Piscine Unity - Rush01

Hack and Slash

Staff staff@staff.42.fr

Summary: This document contains the subject for Rush 01 for the „Piscine Unity" from 42

Contents

Chapter I

General Instructions

- The Unity bootcamp has to be made entirely, exclusively and mandatorily in `C#`. No `Javascript/Unityscript`, `Boo` or any other horrors.

- The use of functions or namespace not explicitely authorised in the exercise header or ini the rules of the day will be considered cheating.

- For a optimal usage of Unity, you have to work on `~/goinfre`, which is on the local drive of your computer. Remember to make appropriate backup on your own, the local goinfre can be purged.

- Unlike any other bootcamps, each day doesn't require a folder `ex00/`, `ex01/`, . . . , `exn/`. Instead you'll have to submit your project folder which will be name like the day: `d00/`, `d01/`, ... However, a project folder, by default, contains a useless folder: the `"projet/Temp/"` sub folder. Make sure to **NEVER** try to push this folder on your repository.

- In case you're wondering about it, there is no imposed norme at `42` for `C#` during this bootcamp. You can use whatever style you like without restrictio. But remember that code that can't be read or understood during peer-evaluation is code that can't be graded.

- You must sort your project's assets in appropriate folders. For every folder correspond one and only one type of asset. For exemple: `"Scripts/"`, `"Scenes/"`, `"Sprites/"`, `"Prefabs/"`, `"Sounds/"`, `"Models/"`, . . .

- Make sure to test carefully prototypes provided every day. They'll help you a lot in the understanding of the subject as well as what's requested of you.

- The use of the Unity Asset Store is forbidden. You are encouraged to use the daily provided assets (when necessary) or to look for additional ones on the Internet if you don't like them, exception made of scripts obviously because you have to create everything you submit (excluding scripts provided by the staff). The Asset Store is forbidden because everything you'll do is available there in one form or another.

However the use of Unity Standard Assets is authorised and event advised for some exercises.

- From d03 for peer-evaluation you'll be required to build the games to test them. `The corrector` will have to build the game, you must therefore always push projects/sources. You project must always be properly configured for the build. `No` last minute tweaks will be tolerated.

- Warning: You'll not be corrected by a program, except if stipulated in the subject. This imply a certain degree of liberty in the way you can do exercises. However keep in mind the instructions of each exercise, don't be LAZY, you would miss a lot of very interesting things.

- It isn't a problem to had additional or useless files in your repository. You can choose to separate your code in different files instead of one, except if the exercise's header stipulate a list of files to submit. One file must define one and only one behaviour, so no namespace. Those instructions don't apply to the `"projet/Temp/"` sub-folder which isn't allowed to exist in your repositories.

- Read carefully the whole subject before beginning, really, do it.

- This document could potentially change up to 4 hour before submission.

- Even if the subject of an exercise is short, it's better to take a little bit of time to understand what's requested to do what's best.

- Sometimes you'll be asked to give specific attention on the artistic side of your project. In this case, it'll be mentioned explicitely in the subject. Don't hesitate to try a lot of different things to get a good idea of the possibilities offered by Unity.

- By Odin, by Thor ! Use your brain !!!

Chapter II

Specific instructions of the day

The time has come for the last rush of your Unity Bootcamp, it's now or never to show off everything you learned and everything you can do. This time you are allowed ABSOLUTELY EVERYTHING you want. Meaning you are now allowed to use the: Asset store entirely (Yes even the scripts), all the 3d models / sounds / musics / etc. you can find on the Internet, as well as all the assets we provided since the beginning of this bootcamp.

The aim is just to do the most cool and stylish game in two days, and by the way that's the usual average time for a GameJam.

Chapter III

Foreword

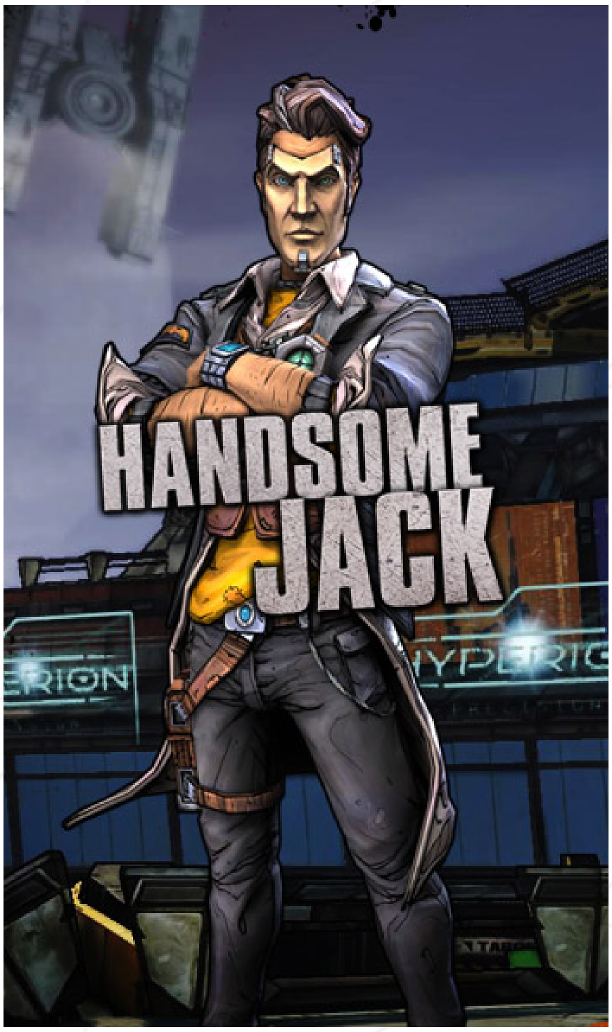

Figure III.1: "I'm scratching my head to find a name for this diamond pony I bought. I though about “Fart face”, in your honor. But, it sounds a little immature. What do you think about “Ass stallion”? No, Worse. I will think about it a little bit more"

Chapter IV

About teamwork - part II

Here is a recap of good practises when working as a team using Unity. To avoid losing precious hours/days of work and a lot of nerve crisis here are some advise to work properly as a group.

- Since we are a nice bunch you will find in the project's assets a .gitignore file which will allow you not to push file that should remain local.

- You should `NEVER` work at the same time on the same scene or the same prefab. Unity still has a lot of issues with the merge of these assets.

- Be cautious of the code of your partner when you update an object containing his script. A missing tag is enough to break down a feature.

- Communicate as much as much with your partner on modification you make on the project.

- Divide the work preperly and avoid working on the same feature or the same gameplay element at the same time to avoid conflicts.

- Commit as often as possible. When facing a conflict you will be able to roll back to a past functional version of the game.

- Put some "milestones" together, meaning a functional version at key stage of the developpement of the game and synchronise your repos before starting a new step. If you do this you should always have a stable version available if one of you breaks something and in worse case scenario you will be able to push your last valide "milestones" and avoid having a broken project.

Chapter V

Mandatory part

I imagine your frustation at the end of the d08 because we had a cool representation of a Hack and Slash but no gameplaye, which is quite sad we have to admit. To rectify this, this final rush will be to finalize your Hack and Slash by making it completely awesome. For example by creating real maps, a loot system, skills, a little more than 2 enemies etc. Let's take a look at this in more details.

V.1 The basics

Let's start from scratch, here is a list of what is required in your basic hack and slash:

- A map with a NavMesh.
- A diving camera, fixed on the hero controled by the player (you can keep on using Maya or take any other character, as long as he is rigged/animable).
- A hero controled with the mouse: left click on the landscape to move, left click on an enemy to attack, the hero keep on attacking if the click is maintained on the enemy.
- Enemies, that follow and attack the hero if he is in their detection range.
- Basic idle/run/attack/death animation for every character implemented in the game.
- An advanced fight system based on stats: strength that increase damages, agility that increases hit chance and parry, constitution that will increase HP.
- An XP system, every ennemy killed give X experience points. When the hero has enough XP he levels up and obtains 5 stats point to spread accross main stats.
- Enemies that spawn must be of the same level than the hero, it being level 1 or level 50.

- A HUD displays the life of the hero/enemy selected, their respective levels as well as the XP of the hero.
- A character window that displays all the stats: STR, AGI, CON, minDamage, maxDamage, XP, xpToNextLvl, maxHP . . .
- Life potions randomly looted by enemies and that heal the hero with 30% of maxHP when he walks on them.

V.2 Skills

To make this Hack and Slash a little more interesting that a simple click on an enemy we will have to add some skills as well as a talent tree!

- After each Level Up, the hero will also win a talent point. We can see the total unused amount in a talent tree window.
- By pressing "N" the talent tree opens. It's possible to use talents points to unlock skills. The talent tree is made of several tier that unlocks every 6 levels. Meaning, from level 1 to 5, only the skill of the first tier can be leaned, from level 6 to 11, the skills of the first and second tier, . . .
- Each skill can be upgraded several times (let's say 5). For each additional point used on a skill, it improves and becomes more efficient.
- You must create at least 6 skills including at least: an area attack to put on the ground by clicking (using a template to aim) on te mouse, an area attack around the hero and following his movements, a direct damage spell (like a fireball), a passive skill to improve an aspect of the hero (except the main stats), a healing spell.
- Each skill must have a tooltip explainig how it work, its damages . . . , as well as its stats on the next level (to allow the player to know if he want to improve it or not).
- You can create a specific resource for your skill, like mana or rage, or use resourceless skills but with a CD (the Cool Down is the simplest to implement but you are free to choose the one that suits you best).
- Skills must have a particule effect or anything else to make the player have a visual feedback about it area of effect.
- You must also implement a skill bar, with a limited number of slots (between 4 and more), so that the player has to do a choice and is not able to use all of them at all time. You must therefore create a way to "equip" the skills using this bar, and allow the player to use them by pressing the keyboard keys 1, 2, . . .

V.3 Loots

A Hack and Slash without loots, it's a little like pong without rackets: it's crap. Let's now implement loots and inventory to store them.

You must create a system of equipements. By equipment we are referring to weapons to start with. When an enemy dies, there is a chance of him dropping a weapon. These weapons must be randomly generated; here are couple of instruction regarding their generation:

- Weapons must have different appearance. The type of weapon must also be randomly selected in a liste of possible models.
- Every weapon musthave a minDamage and an attack speed but it's up to you to add some additional stats if you want.
- The weapon's stats will generally be random but not completely. You will have to take the current level of the player into consideration to affect the randomisation of the values of these stats. The weapons cannot have disproportionate stats against the enemies of the player's level.
- Weapons must have a tooltip that is displayed went the mouse goes over them, when it's on the ground or in the inventory. This tooltip must display the weapon's stats.
- Create rarity levels as well. Some weapons must be more difficult to loot than some others. For example: Common, Uncommon, Rare, Legendary.

That's not it. It's nice to have a weapon generation, it's better to be able to equip them. The inventory will prove useful to carry them around. You must implement a system that allows the player to gather weapons by left clicking on them. These weapons appear then in the inventory window accessible by pressing the "I" key. Once in the inventory, it's possible to organise the weapons as we want and change the weapon currently equipped. it must be possible to throw away a weapon by drag and dropping it on the ground. Obviously the inventory will be limited (12 slots seems to be a good average).

V.4 A little bit of variety

Now that your game system is a little more complete, it's type to be creative! You have to create an outside map, interesting to travel through, as well as a dungeon on several floors (let's say 3). On the last one, you will have to implement a Boss that will have a ton of HP, different attack skills (you can reuse some of the skills from the hero to win some time) that promises an epic battle!

You are free to do whatever you want here, enjoy!

V.5 Soundtrack

To make the game even more immersive, you will have to create a soundtrack. Here again, you are free to do whatever you want, but choose wisely sound effets and musics to give a lot of atmosphere to your game! It's alwo the time to try out the reverb zones that we talked about earlier this week.

V.6 A cheat code to rule them all

The point of a Hack and Slash is to progress on different levels. However to save some time and to avoid having the correctors play 2 hours to test your game, skills and all of this, implement a cheat key, that will instantly level up your character, and another one to spawn a weapon of the hero's level. Your game must be balanced at level 1, 20 or 50. The p2p correction will request the corrector to test the game at several hero levels.

Chapter VI

Bonus part

If everything we asked is done and the rush isn't finished, you are `REALLY AWESOME !`

But if you are still hungry, here are some possible bonus ideas:

- Additional lootable object such as armors, helmets, boots, rings, amulets ...
- Different character's classes.
- Ranged weapons.
- Elementary effects on weapons, for example a flamme sword with particule effects and crazy stats!
- A currency management: enemies loot credits that allow the hero to buy items from a NPC/Shop ...
- A title screen and a backup system that would keep the progression of the player saved to be able to start the game from where he left with his level and equipmenent etc..
- Implement an "Ass Stallion".