Table of Contents

WELCOME TO DEFTLY

Thanks for purchasing Deftly by Cleverous!
This is a basic Getting Started document. It contains information to build the fundamental prefabs.

BEFORE YOU GET STARTED

The way Deftly works is it gives you a few very open-ended scripts that you are required to populate with data. It handles all of the heavy lifting in the background and does so in a fashionably dependable and expedient manner.

No scripting is required to use the basic features that Deftly offers.

You should expect to get out of Deftly what you put in. As the game developer in this equation you will be responsible for providing all of the models, effects, graphics, sounds, animations and other trinkets that you would like to see in your game and Deftly will breathe life into it.

You can expect a good front end for designing cool features, such as:

- Characters can be designed with the Subject script.
- Weapons can be designed with the Weapon and Projectile script.
- Controls are exposed by the Deftly Camera and PlayerController scripts.
- AI can be designed with the Intellect script.
- Future updates will give you even more options, more control and user-requested features.

The Demo scene provides examples for all of that. Please take a look at it and study the example prefabs.

The folder structure is in such a way to allow you to strip features you don't want quickly.

Assets\Deftly\Core contains the core script components you will need to use it. This folder is the barebones.
Assets\Deftly\Demo contains many example prefabs, some example content and miscellaneous scripts for demo purposes.
Assets\Deftly\Co-Op contains the Co-op demo content, a few more miscellaneous scripts and such.

If you want to start from scratch you can simply erase everything except the Core folder and proceed to creating your own content, scripts and such without any demo content getting in the mix.

DONT STRUGGLE!

If you are struggling with some feature, found a bug, hate something in Deftly or anything like that then please respond to the official thread on the Unity forums or contact me directly by one of the ways outlined in the Contact section so we can make Deftly work for you.

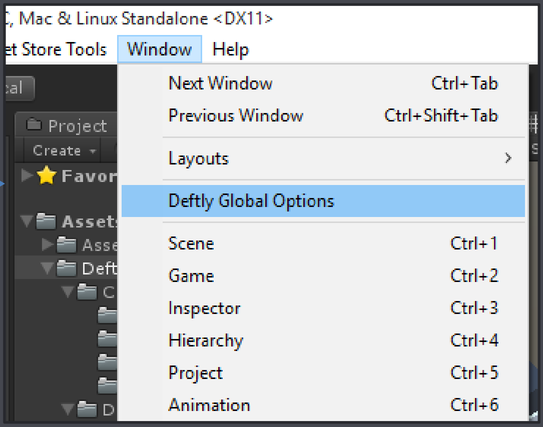
This is an early release product and is a great time to carve in the features YOU want to see in it, so make yourself heard!

CONTACT INFORMATION AND WEBSITE LINKS

Official Website:	http://www.cleverous.com/
Online Documentation:	http://www.cleverous.com/#!tds-getting-started/cht7
Release Notes:	http://www.cleverous.com/#!Deftly-release-notes/c1cdb
Trello Roadmap:	https://trello.com/b/Wc9Zqt6r/cleverous-tds-kit-Deftly
Email:	cleverousdev@gmail.com
Skype:	zer0bounds

Use Skype if you need time-sensitive answers, otherwise use email.

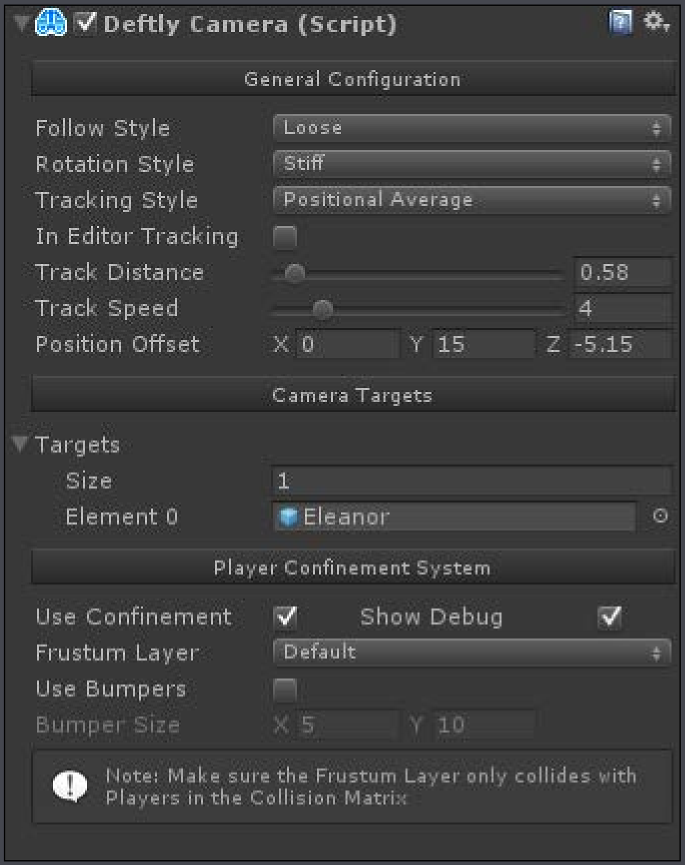
Use Deftly Global Options to customize a variety of options like Floating Text, Weapon AutoSwapping and more



GETTING STARTED

Setup the Camera because it is the all powerful game overlord

I. On your Main Camera (which should be tagged as Main Camera) add a Deftly Camera script
II. Increase "Targets" to 1
III. Drag your character from the Scene into the Targets slot
IV. Changes to the actual transform of the camera now do nothing. They are controlled by the script.
V. Instead, modify the Offset parameters. This is the distance in each direction from the Target Position
VI. The "Target Position" is the *average position of all targets* so if you have more than one target it will be looking at the average weighted position between them.



Deftly Camera (Script)

General Configuration

Follow Style	Loose
Rotation Style	Stiff
Tracking Style	Positional Average
In Editor Tracking	☐
Track Distance	0.58
Track Speed	4
Position Offset	X 0 Y 15 Z -5.15

Camera Targets

Targets	
Size	1
Element 0	Eleanor

Player Confinement System

Use Confinement	✓
Show Debug	✓
Frustum Layer	Default
Use Bumpers	☐
Bumper Size	X 5 Y 10

Note: Make sure the Frustum Layer only collides with Players in the Collision Matrix

In 0.7.1 the camera has been updated with Frustum Constraints which can contain objects on a certain layer, make sure your Collision Matrix supports this – it is setup in the Layer settings when importing Deftly but if you choose to not import these settings you'll have to adjust your Matrix manually.

Adjust it manually means whatever layer you choose to put the frustum walls on (the Frustum Layer dropdown shown in the image) you'll go to your Collision Matrix in the project's physics settings and set that layer to collide with ONLY what you want to constrain inside the screenspace and exclude everything else.

HOW TO MAKE A CHARACTER

If it dies, its a Subject

Currently requires that you are using a configured Deftly Camera

To create a Player you need a rigged and Mecanim-ready Humanoid character model.

I. Create an Empty GameObject (this is the "Container" object)
II. Place your Character Model as a Child of it, adjust as needed.
III. On its Animator component choose an Animator Controller and disable Root Motion.
IV. Back to the Container Object... Add a Capsule Collider and scale it to fit.
V. Add a PlayerController script. (This forcefully adds a Subject and Rigidbody script for you.)
VI. Done with adding components

NOW WE NEED TO CUSTOMIZE THIS SUBJECT

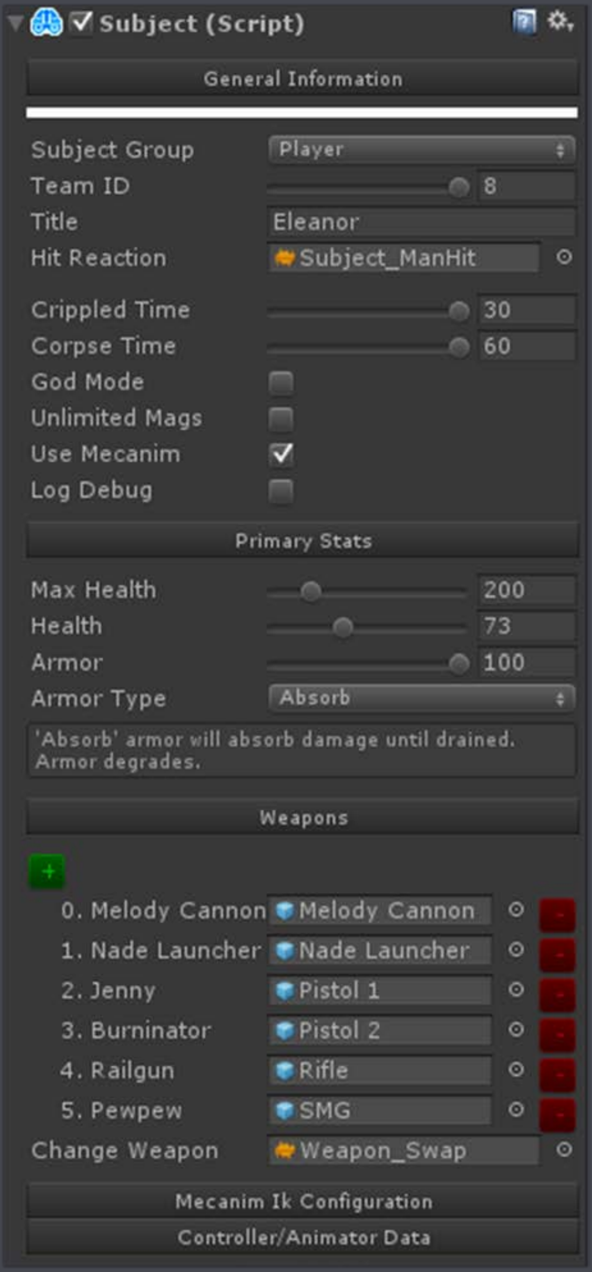
The majority of default values are ok, but we'll need to specify some settings that are unique to your project, like layers, masks, etc.

I. First tell the Subject where the Animator is. Drag your model into the Animator Host Obj field.
II. On the Player Controller script, specify the Mask field (all layers the cursor can aim to, usually just "Floor, Environment" or something.)
III. Specify your Player as one of the Targets in the Inspector.
IV. Now you can press play and control your character. (if you have a camera)
V. Make it a prefab (Drag to Project Hierarchy)
VI.

YouTube Tutorial for setting up IK features

POSSIBLE PROBLEMS

- Animator Host Obj is undefined/incorrect
- Animator Controller was not specified on the Model's Animator.
- Floor is not actually on the Floor Layer (check Mask in PlayerController) or missing a collider
- Player is below the floor or under/inside a collider
- Health starts at zero.
- Character Model import settings not set to "Humanoid" or Root Motion (Animator component) is enabled.
- Subject toggle option called "Use Mecanim" was turned off



Subject (Script)

General Information

Subject Group	Player
Team ID	8
Title	Eleanor
Hit Reaction	Subject_ManHit
Crippled Time	30
Corpse Time	60
God Mode	☐
Unlimited Mags	☐
Use Mecanim	✓
Log Debug	☐

Primary Stats

Max Health	200
Health	73
Armor	100
Armor Type	Absorb

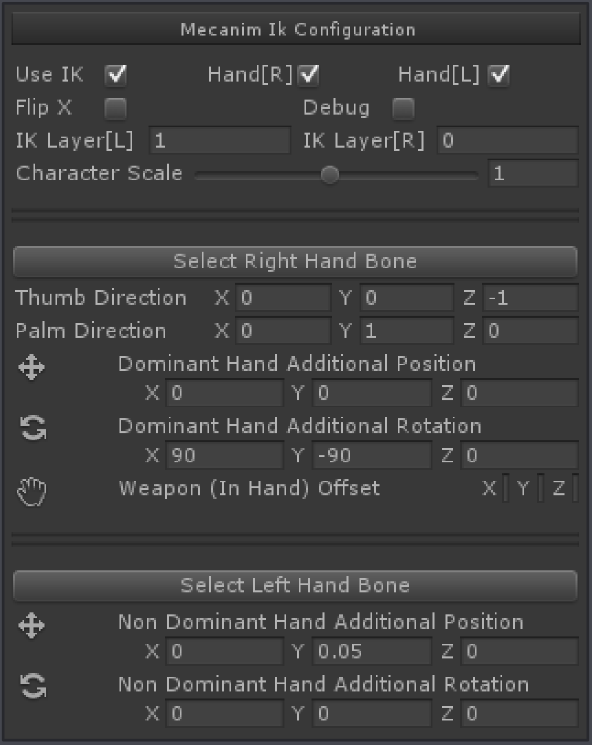
'Absorb' armor will absorb damage until drained. Armor degrades.

Weapons

0. Melody Cannon	Melody Cannon
1. Nade Launcher	Nade Launcher
2. Jenny	Pistol 1
3. Burninator	Pistol 2
4. Railgun	Rifle
5. Pewpew	SMG
Change Weapon	Weapon_Swap

Mecanim Ik Configuration

Controller/Animator Data



Mecanim Ik Configuration

Use IK ✓ Hand[R] ✓ Hand[L] ✓
Flip X ☐ Debug ☐
IK Layer[L] 1 IK Layer[R] 0
Character Scale 1

Select Right Hand Bone

Thumb Direction X 0 Y 0 Z -1
Palm Direction X 0 Y 1 Z 0
Dominant Hand Additional Position
X 0 Y 0 Z 0
Dominant Hand Additional Rotation
X 90 Y -90 Z 0
Weapon (In Hand) Offset X Y Z

Select Left Hand Bone

Non Dominant Hand Additional Position
X 0 Y 0.05 Z 0
Non Dominant Hand Additional Rotation
X 0 Y 0 Z 0

HOW TO MAKE A RANGED WEAPON

These go pew pew

I. Make an Empty GameObject (container object)
II. Add your model as a Child of it.
III. On the container object add a Weapon script

You'll see a couple of red errors in the Inspector. First, you must have a Spawn Point for the Projectile to be created at. The other is regarding IK, you cannot have a Weapon origin at 0,0,0, there must be some offset from the bone it is linked to.

Be mindful of the position

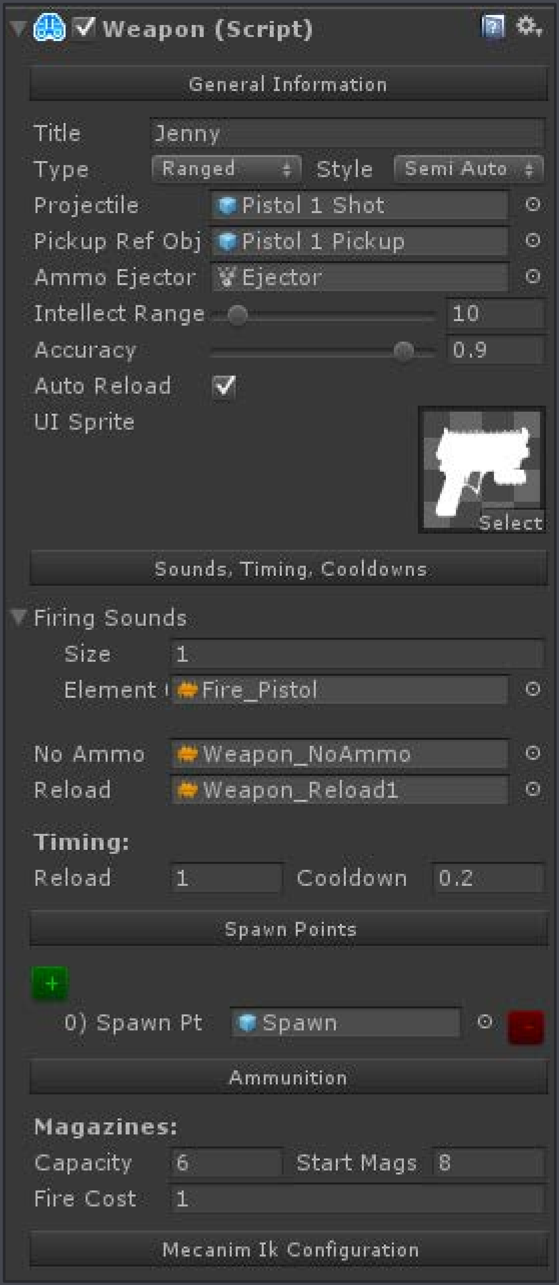
... that you choose for the model as a child of the wrapper.
The Weapon GameObject's will be put as a child of the Hand Bone of the Subject that owns it and zero'd out. This means that the pivot of the Weapon prefab will be exactly on the pivot of the Hand Bone. You can specify additional offset, for instance a small offset to move it into the palm better is basically standard practice here.

I. Customize and populate the basic fields (Audio clips, accuracy, title, magazines, etc..)
II. Create an empty GameObject, make it a child, place it near the barrel.
III. Under "Projectile Spawn Points" press the [+] button and put your spawn point in the new field.
IV. Create another empty GameObject to be used as a target for the Left Hand.
V. Choose the "Pivot From" and input an offset from which this gameobject will stand off from that bone.

Now lets setup the IK stuff...
1. Weapon Resting Position Offset
   a. Position tweaking for Weapon relative to Hand
2. Dominant Hand Additional Rotation
   a. not working yet =(
3. Dominant / Non Dominant Elbow Goals
   a. Positional tweaks for the elbow positions.
4. Non Dominant Hand Goal Object
   a. An object representing pos/rot goal for the Left Hand

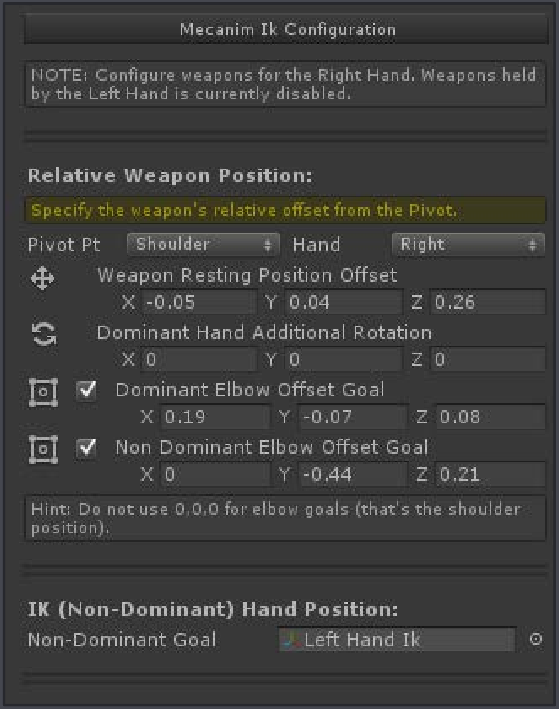
We now have a Weapon, but no Projectiles for it to fire! We need a prefab to put in "Projectile" field in the upper right of the inspector.

Make this Weapon a prefab. (Drag into Project Hierarchy)


Weapon (Script)
General Information
Title: Jenny
Type: Ranged   Style: Semi Auto
Projectile: Pistol 1 Shot
Pickup Ref Obj: Pistol 1 Pickup
Ammo Ejector: Ejector
Intellect Range: 10
Accuracy: 0.9
Auto Reload: ✓
UI Sprite

Sounds, Timing, Cooldowns
Firing Sounds
Size: 1
Element: Fire_Pistol
No Ammo: Weapon_NoAmmo
Reload: Weapon_Reload1
Timing:
Reload: 1   Cooldown: 0.2
Spawn Points

0) Spawn Pt: Spawn
Ammunition
Magazines:
Capacity: 6   Start Mags: 8
Fire Cost: 1
Mecanim Ik Configuration


Mecanim Ik Configuration
NOTE: Configure weapons for the Right Hand. Weapons held by the Left Hand is currently disabled.
Relative Weapon Position:
Specify the weapon's relative offset from the Pivot.
Pivot Pt: Shoulder   Hand: Right
Weapon Resting Position Offset: X -0.05 Y 0.04 Z 0.26
Dominant Hand Additional Rotation: X 0 Y 0 Z 0
✓ Dominant Elbow Offset Goal: X 0.19 Y -0.07 Z 0.08
✓ Non Dominant Elbow Offset Goal: X 0 Y -0.44 Z 0.21
Hint: Do not use 0,0,0 for elbow goals (that's the shoulder position).
IK (Non-Dominant) Hand Position:
Non-Dominant Goal: Left Hand Ik

HOW TO MAKE A PROJECTILE

Similar to making a Weapon.

I. Make an empty GameObject.
II. Add a Projectile script to it.
III. Configure basic parameters (damage, speed, etc)
IV. Add a rigidbody and disable Gravity. You could also lock rotation on all axis.
V. Add a Collider

Thats all there really is to it, you can refer to the tooltips on fields to better understand what each one does.

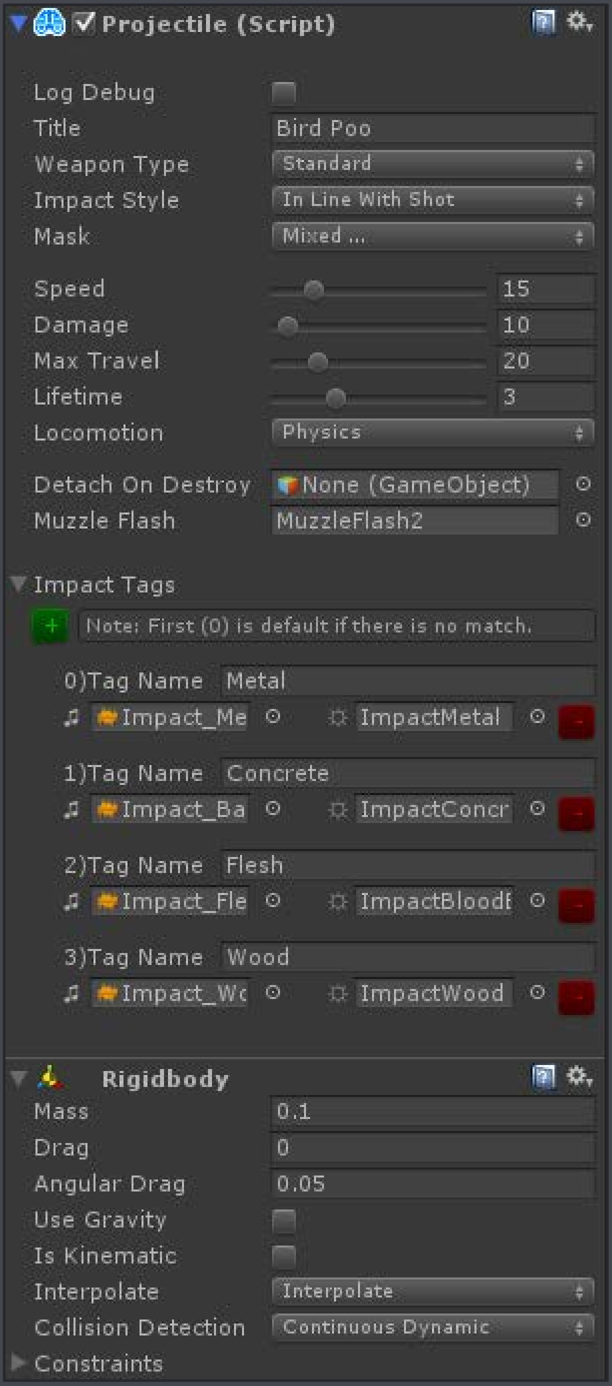
However, I'll briefly explain the tags system.

Hit [+] to add a tag filter. Notice there is Tag Name, Audio Clip and GameObject fields. You'll type in the name of the Tag and specify the corresponding effects you want it to use when it hits something on that tag.

For instance "Wood" for the name, "Wood_Impact" audioclip and "Breaking Wood Particles" prefab as the gameobject field. You'll change the tag of the wooden game objects to Wood and the Projectiles will react accordingly. The first tag is the fallback effect if something unrecognized is hit.

NOTE: It's possible that we'll move to a global control panel for these Impact Tag groups and you'll just identify which category each projectile will inherit from... Communicate if you have concerns about this possible change.

Make this a Prefab (Drag into Project Hierarchy).



Projectile (Script)

Log Debug	☐
Title	Bird Poo
Weapon Type	Standard
Impact Style	In Line With Shot
Mask	Mixed ...
Speed	15
Damage	10
Max Travel	20
Lifetime	3
Locomotion	Physics
Detach On Destroy	None (GameObject)
Muzzle Flash	MuzzleFlash2

Impact Tags

Note: First (0) is default if there is no match.

0)Tag Name Metal	Impact_Me	ImpactMetal
1)Tag Name Concrete	Impact_Ba	ImpactConcr
2)Tag Name Flesh	Impact_Fle	ImpactBloodE
3)Tag Name Wood	Impact_Wc	ImpactWood

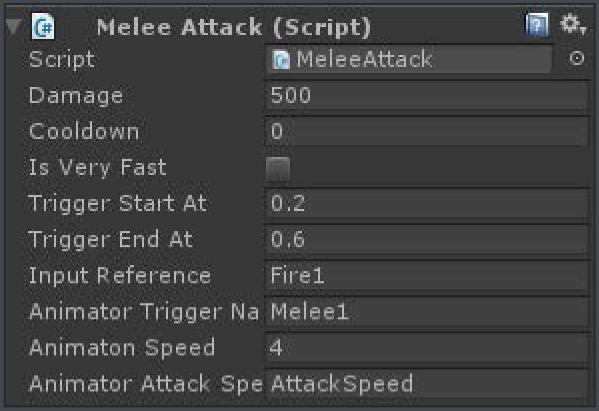
Rigidbody

Mass	0.1
Drag	0
Angular Drag	0.05
Use Gravity	☐
Is Kinematic	☐
Interpolate	Interpolate
Collision Detection	Continuous Dynamic

Constraints

HOW TO MAKE A MELEE WEAPON

Melee Weapons are super unpolished and prone to glitches right now



Melee Attack (Script)

Script	MeleeAttack
Damage	500
Cooldown	0
Is Very Fast	☐
Trigger Start At	0.2
Trigger End At	0.6
Input Reference	Fire1
Animator Trigger Na	Melee1
Animaton Speed	4
Animator Attack Spe	AttackSpeed

First, get acquainted with how Ranged Weapons work and then take a whirl at Melee Weapons. They're not complete but do work in function. They operate differently in that they require instead of a Projectile, they use a Melee Attack. The Melee Attack inspector is not custom yet and some features are not enabled.

First change the Type from Ranged to Melee.

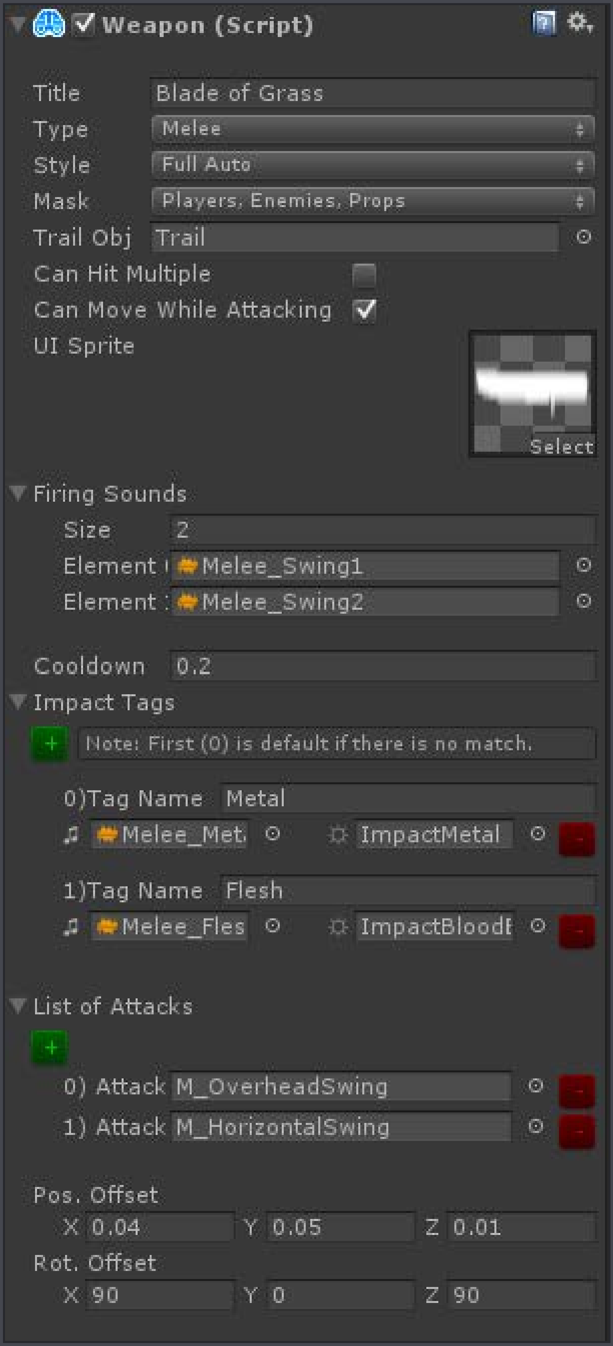
The Melee Weapons use a Trigger instead of Collider, when the trigger passes through a collider it will try to damage it. In order to prevent the Weapon from 'touching' things (for example swiging the Weapon back to prepare for a big forward attack) and damaging unintentionally you should specify a smaller time window for the Trigger to be enabled.

When the animation plays, the Melee Attack will turn on/off based on how complete that animation is from zero to one (start to end).

In the future we'll make some alternative ways to do this and improve the polish on Melee Weapons.

- **Cooldown**: How long after an attack before you can attack again.
- **Trigger Start At**: Percentage of animation where the trigger will turn on.
- **Trigger End At**: Percentage of animation where the trigger will turn off.
- **Animator Trigger Name**: What Trigger you want to send to the Animator Component. Your Animator Controller needs to be configured to accept these inputs and fire the correct trigger.
- **Animation Speed**: The numerical playback speed multipler.
- **Animator Attack Speed**: The name of the variable that controls the animation's playback speed. This should be linked in the animation's state to this variable. It's a fairly new feature (5.1.2 I think), so you probably aren't aquainted with it just yet.

Once you make your Melee Attack a Prefab then you can throw it into the Weapon script's List of Attacks category . Refer to the demo "Blade Of Grass" and its Melee Attack's for a working example.



Weapon (Script)

Title	Blade of Grass
Type	Melee
Style	Full Auto
Mask	Players, Enemies, Props
Trail Obj	Trail
Can Hit Multiple	☐
Can Move While Attacking	✓
UI Sprite	Select

Firing Sounds

Size	2
Element (	Melee_Swing1
Element :	Melee_Swing2

Cooldown	0.2

Impact Tags

Note: First (0) is default if there is no match.

0)Tag Name	Metal	
	Melee_Met	ImpactMetal
1)Tag Name	Flesh	
	Melee_Fles	ImpactBloodI

List of Attacks

0) Attack	M_OverheadSwing
1) Attack	M_HorizontalSwing

	X	Y	Z
Pos. Offset	0.04	0.05	0.01
Rot. Offset	90	0	90

HOW TO WRAP IT UP

Put all the pieces together!

Now you should have a Prefab of a Subject, Weapon and Projectile.

Select the Weapon Prefab and put the Projectile Prefab in the "Projectile" slot for the Weapon. Now when the Weapon fires it will be spawning that prefab. Pooling will be in a future release.

Now select the Subject Prefab and press the + button to add a new Weapon Slot. Fill that slot with the Weapon Prefab. Now when that Subject is brought into the scene at runtime it will be using that first Weapon slot. Do not leave empty slots.

Easy peasy!

CONTROLLER SUPPORT VIA INCONTROL (BY GALLANT GAMES)

If you want to use Controller's in Unity and do it properly you will be using a plugin like InControl, cInput or ReWired. Currently there is only very basic Controller support built into Unity by default and a more robust system is necessary if you plan to handle various controller types, cross-platform development and just ease of life improvements in general.

I. Import InControl to the Project. You might need it.
II. Navigate to Deftly / Core / Addons / and import InControl_and_CoOp.unitypackage
III. Replace the existing PlayerController script in the on your character with the new InController script.
IV. Migrate settings between scripts if necessary. (assuming you are upgrading/exchanging scripts here)
V. Check the Use In Control box to let it know you want to use a controller instead of keyboard/mouse.

The script is ready to handle InControl devices. You have about 3 options from here.

1. Use the Co-op demo example as a template to manage Players and Devices.
2. Roll your own Player manager scripts and assign devices manually.
3. Use DeviceID – which is not recommended for numerous reasons (check InControl docs) but available anyway if you're truly determined to use it.

If you have questions about this, check the co-op example demo. If you still have questions, feel free to post them on the forum thread or contact me directly.

I've tested this with a DS4 WIRED USB controller with various drivers and it seemed to work fine.
I've also tested with the XBOX 360 S WIRED USB controller and it performed very well.

OTHER STUFF AND FEATURES

FLOATING TEXT AND DAMAGE

Why not both?

Floating Damage is a three part system.

1. Global Option to enable Floating Text
2. Prefab in a Resources folder which is used for all Floating Text
3. GUIText and Floating Text components on that prefab (this uses legacy GUI)

To use the Floating Text system, go to Window > Deftly Global Options and toggle on the Show Floating Damage checkbox if it is not already on and put your prefab into the slot below. Doing this makes every Subject script spawn a Floating Text when it processes damage. It is required that you have the bulletpoints above completed.

Confirm that...

1. Floating Text is enabled and the prefab is placed into the field.
2. The prefab is correctly configured: Need GUIText and Floating Text components. Refer to example in Demo/Prefabs/Resources for a goby, its pretty straightforward.
3. The prefab is in a Resources folder. This is required, it uses Resources.Load() once to grab the prefab when the game starts so if the prefab is not in a folder called Resources then Unity cannot find it.

Once the prefab is spawned the Floating Text script will position itself and start floating away based on the variables you have put in those fields to configure it.

That is all that is required to make Floating Text work. *Again*, you must enable Show Floating Damage and type in the name of your Prefab that must also have a GUIText component and Floating Text component on it. Expect epic failure otherwise.

If its not working properly, try changing the text size or removing the material. You can always reference the demo scene for a proper example.

POWERUPS

And those sparkly bits

Powerups are easy. =)

1. Make a Prefab with a Trigger Collider
2. Add a Powerup component to it
3. Configure the Powerup to do what you want (this is quite straightforward, but use the tooltips if necessary)
4. Profit

You can implement their collision detection however you want, but I made the demo Powerups by making a Collider (to keep it off of the ground), a Trigger (for the Pickup detection) and a Rigidbody. The layers of these object can be independent from each other – ie the Trigger could be on some layer, then the Collider on another while the Collision Matrix configuration handles your unique scenario (if your design requires it).

You aren't forced to do it that way, but a Powerup requires a Trigger.

More features for Powerup stuff soon.

WEAPON PICKUPS

Not your average powerups

These aren't like Powerups, they're intentionally separate and you have to use the built in Pickup Reference field on each Weapon to tell it what prefab is designated as its Pickup.

The reason is when a Weapon is dropped it will have to actually drop something and it can't be the same object because that Weapon object that the Character is carrying is not designed for that purpose.

So, we'll just make a simpler object to float around as loot and it'll be way easier to manage and design around.

I. Create a prefab with a Trigger, a Non-Trigger Collider and a Rigidbody.
II. Add a Weapon Pickup script
III. Reference the Weapon Prefab which it will give, customize any fields.
IV. Add the model you want to show
V. Adjust any trigger and collider sizes and you're good to go!

You can always check out the demo for example prefabs to see how they are intended to work.

ENEMY SPAWNERS

LEGACY / OBSOLETE. SEE "ENEMY SPAWNER" BELOW FOR NEW SYSTEM

Spawners are handled by a Spawn Trigger, when something on the specified layer enters the trigger it will create the specified Spawners in the specified positions. If you have two positions, you must specify two Spawners because it looks to the 2nd entry when it considers the 2nd position. You'll get errors if you don't do this. Custom inspector in next patch to make this easier.

Spawners can be sort of 'invisible wave spawners' (*Fixed Count*) or 'physical spawner buildings' (*Destroyable*) that you must destroy to stop. If you want an invisible wave spawner, just make an empty GameObject and add a Spawner Script – save as prefab and you're ready to go. If you want an interactive Spawner you will need a Subject script on it (which means Rigidbody and Collider too!) so that it can properly recieve damage and die.

Destroyable Spawner

I. Create and Empty GameObject
II. Set Layer and Tag properly (it is an enemy)
III. Add a Subject Script
IV. Choose Subject Group as "Other"
V. Specify Health in Stats
VI. Add a Spawner Script
VII. Choose Type of 'Destroyable'
VIII. Add at least one Enemy Prefab and specify how many to spawn per wave
IX. Choose Spawn Points (Empty GameObjects as positions)
X. Save as prefab.

This Spawner will now Spawn the indicated enemies and their amounts, randomly at any of the provided positions, in waves, until you kill it. However now you probably need a Spawn Trigger if you're planning on popping this in a level when you enter a room or something.

Spawn Trigger

I. Create an Empty Game Object
II. Add a Box Collider and toggle Is Trigger to on and adjust the size to what you want.
III. Add a Spawn Trigger script
IV. Specify the Positions (other empty GameObjects)
V. Specify which Prefabs of Spawners you want in those positions
VI. If you want the Spawn Trigger to manage door locking when triggered – specify which doors will be affected. Those doors should probably have their "Manually Triggered" bool turned on, although Locked will be toggled by the Spawn Trigger anyway so they can't be opened until the Spawn Trigger says the victory conditions are met.

    Victory conditions are: All Spawners and enemies spawned are dead. If you're using a Fixed Count Spawner it will die by itself when the last wave spawns. Destroyable Spawners will continue spawning waves until you beat some sense into them.

ENEMY SPAWNER

Replaces the previous way of making spawners.

This spawner requires one a few principle things to work.

A "**Host**" GameObject to host controlling the actual spawner and has a Trigger on it for activation.
A " **Destroyable Avatar**" GameObject which can be destroyed, and represents the spawner's physical presence/life.

I. Create an empty GameObject with an EnemySpawner script on it. This is the "**Host**".
II. Add a Box Collider and scale it to a size to represent the 'trigger area' which will activate the spawner.
III. Specify the Destroyable Avatar gameobject. This is a Subject with a Rigidbody which can recieve damage and be destroyed. This is the "**Destroyable Avatar**".
IV. Set it's Crippled Time and Corpse Time to zero so that it disappears immediately when killed. You may want to point a Death FX prefab to it to represent what it looks like or what appears there when it is destroyed.
V. Specify a Spawnable, which is an enemy prefab and a count of how many of them you want to be spawned.
VI. Specify Spawn Positions – they are simply empty GameObjects to mark positions.

Now in the General tab of the Enemy Spawner (Host) ther are dropdowns to help you configure how you want your spawner to work. You've already configured the the necessary information to make it work under any circumstances and now you can adjust how exactly it should behave. For example, Victroy Conditions define if you have to destroy the Avatar for the spawner to stop or if there are no enemies left or if it is the last wave.

Can Trigger This is a layer mask of things that will activate the Enemy Spawner when they touch the Trigger Volume you configured.

DAMAGE TYPES

Some things hurt more than other things…

Damage Types are setup in 3 parts. First, they need to be named in Options (Window>Deftly Global Options>Damage Types Tab). After you assign names to them you then proceed to any of your Subject's and choose their reaction to each Damage Type using the built in sliders found in the Damage Types tab. Third you move on to a Projectile prefab and choose which damage type the Projectile will inflict.

After that the Subject will react by multiplying all incoming damage based on the given slider value of the damage type the Projectile which hit it was inflicting.

MISCELLANEOUS

LIMITATIONS

Stuff you can't do.

- Varied floor/terrain height. You can *travel* vertically by making your Rigidbody a proper weight but you cannot *aim* vertically because it is a very complex task to support this properly.
- Use empty Weapon slots
- …

QUESTIONS?

Personally written letters by mail are preferred

Please contact me by any of the methods outlined in the contact section on the first page. Deftly is in early development stages and will benefit greatly from your input! Help shape it by communicating how it can be improved and by reporting bugs when you find them. =)