

Basic Unity **Tactical** Tool

Save a lot of time to **make your own tactical** game!

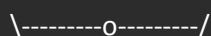
Create a tile board with holes and obstacles, easily **customize your tile board**, **add characters** on, create **teams**, **NPCs**, **add behaviors** and **let's fight!**

Enjoy your **Basic Unity Tactical Tool!**

***NB:** I advise you to read the "Quick start" category first. Everything is done to be the most simple to understand and use.*

SUMMARY

- FEATURE LIST	1
Movement along the grid	1
Ranged attacks	1
Characters	2
Camera	2
Board creation	2
Others	2
- QUICK START	3
Tile board	5
Advanced : create a new type of tile	5
Tutorial : Create the tile "Little obstacle"	7
Characters	8
Move	9
Look	10
Attack	10
Health	10
Infos (Name and team)	11
Advanced : Create a new team	12
Behavior	12
Advanced : Other children	13
Game rules	13
Movement	13
Vision	14
Turns	15
Advanced - Other customizable points	16
Inputs	16
Feedbacks	16



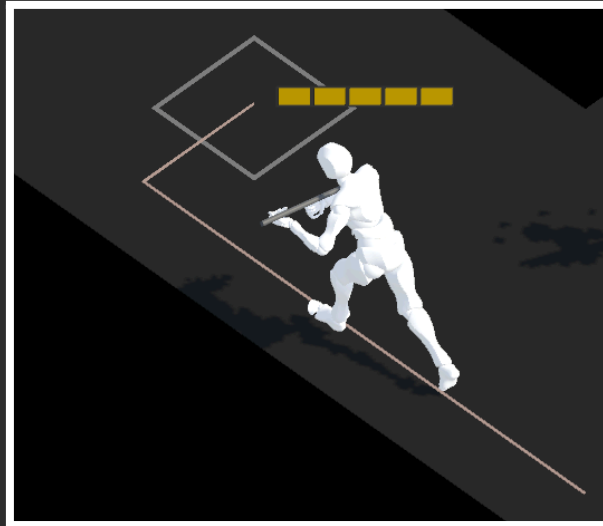
/-----o-----\

Camera	17
- TECHNICAL DOCUMENTATION	18
How works a game turn	18
Managers	20
Character components	20
Other scripts	21
- CREDITS	21
- LICENSE	21

\-----o-----/

- FEATURE LIST

Movement along the grid



Pathfinding on 2D grid (A* algorithm) using diagonals or not, through other characters or not, with holes and obstacles.

Movement feedback with character animation, movement line, allowed/forbidden tiles.

Ranged attacks

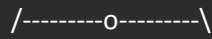


Line of sight with or without obstacles.

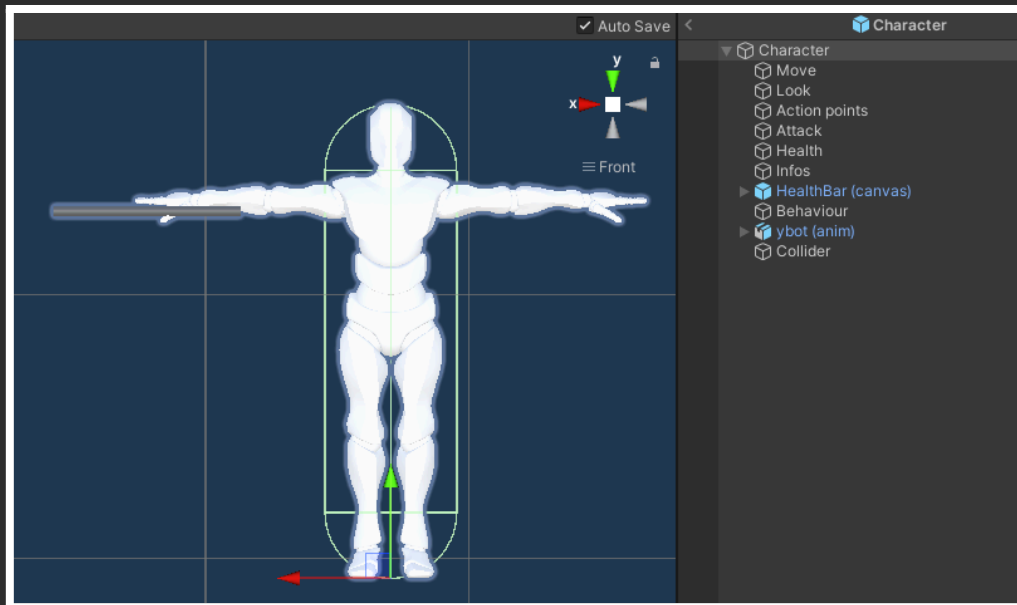
Shooting range.

Damage range to set damages between two values.

Percent of chance to touch the enemy.



Characters

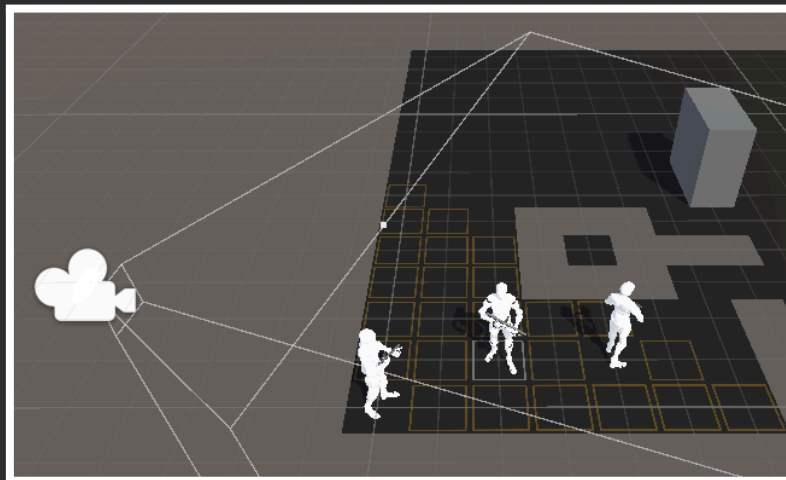


Health points with hurt and death animations.

Teams and team color system.

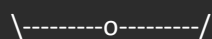
AI Behavior with basic behaviors (follow and attack).

Camera



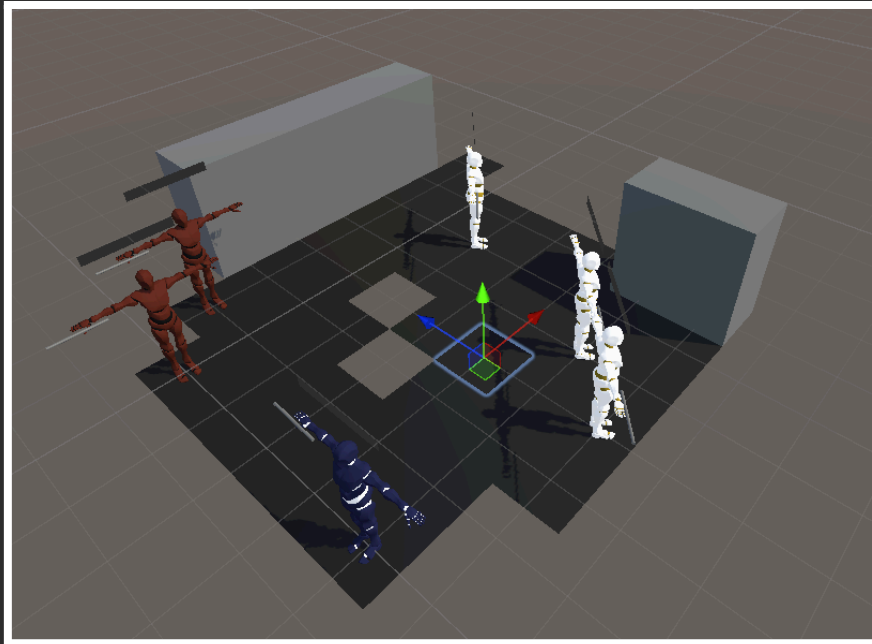
Mouse movement on the borders of the screen.

Center position on current character.



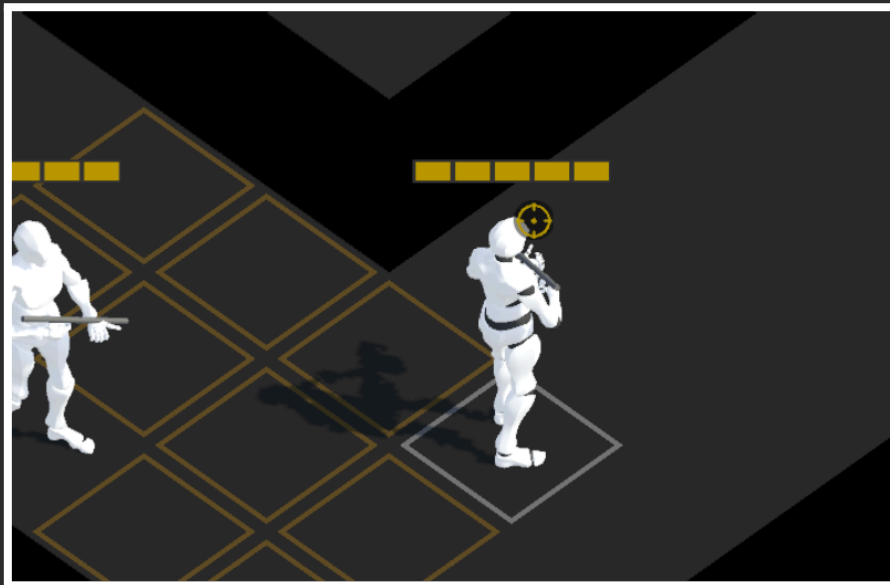
/-----o-----\

Board creation



Create your scenes easily in the Unity editor with only drag and drop.

Others



Custom mouse cursors, easy to modify and adapt to your game.

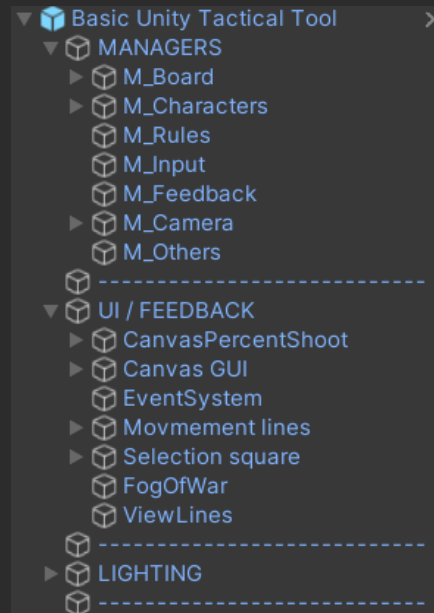
Fog of war showing your range view.

Victory screen for a fancy reward!

\-----o-----/

- QUICK START

The project is based on **big managers**, each one managing a part of the game (board, characters, etc.). They are named `M_Name`.



If you want to **create a new scene**, that's quite simple:

1. Drag and drop the **Basic Unity Tactical Tool** prefab in your scene.
2. Drag and drop **tiles** and create your board.
3. Drag and drop **characters** and parameter them.
4. Click on the Play **button** and start your game.

How to customize your tile board, characters and many other things is explained below.

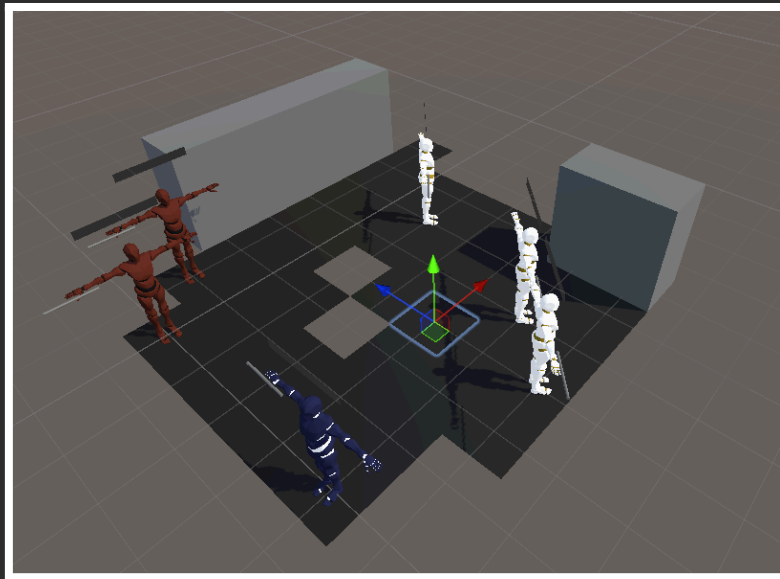
This part of the documentation doesn't need any programming skills.

*If you are looking for the technical details, the **Technical documentation** part is made for this.*

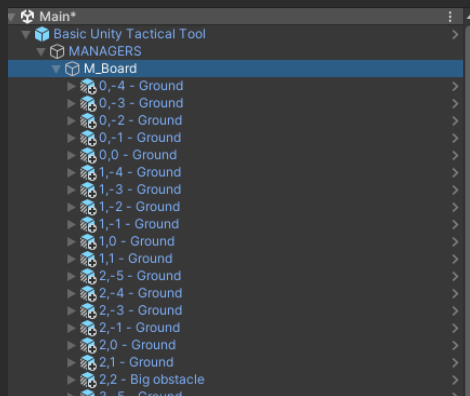
The **Basic Unity Tactical Tool prefab** is in Prefabs/Main.

/-----o-----\

Tile board



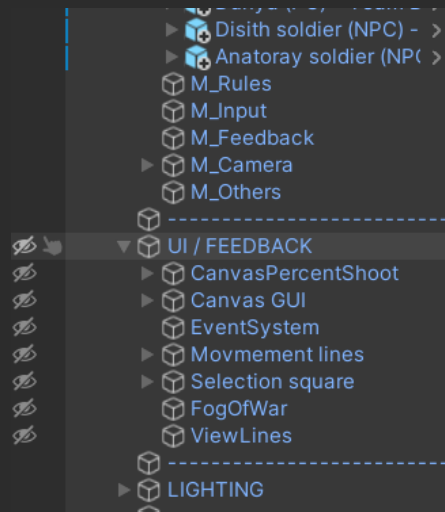
Drag and drop the tile prefab of your choice (Prefabs/Tiles) where you want in the scene. It automatically **snaps on the grid** and it's stored in **MANAGERS/M_Board**.



NB: When you use the tile board or character, I advise you to mask the UI in the editor, like in the image below.

\-----o-----/

/-----o-----\



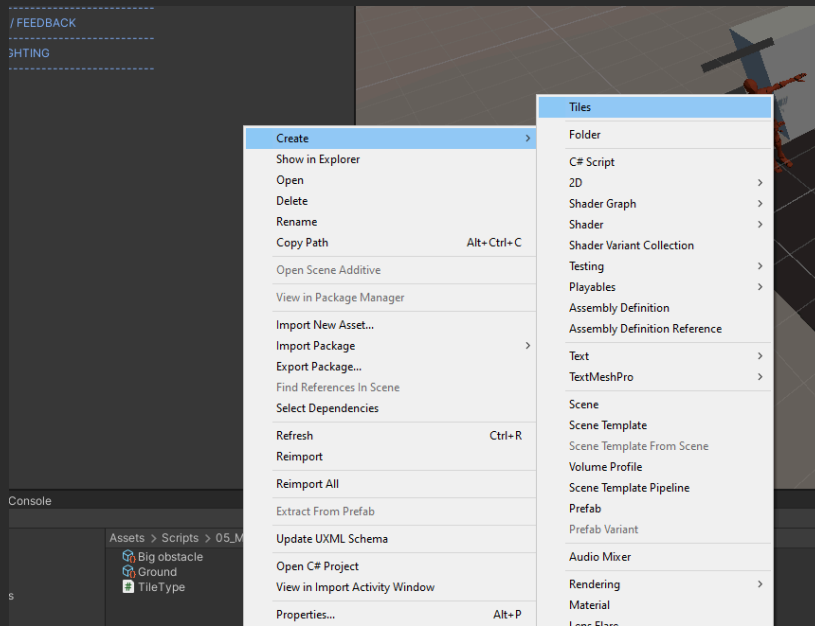
Advanced : create a new type of tile

Currently there are two types of tiles : ground where characters can walk, and big obstacles which block movement and line of sight.

You can **create another type of tile** if you want (for example a Smoke tile who blocks line of sight but is walkable):

- Just **copy one of the tile prefabs** in Prefabs/Game_Tiles, and change the Type variable by a new type.
- To create a new type, **right click in the Project window** and select **Create/Tile**. You can store it in Data/Type of tile if you want.
- Simply **name the new one** as you like (by exemple Smoke).
- **Don't forget to assign** the new type to the new Tile prefab.
- *NB: You can parameter walkable tiles and line of sight blocker in the characters, cf. Characters part.*
- *NB: Tiles prefabs are in Prefabs/Game_Tiles, types of tiles are in Data/Type of tiles. But you can move it everywhere in the assets if you want.*

\-----o-----/



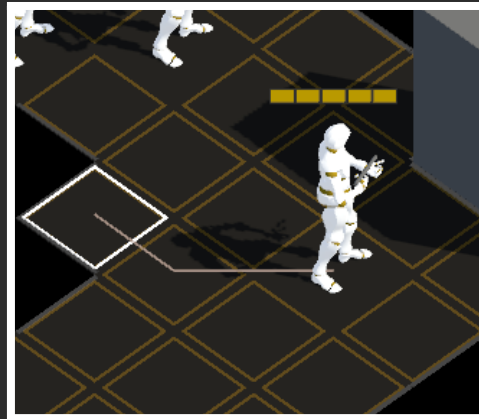
Tutorial : Create the tile “Little obstacle”

To learn how to **create a new tile**, you can create the tile “**Little obstacle**”.

The little obstacle is something like a crate, blocking the movement but not the line of sight.

1. In Data/Type of tiles, duplicate one of the existing type and rename it “Little obstacle” (or create a new one with right click/Create/Basic Unity Tactical Tool/Tile).
2. In Prefab/Game_Tile, duplicate one of the existing tiles and rename it “Little obstacle”.
3. In this object, in the Inspector window, change the “Type” property to Little obstacle.
4. Now you have a new type of tile, but it has no appearance. Double click on the prefab to open the Prefab view, and add a 3D cube on it (for the Transform metrics, you can use Position = 0, 0.5, 0 / Rotation = 0, 0, 0 / Scale = 10, 1, 10).
5. You can drag and drop in the scene: you can see this blocks the movement and doesn't block the line of sight (there are default parameters).
 - a. If you want it to be walkable: go in Prefab/_Main, open Character_Original, and select the Move child. Add the type (in Data/Type of tiles) Little obstacle in Walkable tiles.
 - b. If you want it to block the line of sight, in Character_Original, select the Look child. Add the type Little obstacle in Visual Obstacles.

Characters



There are some **example characters** in Prefabs/Game_Characters. They are named "Hero" (playable character) and "Bot" (NPC).

Same as tiles, you only have to **drag and drop a character** prefab in the scene **where you want him to start**.

It's automatically stored in MANAGERS/M_Characters and works immediately.

NB: You can duplicate the prefab Character (in Prefabs) to create multiple characters prefab if you want.

NB: you can bake characters and tiles on the board at the same time with the Bake button.

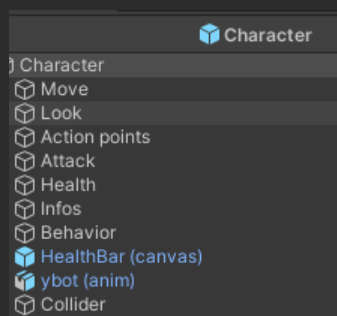
You can Play now!

*Simply click on the **Play button** of the Unity Editor.*

If you want to **customize** your experience, characters have **numerous parameters**.

Let's simply check its children.

I advise you to duplicate one of the existing character prefab to create a new one and follow the next examples.



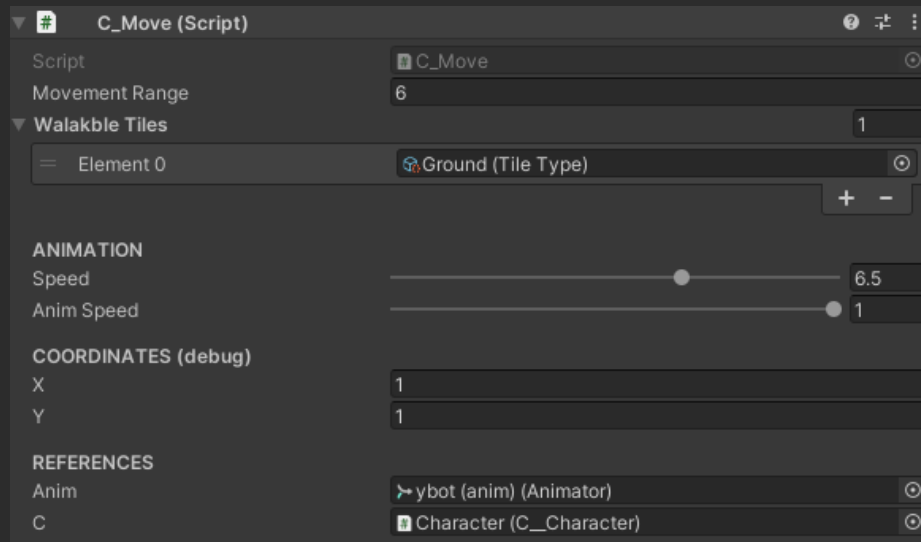
If you just want to parameter general options, you can skip to Game rules.

Move

Move manages the **character's movement**.

The two main parameters useful for you are:

- **Movement Range**, which determines how many tiles the character can move by turn.
- **Walkable tiles**, which determines the type of tiles where the character can walk (*cf. Tile board / Advanced to create a new type of tile*).



- **Speed** is the character's visual speed.
- **Anim Speed** is the speed of the animation.

Other parameters are for code and debug.

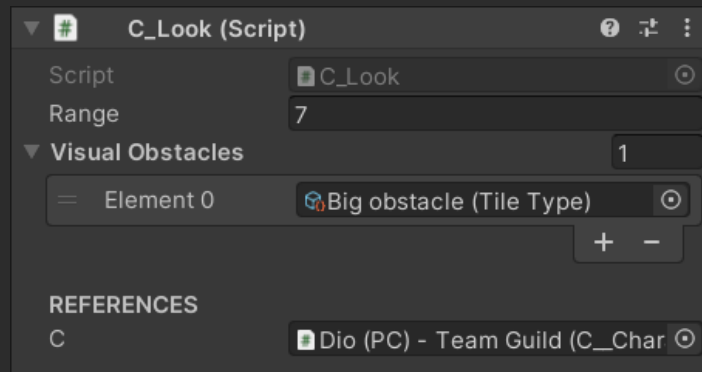
Look

Look manages the **character's view**.

The two main parameters useful for you are:

- **Range** determines the maximum distance (in tiles) of the character's line of sight.
- **Visual obstacles** are the type of tiles which are blocking the line of sight.

/-----o-----\



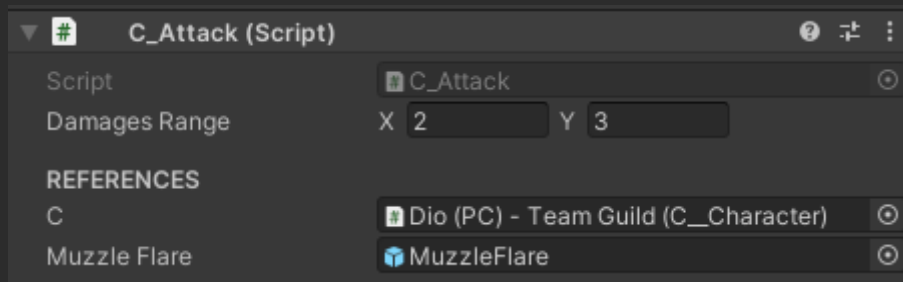
REFERENCES zone is for the code.

Attack

Attack manages the **character's attacks**.

The main parameters useful for you is:

- **Damage Range** is the range of damages (X = minimum, Y = maximum) dealt by the character.
If you want a constant damage value, use the same X and Y.



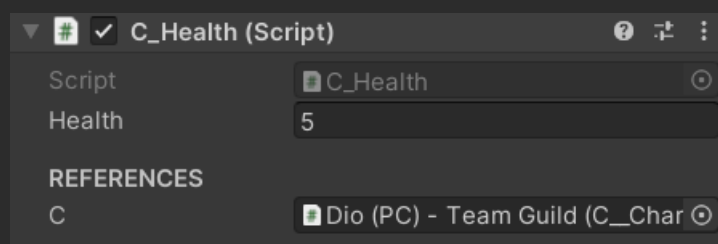
REFERENCES zone is for the code.

Health

Health manages **character's health points**, damages absorbed, healed and death.

The main parameters useful for you is:

- **Health** who gives the total health points of a character.



\-----o-----/

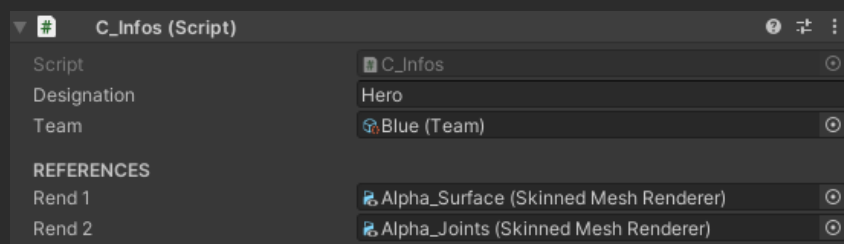
Infos (Name and team)

Info manages the **character's main information**, like its name or its team.

The two main parameters useful for you are:

- **Designation** is the name of the character. *When you rename it, you can push the Bake button in MANAGERS/Board to update the description of the gameObject.*
- **Team** is the character's team, determining who are its allies or enemies. You can find the different team presets in Data/Teams. *To create and modify a team, I advise you to read "Advanced : Create a new team" below.*

NB : If you modify the team, team's name, team's color or character's name, the character is automatically updated.



Other parameters are for the code.

Advanced : Create a new team

Go to Data/Teams: here you can **create and manage a new team**.

NB : They are stored here by default, by you can stored them everywhere you wan.

To create a new team, **right click** and Create/Basic Unity Tactical Tool and just **name** the new object (it's the team name).

You can assign **two materials** for your character's team color.

Now you can assign the team (*in character's prefab / Infos*).

NB: If you use your own character's model, you can modify the destination of the renderer's materials in the Infos component of the character (if there's no renderer, it still works).

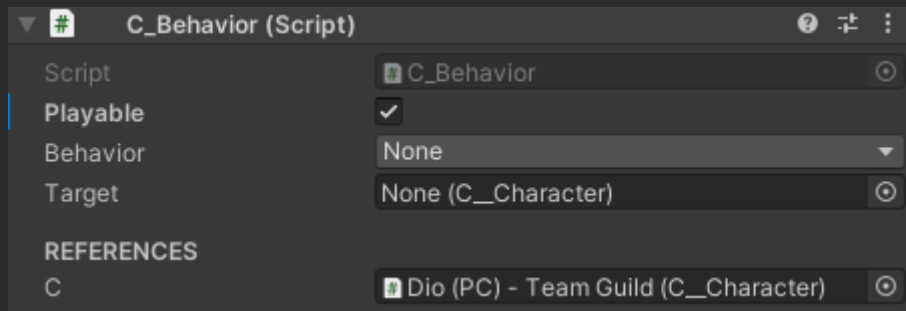
Behavior

The three main parameters useful for you are:

- **Playable** says if:
 - the character is controllable by the player (PC) if it's checked
 - The character is a non playable character (NPC) using a behavior if it's unchecked

/-----o-----\

- **Behavior** is the type of behavior of an NPC (if Playable is unchecked).
 - None: the character pass its turn.
 - Follower: the character follows the Target (parameter below). If there is no target, it passes its turn.
 - Offensive: the character attacks the closest enemy on sight. If there is no enemy, it passes its turn.
- **Target** is the character used by some behaviors.



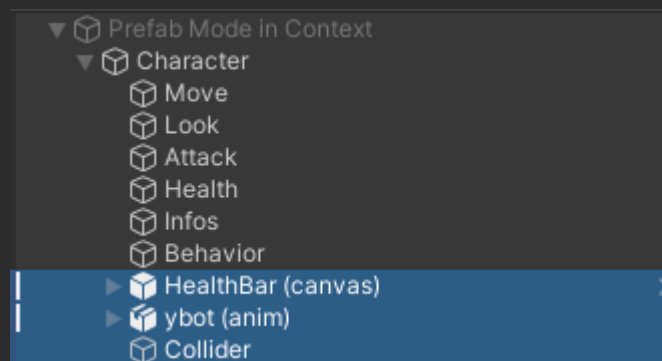
REFERENCES zone is for the code.

Advanced : Other children

There are other children on the character prefab.

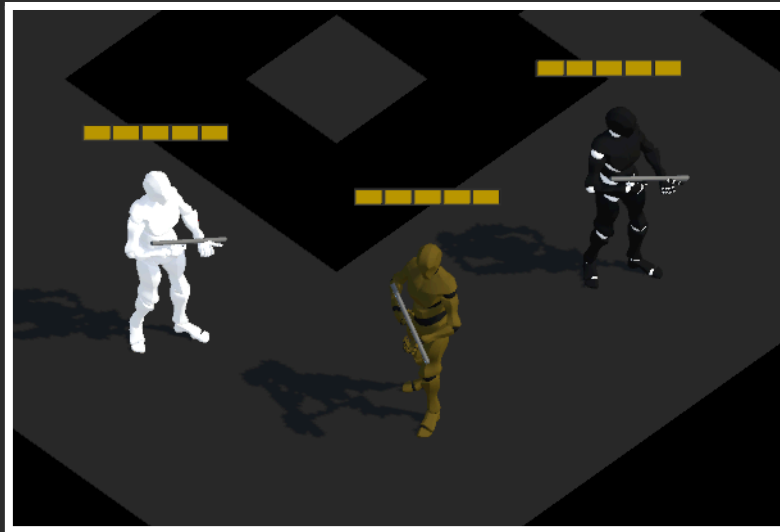
You don't need to parameter them, but this is an explanation of their utility:

- **HealthBar** is the world canvas containing the Health Bar UI.
- **ybot (anim)** is the parent of the 3D model and hold the animator, and the component C_Anim which manages the anim. *If you want to replace the 3D model, you'll need to work with this part.*
- **Collider** contains the collider for the mouse raycast.



\-----o-----/

Game rules



You can **customize** the game with the different **game rules**.

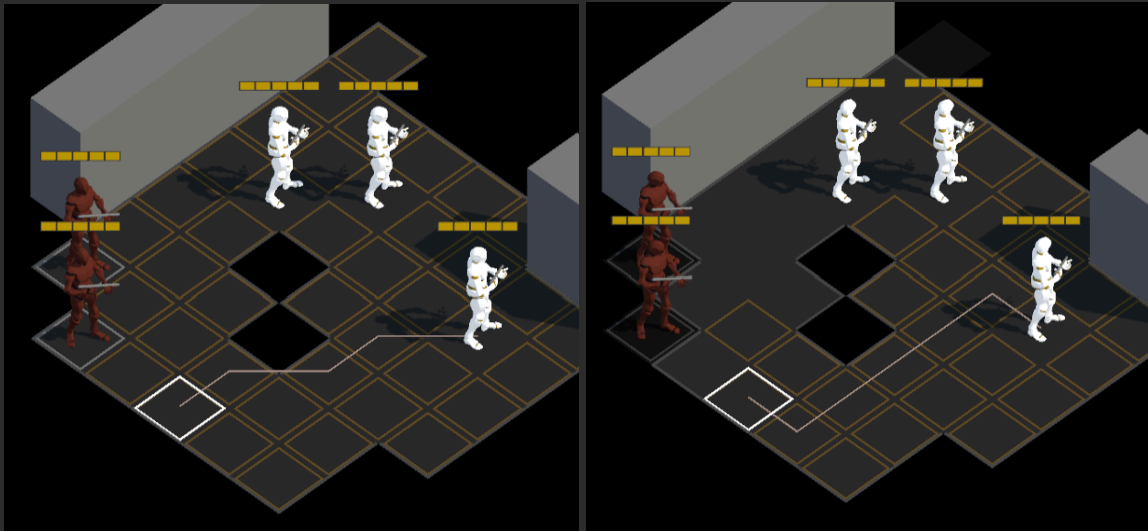
Movement

MOVEMENT	
Use Diagonals	<input checked="" type="checkbox"/>
Can Pass Through	Nobody

Use Diagonals changes the pattern of movement with or without the diagonals.

- If it's checked, the character can use diagonals to move.
- If it's unchecked, the character can't use diagonals to move.

/-----o-----\



Can Pass Through determines which kind of character will block the character's path.

- **Everybody:** the character's path isn't blocked by anybody.
- **Nobody:** the character's path is blocked by every other character.
- **Allies only:** the character's path is only blocked by enemies (and not blocked by the team's characters).

Vision

VISION	
Percent Reduction By Distance	<input type="text" value="5"/>
Can See And Shot Through	<input type="text" value="Everybody"/>


Percent Reduction By Distance is the reduction of chances to touch an enemy by tile of the range (for example, if the *Percent Reduction By Distance* = 5, and the enemy is at 3 tiles of distance, the character has $100\% - (5 \times 3) = 85\%$ of chances to touch the enemy).

Can See And Shot Through determines which kind of character blocks the line of sight or the shoot.

- **Everybody:** the character can see and shoot through allies and enemies.
- **Nobody:** every character blocks the line of sight and shoot.
- **Allies Only:** only the enemies block the line of sight and shoot.

\-----o-----/

Turns

TURNS	
Chosen Character	None (C_Character) 
First Character	First Character Of The First Team ▼
Bots Plays	After Playable Characters ▼

Chosen character is only an option for First character (explanations below).

First character is the character who begins the game.

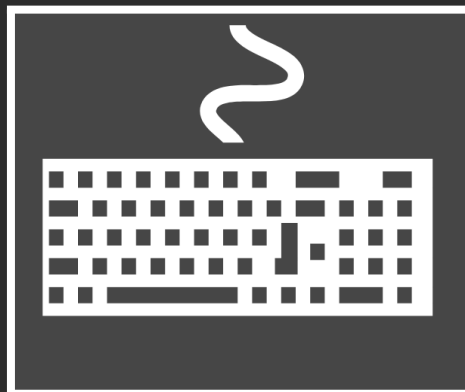
- Random: A random character begins the game.
- Chosen Character: the character of the parameter Chosen Character begins the game.
- First Character Of The First Team: the first character of the first team (in the team list) begins the game.

Bots Play determines if the NPCs play their turn before or after the playable characters.

- Before Playable Characters: at the team turn, NPCs play their turn before the playable characters.
- After Playable Characters: at the team turn, NPCs play their turn after the playable characters.

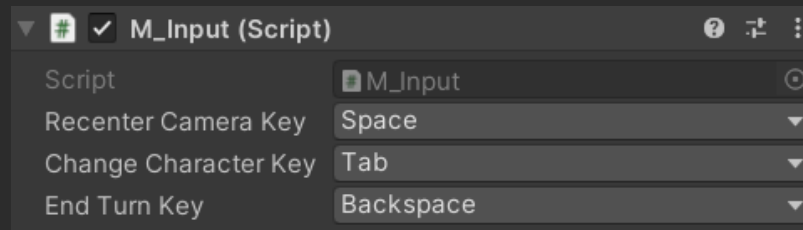
Advanced - Other customizable points

Inputs



You can change the **keyboard player's shortcuts** in **MANAGERS/M_Input** gameobject in the hierarchy.

/-----o-----\



Recenter Camera Key recenters the camera on the current character.

Change Character Key switch to another playable character in the team.

End Your Turn Key ends the turn of all your playable characters.

NB: Keycodes include mouse clicks.

Feedbacks



You can **customize the mouse cursor** when you hover any objects.

You can set it in **MANAGERS/Feedbacks**.

Aim Cursor is enabled when you hover an enemy you can attack.

No Line Of Sight Cursor is enabled when you hover an enemy out of your line of sight.

Out Of Actions Cursor is called when you have an enemy but you have already played.

Camera



You can **customize camera parameters** in **MANAGERS/M_Camera**.

\-----o-----/

/-----o-----\

X, Y and Z Offset values set the offset between the camera and the current character.

Moving Time (the time the camera needs to reach its destination) makes the camera quicker/slower. The higher this value is, the slower the camera is.

Moving Time Curve smoothes the movement (preset is a logarithmic curve: a quick movement at the beginning and slow in the end).

Border Multiplier multiplies the camera's speed when the mouse is on a screen border.

\-----o-----/

- TECHNICAL DOCUMENTATION

How works a game turn

Game initialization

- M_Characters :
 - **finds all the characters on the scene** and puts them in the character's list.
 - **starts the turn of the first character** chosen by M_Rules
 - passes to the next step.

Beginning of a turn

This is the **initialization of all the elements of the turn**. It's managed by **M_Turn**, who manages the characters' turn.

NB : The method who initializes the situation is NewCurrentCharacter().

- It **clears all the feedback**.
- It gives the **camera** a new target and recenters it.
- It **fills the action points** of the character.
- The next step depends if the character is a playable character (PC) or a non playable character (NPC).
 - If the **character is a PC**, it enables the player's input, player's UI and enables the tiles feedback of the character (go to the step Playable character's turn).
 - If the **character is a NPC**, it disables the player's input, disables the player's UI and plays the NPC's behavior (go to the step Non playable character's turn).
 - *NB : To know if a character is a PC or a NPC, uncheck or check the Playable parameter on C_Behavior, on the character (in the scene).*

Playable character's turn : Player's action choice

A player plays his character's turn.

M_Input checks for the player's input.

- He can **put his pointer over tiles** :
 - If the tile is **not allowed** (hole, an obstacle or if the pointer is out of the board), it clears feedback (line, cursor, pathfinding values, etc.).
 - If the **tile is occupied** by an enemy, it sets the cursor depending on the line of sight on this enemy and the action points.
 - If the **tile is free** and accessible (no matter the action points), it sets moving feedback.
 - *NB : M_Feedback manages feedback and the cursor and M_Pathfinding returns the path and the line of sight.*
- He can **click on a tile** :
 - If this **tile is occupied** by an enemy, it allows the character to attack the target (the character will block the attack if the enemy is out of sight or the character hasn't enough action points). The attack is managed by the C_Attack script of the character.

/-----o-----\

- If the **tile is free** and the character can move on (matter its action points), it allows the character to move. The movement is managed by the C_Move script of the character.
- He can press the shortcut to **pass to his next playable character**, managed by M_Characters (back to the Beginning of a turn).
- He can press the shortcut to **pass to the next player's turn**, managed by M_Characters (back to the Beginning of a turn).
- He can press the shortcut to **recenter the camera** on the current character, managed by M_Camera.
- He can put his pointer over the borders of the screen, so it will **start the camera movement** (managed by M_Camera).

Non playable character's turn : Behavior execution

If the character is a NPC :

- M_Characters disables the UI and starts the NPC's behavior (on NPC's C_Behavior script).
- C_Behaviour checks the current behavior :
 - None : character waits 1 second and passes to the next turn.
 - Follower : if the character has a target parameter in C_Behavior component, it goes the closer it can to the target. Else, it passes to the next turn.
 - AttackerOnce : character finds the closest enemy on sight and range and attacks it once.
 - Offensive : like AttackerOnce, but the character attacks until it has no Action points.

Team victory

At the end of a character's turn or action, **C_Characters checks if the team wins**.

In this case, it shows the victory screen.

Managers

All the big parts of the game are managed by a **manager class**, named *M_PartOfTheGameToManage*. They are all **singleton**.

They all begin by **M_** to signify they are managers.

M_Board manages and bakes the tile board, containing the tiles list.

M_Rules contains the different rules of the game, to change them easily and make quick tests.

M_Input checks the player's controls.

M_Characters manages the characters list.

M_UI manages all actions made by UI elements (and UI elements themselves).

M_Feedback manages feedback and pointers.

M_Camera manages the camera of the game.

M_Pathfinding manages pathfinding and line of sight methods.

M_Turns manages turns of the character (next turn, next character to play, victory, etc.).

\-----o-----/

/-----o-----\

NB : **M__Managers** is used to reference all the managers and simplify their call in the code.

Character components

A character is an **autonomous entity**, but the player can take control of it.

The character contains **different components** working together (health, movement, attack, behavior, etc.).

C__Character contains references of all these components, and all these components contain a reference to C__Characters to make easy code references.

They all begin by **C_** to signify they are character components.

C_AnimatorScripts manages the animations of the character (with events on the animations).

C_Attack manages the attack.

C_Behavior manages the behavior of non playable characters.

C_Health manages health and death.

C_Infos contains the name, the team and gives the team's colors to the character.

C_Look manages line of sight, range and target acquisition.

C_Move contains coordinates and manages movement.

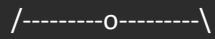
Other scripts

Feedback scripts begin by a **F_** to signify they are feedbacks.

UI scripts begin by a **UI_** to signify they are UI.

***NB** : Some scripts have no letter on the beginning, they are misc scripts.*

\-----o-----/



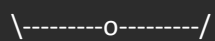
- CREDITS

Icons from game-icons.net.

Character and animations from mixamo.com.

Thanks to [@henriforshort](#) and [@JFaedus](#) for their help.

Made by [@dracotnu](#) for fun.



/-----o-----\

- LICENSE

This is an **open source** project.

You are **free to use it** for every usage (including commercial) **without citation**.

But you are free to let me know if you use this project, I'll be happy to know it.

\-----o-----/