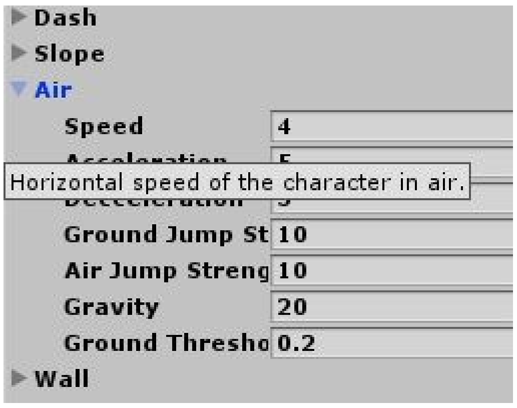
2.5D Platformer

All properties contain tooltips that can be seen by moving the mouse over labels. Some properties are accessed by rolling down sections inside the Inspector.



```
▶ Dash
▶ Slope
▼ Air
    Speed              4
    Acceleration       5
Horizontal speed of the character in air.
    Deceleration       5
    Ground Jump St     10
    Air Jump Streng    10
    Gravity            20
    Ground Thresho     0.2
▶ Wall
```

Video Tutorials

https://www.youtube.com/watch?v=855U2myarm8&list=PLT1IN78hbXgI8bK9qNT1svjr9qaXN8pZA

Character Motor

Characters must have a Character Motor component attached. It manages the character, it’s movement, appearance and use of weapons. It handles gravity and therefore gravity should be turned off in the RigidBody component to avoid conflicts.

Motors also perform IK (inverse-kinematics) when hanging on edges and ropes. IK is calculated by adjusting bones until certain objects reach defined targets. Target objects must be part of the skeleton in order for changes to modify their transform.

Characters can have a weapon and a shield. They are specified in the Inspector and should be attached to hand bones. Weapons used by characters must have a Weapon script added to their objects.

Characters have a fallen state in which they lie on the ground and have to get up. It is caused by weapon attacks or environment triggers.

Weapons

Weapon script specifies damage done by an attack or a combo. It also manages trails.

Each weapon has a simple attack and a combo. Simple attacks are performed when the character is running or in the air. Combos are performed when the character is standing still on the ground. A combo is optional and can be turned off (by setting its length to zero), in which case a simple attack will be always performed.

Attacks not only have damage associated with them but also dashes and pauses. If an attack contains a dash the character will move forward. A strong dash causes enemies to fall upon collision with the attacker. If the attack contains a pause it causes the time to freeze for a duration of time. For unaffected characters to be exempt from that freeze add a Character Time component to them.

Player Controller

Takes player input and translates that to Character Motor commands. On touch-screens player input is generated using canvas objects containing components like Touch Movement or Touch Attack. Those canvas components must be linked with the Player Controller.

AI Controller

Makes the character patrol and attack enemies at close distances. Patrols are performed between two points. When attacking the AI will run towards the enemy and start hitting.

AI performs attacks while waiting for “Attack Wait” duration of time. It can be setup to restart its wait when hit by an enemy.

Platformer Camera

Maintains camera position and orientation arounds its target.

Tries to keep the target object inside the screen. The behaviour can be controlled by adjusting the Borders property which controls how close to a screen border the target has to be for the camera to move.

Camera zoom (distance to the target) can be set by commands.

Turn

Makes characters moving through the trigger change their movement direction. Turns have defined enter and exit angles and work both ways, when moving backwards the meaning and angles of enter and exit switch places.

Orientation of the turn object itself affects the angles defined in the inspector.

Rope

Each joint of a rope has to have a rope component attached. When a character motor collider enters its trigger area the motor enters rope mode.

Edge

Denotes an edge a character can grab and hang on.

The orientation matters - a character can only grab when on the front side. Orientation is depicted in the editor by a red line that extends from the front.

Size of the collider box affects where the character puts its hands.

Triggers

Death Trigger

Causes an entering character to die.

Fall Trigger

Causes an entering character to enter a fallen state from which they have to get up.

Various character components

Character Effects

Plays effects on certain character events like footsteps, death, hits etc. Effects are played by cloning and activating a prefab object.

Character Health

Keeps track of character’s health and registers taken damage.

Character Inventory

Keeps track of character’s inventory. Each item is depicted as as a string value (name) with a number. To add items to the inventory use either C# scripts or the command system provided with the whole system.

Character Name

Stores the character name to be used in the UI. If no Character Name is present the name is taken from game object itself.

Character Platform

Makes an object (not necessarily a character) to stay on the same spot on a moving platform.

Character Side

Used by AI to tell which character is a friend or enemy. When no Character Side is attached to the object the presence of an AI Controller is used to see if the object is a player for AI to attack.

Character Sleep

Turns off some components of character object when it is far away from the player's character. Components are turned on when the player approaches the character.

Character Time

Used to keep characters unaffected by a combo moving and animating when the whole world is paused.

Commands

Commands are instruction lists executed on some events. Commands are executed by one of the components listed below when the conditions are met.

Components

On Action

Prompts the player to perform an action. Executes its commands when the player performs the prompted action.

On Arrive

Executed when a moving object reaches its destination. The event is called by Move Towards and Move Towards UI components.

On Death

Executed on character's death.

On Enter

Executed when an object enters the trigger area owned by the command component.

On Leave

Executed when an object leaves the trigger area owned by the command component.

Scripting

Each scriptable components is divided into two sections - conditions and commands.

Conditions depict requirements for the commands to be executed. They are most often used to check if the actor (object that triggered the script) is the right game object.

Conditions can be checked and commands executed on multiple objects at the same time.

Available command/condition targets:

- Self - target is the object containing the script.
- Actor - target is the object that triggered the script (for example, On Enter would have actor as the object that entered its trigger area).
- Specific - target is an object picked from the scene.
- Tag - all objects with the tag are targets.
- Find - all objects inside a hierarchy of a depicted object with the matching name are picked as targets.

UI

Enemy Display Manager

Creates and manages objects with Health Bar for all enemies visible on screen.

Health Bar

Takes a relative health from a Character Health component and displays a bar that displays the value.

Inventory Count

Displays a count of a specified item type that is stored inside a Character Inventory component.

Move Towards UI

Moves a UI element towards a specific canvas position. Upon arrival tries to execute On Arrive if it is attached to the object.

Touch Action

Registers touches and causes an object with an On Action component that’s close to the player to execute its commands.

Touch Attack

Registers touches and causes Player Controller to attack.

Touch Jump

Registers touches and causes Player Controller to make a jump attempt.

Touch Movement

Takes directional movement input from the touch screen and passes it to Player Controller.

Extra

Weapon Effects

Weapon effects are caused by an additional Weapon Effects component that instantiates prefab objects as effects during certain weapon events.

Trail

Builds a trail mesh that follows an object. Used for weapons and fast moving objects.

Cycle

Moves and rotates an object back and forth between two states. Can be used for moving platforms or revolving doors.

Delayed Disable

Disables an object after a certain amount of time passes.

Move Towards

Moves an object towards a point in world space. Upon arrival tries to execute On Arrive if it is attached to the object.

Random Audio

Picks a random audio sample from the supplied list and sets Audio Source to play it.

Reset on Death

Reloads the level upon player’s death.

Scale

Scales an object till it reached the target size.