfectly in sync can feel unnatural and stiff to the player.
        - Something we can do to aid this is add elements that trail (examples being scarfs, cloths, hair) that are simulated but not directly part of the animation and thus create a sense of aliveness.
    - Slow In/Slow Out
        - Easing the animations in and out at a varied rate will increase believability of them and convey weight/inertia better.
    - Exaggeration & Strong Posing
        - Attacks and abilities should be exaggerated and have strong posing. This will allow us to get away with spending less time on certain parts and having lower fidelity as the player can read the pose and process the information at a glance.
    - Damage values should match the feel and strength of an attack and vice versa. Weightier attacks with more hit stop, feedback, etc. should do more damage.
- Enemy Specific Animation
    - Telegraphing
        - Enemy incoming attacks should be clearly telegraphed to the player.
        - This does not mean they need long windups, just visual clarity.
        - Attacks should have clear poses and very strong anticipation.

        - At later difficulties we can use this to subvert expectations and increase challenge.
    - If 3 different enemies are on the screen attacking, it should be clear from the animation alone which is most dangerous.
- Recovery Time
    - We should build recovery time into the animation which will make it feel more natural and fluid.

Implementation Checklist

- ☑ ~~Interrupt Hierarchy~~
- ☑ ~~Aim Assist~~
- ☐ Magnetism
- ☑ ~~Hit Impulse (knockback)~~
- ☑ ~~Input buffering~~
- ☐ Hit reactions
- ☑ ~~Hit flash~~
- ☑ ~~Hit stop~~
- ☑ ~~Hit vfx~~
- ☑ ~~Screenshake~~
- ☑ ~~Controller feedback~~
- ☑ ~~Audio feedback~~
- ☑ ~~Camera zoom with enemies~~
- ☑ ~~Camera zoom on hit~~
- ☑ ~~Camera-obscuring object translucency~~

Environmental Mechanics

Planets

- Each planet will include some type of simple environmental hazard or mechanic that is unique to it.

Mechanic	Description	Planet Introduced	Notes

Falling Pillar	A pillar that falls and becomes terrain that the player can walk across.	Planet 1	
Waterway	Paths of water that slow entities that enter them, including the player.	Planet 2	Stretch goal: If electricity comes in contact with them the waterway becomes electrified, doing damage each second an entity is inside. This expires after 5 seconds and the water returns to normal.

Sound Effects

Player Sound

- Player sounds for abilities should match the weight, damage, and animation of an ability.
- Sound effects should match the type of the attack (make fire sound like fire not a gunshot).
- When the player hits an enemy a sound effect should play.

Enemy Sound

- Enemy's should have sound cues before attacks to compliment their telegraphed animation anticipation.
- Enemy hits should have a sound effect on the player to indicate being hurt.