EasySnap

V1.5

First of all:

Thank you for buying my asset! :-)

If you have any problems, feel free to contact me at

mail@ondeth.de

or visit

https://bugs.ondeth.de/?lang=en_US.

Thanks again!

Soo, what's EasySnap?

Well… EasySnap is a small asset, which creates an unlimited number of invisible in-game 3D grid and allows you to snap objects at variable intervals to it –
everything without typing a single line of code.
On top of that: EasySnap works in 2D games as well.

A short feature overview

- No coding knowledge required (GUI-only)
- Supports both edit and play mode (selectable)
- Invisible 2D/3D grid using Unity's coordinate system
- Customizable visual grid in edit mode
- Unlimited number of grids simultaneously
- Snapping can be enabled per-axis and per-object
- Snapping intervals can also be set for each axis individually
- Snap checks intensity can be edited for a better efficiency and lower performance requirements
- Snap accuracy can be edited to allow variations and a more natural look
- Well commented code to easily understand the procedure

Latest manual update: 18/10/19

Getting ultra-quick-started (for lazy guys like me)

- Drag and drop the prefab EasySnap Manager from the EasySnap asset folder into the Hierarchy Window

- Drag and drop the *SnapItem* script (found in the *Scripts* folder) onto every object which should snap to the grid

- Click on the previously added EasySnap Manager in the Hierarchy window

- Startup tab of the EasySnap Manager: Check the enabled Unity mode(s) and every **axis** you want to have snaps on

- Basics tab: Set your **distance** between two snapping points (the rasterization interval) and the two outer **edge points** of your desired grid (the grid will be created as a rectangle (2D) or cuboid (3D) between those two points). Also check the corresponding **checkboxes** if you want to prevent the *SnapItems* to move out of the grid

- Advanced tab: Set your manager's snapping **accuracy** (which is the allowed radius around a snapping / rasterization point) if you want to allow your SnapItems to be a bit displaced of the snapping points (looks more natural sometimes).

- Check the corresponding checkbox if you want to enable snapping your *SnapItems* to a terrain or to a specific object (terrain snapping is automatically locked to the Y axis, while object snapping can be enabled per-axis). If so, please drag your terrain or object from the Hierarchy window into the proper **terrain/object field**. Enter your desired **position offset** into the corresponding input field(s) below if the objects are not exactly at the right place (this will affect ALL *SnapItems* at once)

- If you want to chance the position of a SINGLE *SnapItem*, check out it's attached *SnapItem* script and change it's **position offset** over there

- Each manager creates one grid. Dragging more managers into the Hierarchy Window will add more grids. To use them simultaneously, please change the **manager ID** of each manager to a unique number (auto-increases since v1.3) After that you'll need to change each *Snap Item's* **manager ID** to the ID of the manager it should work with

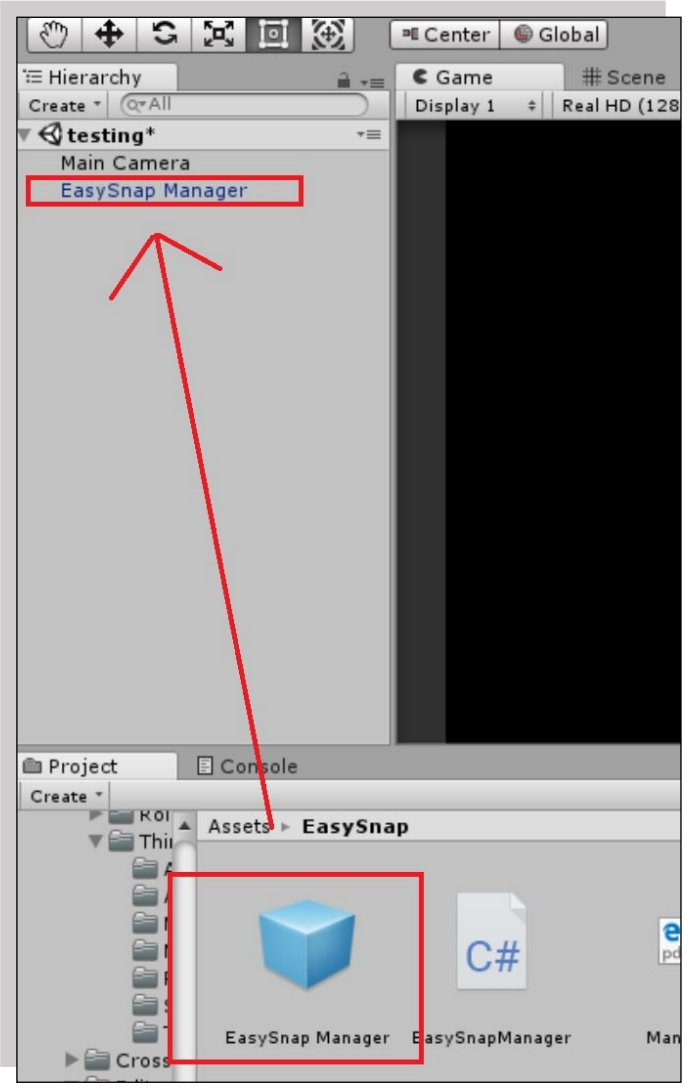
Now you should have learned the basics of EasySnap. Check out the editor-only demo scene in the *"Demo"* folder and play around with it a bit or simply read further ;-)

Bug found? https://bugs.ondeth.de/open.php?lang=en_US
Questions? Check out the ***long version of this manual first***! If that doesn't help: mail@ondeth.de

Getting started (the longer version)

Simply drag and drop the prefab EasySnap Manager into the Hierarchy Window and click on it there.

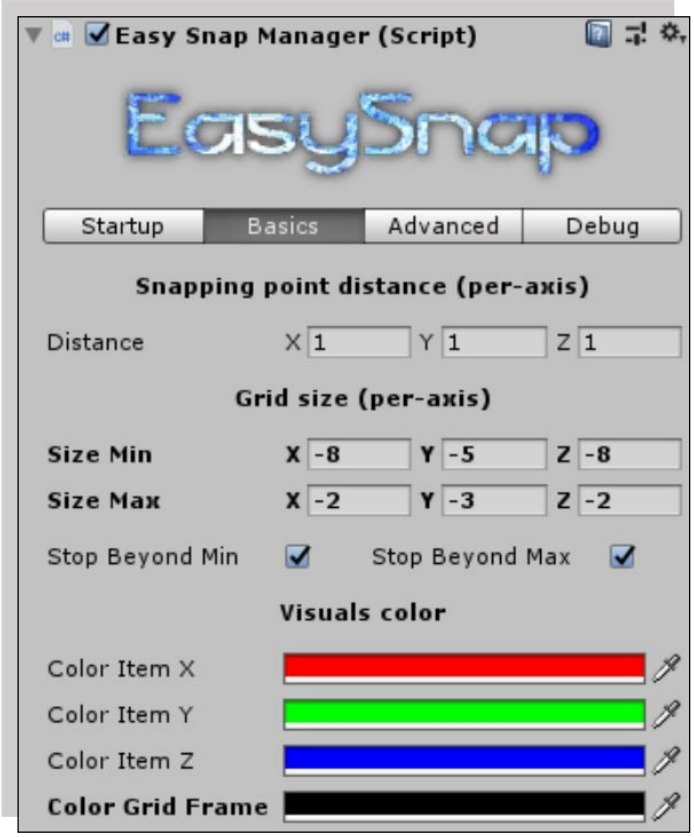
Hooray, you've mastered the first important step!
Let's continue with the slightly more complex part!



The configuration window (Aaargh configuration…)

Well, it is not that complex – but everything is more complex than drag'n'dropping a file, isn't it? ;-)

Welcome to the heart of EasySnap:



The EasySnap Manager script

On the following pages, I'll describe the function of each of those checkboxes, text and color fields - so stay tuned!

The configuration overview
(Much easier than it seems)

The ”Startup” tab

Enable EasySnap: EasySnap can run in edit mode (check the *Enable in edit mode* box) and in play mode (check *Enable in play mode*).
Checking both checkboxes will enable EasySnap in both modes, unchecking both will disable EasySnap completely.

Enable axis-specific snapping: Here you can enable rasterization for each axis in Unity:

On the X axis: Enables rasterization on the horizontal axis (needed for 2D & 3D)
On the Y axis: Enables rasterization on the vertical axis (needed for 2D & 3D)
On the Z axis: Enables rasterization on the “depth” axis (only needed for 3D)

To make it short: Simply check those axes, which should have the “rasterization grid” effect.

To make it long: If you are planning to create

- a **2D game** and you want **snapping in both directions**, simply check *On the X axis* and *On the Y axis* (a top-down chess game for example)
- a **2D game** and you want snapping only on the **X axis,** only check *On the X axis* (a very basic side-scrolling game for example)
- a **3D game** and you want snapping **in all directions**, simply check all checkboxes (a 3D puzzle game for example)
- a **3D game** and you want snapping **only on the Y axis**, only check *On the Y axis* (an elevator management game for example)
- a **3D game** and you want to… well I think you’ve understood already, let’s continue!

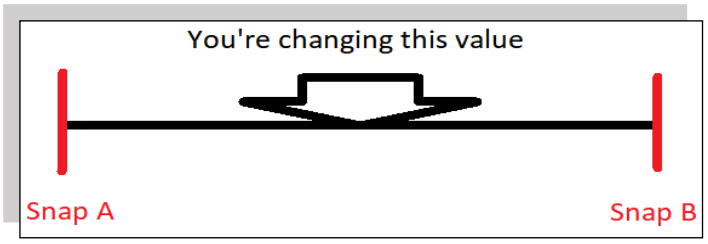
Enable visuals: Those two checkboxes control the visual grid in edit mode.
Checking *Show Grid Frame* will enable the visual grid outline, which makes it a lot easier to get an overview over the created grid.
Checking *Show item moves* will enable the item-specific movement axes to get an overview over all currently possible moves for each item.
Checking *Show snapping points* will visualize the item-specific snapping points available at the current item position to get an overview over all currently possible snap points for each item.

Manager ID: This is the „group of snapping objects“ this manager controls.
Changing the *Manager ID* value is only required if you are using multiple EasySnap manager (if you want to use multiple snapping grids simultaneously).
You will find more information on using multiple snapping grids at the end of this manual.

The “Basics” tab

Snapping-point distance: Here you are defining the distance between two snap points in Unity coordinates for each of the axis individually. Simply change those values the way it fits your game

- A smaller size means smaller spaces between two snap points and a less jumpy movement.
- A larger size means more space between two snap points and a better rasterization.



Changing snapping-point distance

Grid position / size: Those two Vector3 values (X, Y, Z values) define the position of the rasterization grid (*GridPosMax* for the max and *GridPosMin* for the min point in the world, the grid will be created as a rectangle (2D) or cuboid (3D) between those two points).

There is no snapping / rasterization beyond the grid. If you want to prevent items to move out of the grid, simply check the *StopBeyond* checkboxes (*StopBeyondMax* prevents moving out of the max and *StopBeyondMin* out of the min coordinates).

Visuals color: Here you can change colors for the edit mode visuals (which you can enable by checking *Enable Grid Frame* and *Show item moves* – you remember? ;-)

The three colors named *Color Item X*, *Color Item Y* and *Color Item Z* are used for each item-specific movement axis of this manager, that means changing *Color Item Y* to pink will make all manager - related items Y axes pink (again: only those with the same manager ID will get pink, don’t forget).

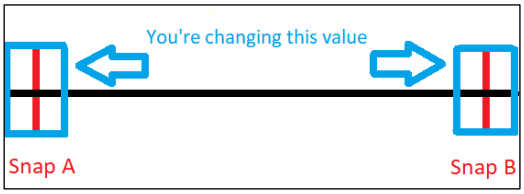
The next color (named *Color Grid frame*) is used as the color of the visual grid outline.

The last color setting (called *Color Obj Link* in the manager) will define the color, which will be used to indicate the link between SnappingItems and their object to snap to.
You’ll find more information about object snapping settings on the next page.

The “Advanced” tab

Set the accuracy of snap checks: Increasing the *accuracy* value will increase the allowed snap section per-axis. That means it‘ll allow the objects to be a bit displaced on the axis without being repositioned to the nearest snap point. This feature helps you to provide a more natural-like look for example.

Let’s say you’re developing a 3D chess game: Increasing this value will allow the chess pieces to be a bit displaced from the current tile center of the chess board, which is much more realistic than having a chess board with perfectly centered chess pieces.



1

Terrain snapping: Enabling the *Snap to terrain* checkbox allows you to snap the Y-axis to a terrain instead of snapping to the virtual grid. Simply drag and drop your active terrain item from the Hierarchy window into the *Terrain to Snap* box and you‘re ready to go.

The *terrain offset* value enables you to adjust the height of all SnappingItems of this manager in case they’re too low or too high.

Object snapping: Enabling the *Snap to object* checkbox allows you to snap all manager-related SnappingItems to a specified object instead of snapping to the virtual grid.

Simply select the axes you want to snap by checking the appropriate checkboxes (*Snap X axis TO*, *Snap Y axis TO*, *Snap Y axis TO*) and then drag and drop your desired object from the Hierarchy window into the *Object to Snap* box and you‘re ready to go.

The *object offset* value enables you to adjust the position of all manager-related SnappingItems in case they’re not in the correct position.

Snap checks intensity (in frames): This is optional stuff. Normally, EasySnap checks each frame (*Intensity* = 1), if there’s anything to snap to the grid (that’s called **a snap request**). For a better performance, you can increase this value to 2 for example (so it only checks every second frame). **Nevertheless, be warned: Increasing this value may cause a serious delay in correct grid snapping.**

The “Debug” tab

Log X, Y, Z axis to console: This debugging feature allows printing out each successful snap on the console (per-axis). Enable it to see at which time which object was placed to which position.

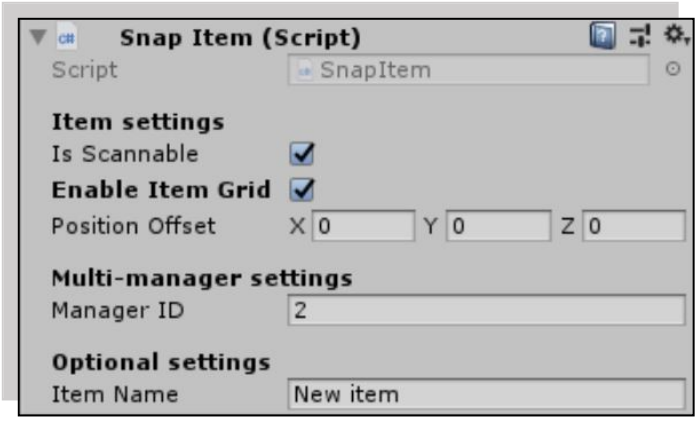
Log snap requests: This debugging feature allows printing out each snap request on the console. Enable it to check how often EasySnap checks if there is anything to snap to the grid. Be warned, that this debugging feature will create massive spam in your console ;-)

Log snap procedure: This debugging feature allows printing out the current code segment, which was used to process the snap request of an item. Be warned, that this debugging feature also will create massive spam in your console, so disable it as long as you can ;-)

Okay great, you’re done with the user interface and you’re *almost done with EasySnap*!
Wasn’t that hard, was it? But one important thing is missing – any idea what?

Sure thing! EasySnap Manager needs to know, which objects it needs to snap to the grid!

The SnapItem component (The object “marker”)



Marking an object as a “snapping” object is really easy!

Create a new empty GameObject: Simply drag’n‘drop the EmptySnapObject prefab into the SceneView to create a new empty object with a SnapItem script already attached.

or use an existing GameObject: Simply drag’n’drop the SnapItem script (in the folder called *Scripts*) onto each existing object you’ll need to snap.

That’s all!

Using multiple grids simultaneously (More is always better!)

EasySnap provides a very easy way of using multiple snapping grids at the same time!
Simply drag’n‘drop as many *EasySnap manager prefabs* as you want to have grids into the scene.

Each manager needs its own, unique ID to work properly. Since EasySnap v1.3, dragging a new manager prefab to the scene will automatically increase its manager ID (starting with 1)
(Example: *Manager 1 uses ID 1*, *Manager 2 uses ID 2*, *Manager 3 uses ID 3*, …)

Manager ID’s, which are existing twice or should be changed can be automatically corrected by clicking the “Auto-set manager ID” button. EasySnap will then assign the lowest possible (still unused) ID to this manager (starting with 1)

Finally you can select the manager, which controls a specific snapping object, by changing the Manager ID in the snapping object‘s *SnapItem component* to the ID of the desired manager.

Congratulations, you’re done!

On the following page, I'll describe the advanced settings of the SnapItem component, which are not required to get things started

Advanced SnapItem settings overview (Configuration…again…)

Is scannable: If you want to disable an object's snapping property temporarily, disable the checkbox called *Is scannable*. Then, this object is invisible for the corresponding EasySnap Manager and you may move it freely around.

Enable item grid: Uncheck the checkbox called *Enable item grid* if you want to disable the visual item-specific movement axes *for this item only*. The movement axis of all other items will stay enabled.

Position offset: Those Vector3 (X,Y,Z) values are only needed when *Snap to object* or *Snap to terrain* is enabled in the corresponding EasySnap Manager and if you want to change the offset for this item only (Remember : In the EasySnap Manager you can change the offset for all corresponding items simultaneously.)

Then, only this item's position will be corrected and not the position of every item associated with the manager.

Manager ID: This object will be controlled by the EasySnap Manager with this ID.
Changing this is only necessary if you are using multiple EasySnap Managers (if you want to use multiple snapping grids simultaneously)

Item name: Here you are able to name your object so you can differentiate between multiple objects when enabling the debug features. This feature is purely cosmetic!

Conclusion (Got that?)

That's the end of my small EasySnap manual. I hope that you've understood everything, but I think, it's not that hard.

If there's anything unclear, feel free to write me at mail@ondeth.de!
Best regards!

P.S:
If there's a bug lurking around, please do not hesitate to open a bug report ticket over here:

https://bugs.ondeth.de/open.php?lang=en_US

I'll look into it as soon as possible! :-)