Game Editor Manual

Company | [Company address]

Everyhing you need to know about editor

bojan cup

2012

Contents

[Introduction 3](#_Toc348998980)

[How to find the tools and support 4](#_Toc348998981)

[Folder structure after installation 5](#_Toc348998982)

[Bin Directory 5](#_Toc348998983)

[Resources Directory 5](#_Toc348998984)

[Utils Directory 6](#_Toc348998985)

[How to share what I made in the editor for others to see? 6](#_Toc348998986)

[What does game load by default, which level? 6](#_Toc348998987)

[What kind of editor it is anyway? 7](#_Toc348998988)

[Are there any settings for the editor? 7](#_Toc348998989)

[How to use the editor 8](#_Toc348998990)

[Navigating through the editor 9](#_Toc348998991)

[Layers and the Main Layer 10](#_Toc348998992)

[How to add a sprite into the layer? 10](#_Toc348998993)

[Where can I see properties for this object? 12](#_Toc348998994)

[Position 13](#_Toc348998995)

[Origin 13](#_Toc348998996)

[Rotation 13](#_Toc348998997)

[Scale 13](#_Toc348998998)

[Tint 14](#_Toc348998999)

[Transparency 14](#_Toc348999000)

[Flip Horizontally 14](#_Toc348999001)

[Flip Vertically 14](#_Toc348999002)

[Primitives 15](#_Toc348999003)

[Game entities 16](#_Toc348999004)

[Player 16](#_Toc348999005)

[Exit 16](#_Toc348999006)

[Game Enemies 16](#_Toc348999007)

[How to run the game with the current level 17](#_Toc348999008)

[Troubleshooting 17](#_Toc348999009)

[Game physics 18](#_Toc348999010)

[Physical shapes 19](#_Toc348999011)

[Making custom shapes and shapes database 20](#_Toc348999012)

[How to load Custom Shapes Database 21](#_Toc348999013)

[Making new custom shape 21](#_Toc348999014)

[It’s good to define shape for only what you need 23](#_Toc348999015)

[Shape scaling 23](#_Toc348999016)

[One last thing 24](#_Toc348999017)

[Game behaviours 25](#_Toc348999018)

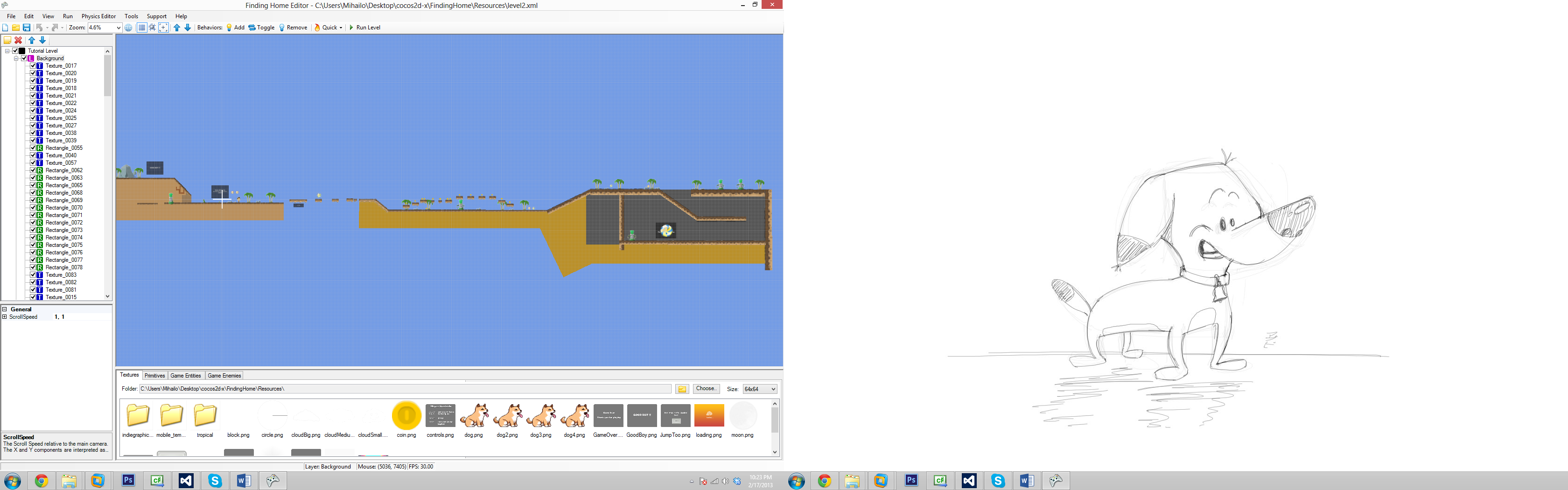
[Example levels and notes 26](#_Toc348999019)

# Introduction

Welcome to the Finding Home (working title) level editor manual.

The purpose of this document is to give you some clear insight on how level editor works so you can jump right in and make amazing worlds that always inspired me from early childhood.

This is how the editor looks like when full screen. It is based on old Gleed2D source code and heavily modified to make level editing for our engine and this particular game a breeze.



Editor is NOT the only utility we need to create complex maps, for custom body shapes we use Physics Editor, but more on physics and all that later.

Read on.

# How to find the tools and support

You can download the editor and the tools from our forums <http://recapture.freeforums.net/thread/4/lastest-builds>.

Links uploaded there always contain the latest build, so even if you don’t see any new posts newer update tools from time to time, until automatic update checking is completed (scheduled for development).

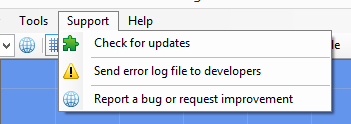
You will find two different download options on our forums:

* **Setup.exe** - contains latest playable demo in release mode, if you want to brag to your friends or send potential employer – this is what you are looking for)
* **SetupEditor.exe** – contains game executable but with level editor and other tools used for development. Since its planned to be used only internally so things here may be a bit unstable, but don’t worry the techies are already working on fixing that.

If you have any issues in running the game or the tools see that section about what prerequisites you need to install. I will keep that up to date as well.

Please note that many features in the tools are pretty new and still not heavily tested, so if you have issues let me know. I created separate board for that specific purpose <http://recapture.freeforums.net/board/13/support-improvements>.

**And don’t worry about remembering all those URL’s, everything regarding support is integrated into editor already.**



I encourage you to send me your error log files after each crash with brief description about how and when it occurred (what were you trying to do).

Also if you want to report specific bug you discovered or or report an improvement you can do so with “Report a bug…” link – it will open our support forum board for you.

# Folder structure after installation

Every tools have their own way how they organize files and directories and we have some rules too (pretty simple stuff don’t worry).

The tools install to your desktop for your convenience. Uninstall tool is included for easy removal as well - **but be careful when you are using it** as it may remove some of your asset or level files if not backed up.

After the installation of the tools you should have following file structure:

* Your desktop:
  + FindingHomeTools
    - Bin
    - Resources
    - Utils
    - Uninstall.exe

Some more files may be present but they mean little so I won’t cover them here.

## Bin Directory

Contains game engine executable and other binary files required for game to run. Nothing here is editable so you probably shouldn’t even care ;)

But in case you want to start the game without the shortcut find **“FindingHome.win32.exe”** here and double click it.

## Resources Directory

Remember the rules I mentioned at the start of this page? Well we actually have only one rule for now: ALL CONTENT GOES TO THE RESOURCES DIRECTORY.

Pretty simple, your level files (XML) and your PNG’s can be organized however you like, **but put it all and keep it all in the Resources directory**.

Also changing and rearranging resources after the level is made can break your level (resources not where they should be) so be careful about that as well.

In resources directory you will find:

* XML files (usually game levels, but some configuration files could be XML too)
* ShapeData.JSON (used by Physics Editor – this file is mandatory)
* Loading.png (Used for loading screen)
* weather\_controller.png (Used for controlling weather system – duh – and mandatory ;)

Everything else can be modified, but best practice is to leave those files that are installed be, and create your own directory inside resources dir.

## Utils Directory

Contains the tools we are here talking about, and their setting files. Do not modify or remove anything by hand here.

Structure:

* gleed2d.exe – This is the actual editor executable, so if you don’t want to use shortcut on your desktop you double click on this as well
* physics-body-editor.jar – This is the executable for the Physics Editor. It can be run through the Level Editor but you will find it here in case you have issues.

# How to share what I made in the editor for others to see?

Well, since we already agreed that everything goes into the Resources directory you could just zip that whole directory and put it on Dropbox or email it if not too large.

**To make it easier for others you may want to zip complete “FindingHomeTools” directory on the desktop (just Right Click / Send to compressed folder) and share that zip file. Others can extract your archive and open it manually in editor (see Folder structure above).**

This is just for now since we don’t figure some better approach, and once we set up some kind of file sharing system between everyone it will be just adding files to Resources (then shared) and making level files ;)

# What does game load by default, which level?

For now (as we have no main menu yet) the game load Level2.xml file from resources.

This will very soon change and you will have option to change initial load path in some form of settings file, I will let you know.

So for now if you want to share a game that loads your level from the start just name your level Level2.xml and remove existing one. It should be fine.

# What kind of editor it is anyway?

Well, if you would expect tiled editor that works with predefined images of fixed size - you would be dead wrong.

Although it is a 2D editor it works with images of any size and without needs for making tile set or create a house from 10 different blocks or anything.

What that means for you, the content creator, is that anything PNG can be used directly into the game without processing. Just make it in Photoshop or download from the internet, put in resources directory and insert into the level.

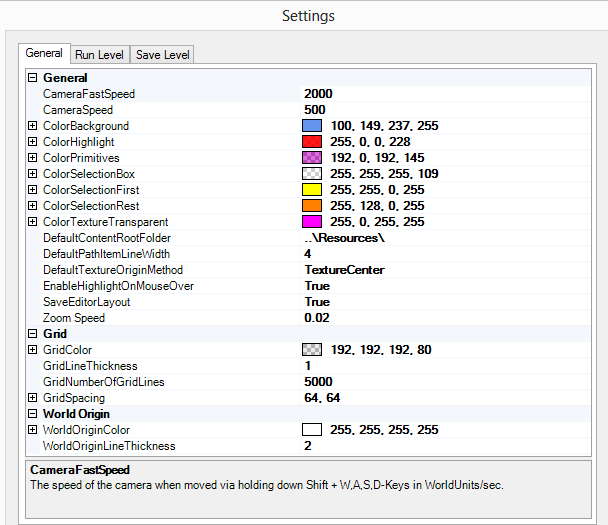
It’s that easy.

PNG is the only supported image format currently.

To make sure the game runs smoothly make sure your image files are not too large (I mean like 3MB per image). At least for now, without further engine optimization.

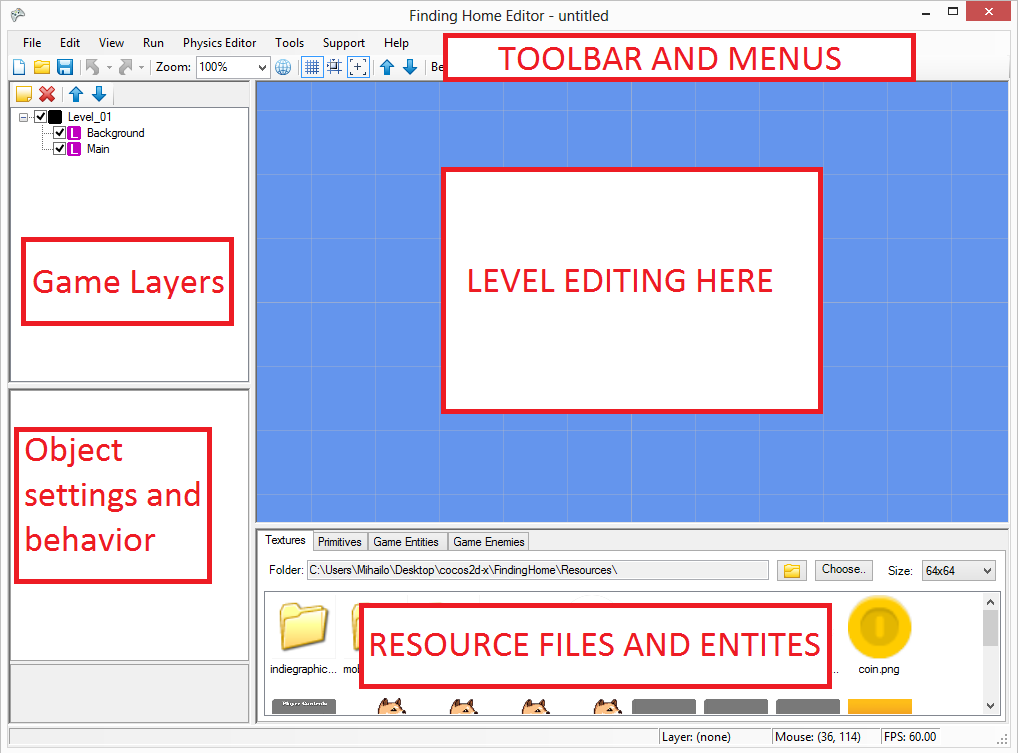
# Are there any settings for the editor?

There are, but very simple stuff for now. You can see what we have for now using Menu/Tools/Settings option.



# How to use the editor

The best way is always by playing in it and experimenting, so feel free to do so. Here is the quick overview of what goes where.

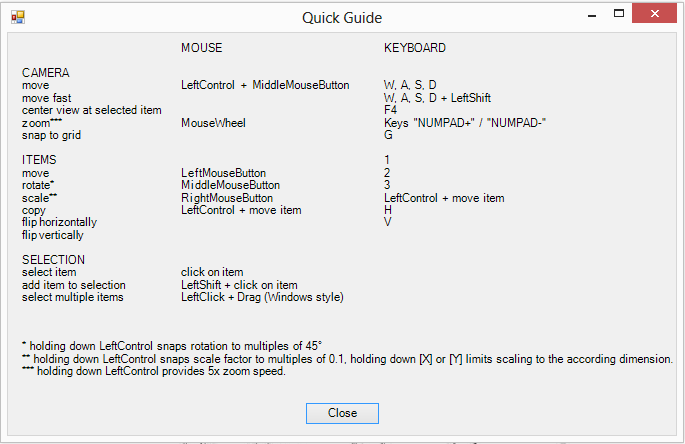


I will not explain every single feature as there are many and will constantly be changed as the editor is improved, I tried to make a tooltip description for every button and to correctly name everything so you won’t get lost if you pay just a little attention.

I will however explain a few basic things needed for a simple level and what kind of entities there are.

## Navigating through the editor

We navigate through level using keys and mouse, and full list of supported (well maybe not full full) commands can be found in editor itself using menu Help/Quick Guide.



Few important notes (and we can update keyboard layout later to suite your needs better):

* You move level with Ctrl + middle mouse button
* Control does not add to selection – it copies the selected item!
* Shift adds item clicked on to current selection
* You can SHIFT W/A/S/D for faster camera movement
* Holding G will enable snap to grid when placing editing or moving any kind of object
* You can zoom in and out with mouse and keyboard, but hold SHIFT to speed it up
* Hold right button / middle button on sprites to scale and rotate them

Bug #1. For now you cannot nudge items as you probably used too using the Left/Right/Up/Down keys as Windows Forms takes away focus automatically. I will work to bring nudge feature soon.

## Layers and the Main Layer

The editor creates two basic layers (for your convenience) named Main and Background so you can jump right in when it starts. Both layers can be removed.

Everything about the layers is as usual and seen in many other applications, except for the “Main” layer (the only difference is the name but engine parse it differently).

Later as we talk about the Behaviours and how player gets to walk over and push things around you will learn more, but for now remember that all items in Main layer have Solid behaviour enabled (physical, and can’t be disabled). This is made to speed things up and you don’t have to use the Main layer at all, but it is quite easy to create a walking ground for the dog and keep it all in this Main layer so you don’t need to add Solid behaviour for every new item player needs to step on.

## How to add a sprite into the layer?

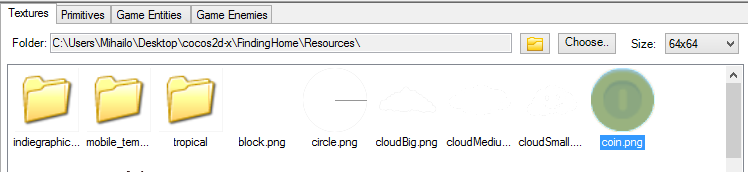
Sprites are 99% of your content so let’s start and add one to the level already.

Before you can add the sprite you need first to select the parent layer you want to insert layer into.



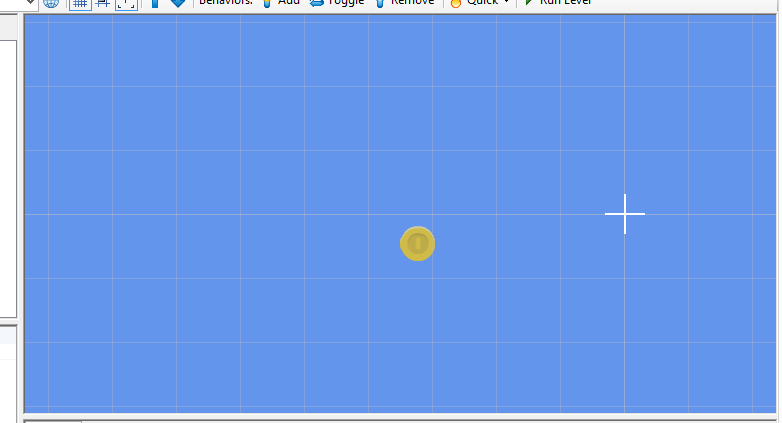
You can see here the main layer is selected.

After that select the sprite as in image bellow, and double click on it.



This will tell the editor (blue screen in the middle) that this is the image we want to insert.

Now move your mouse into editor area and left click it.

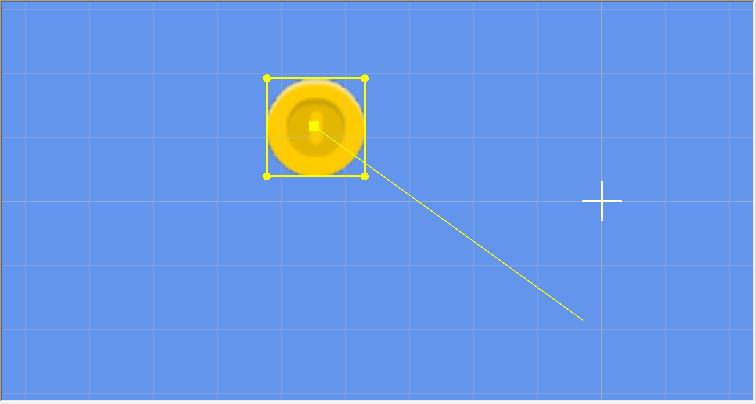


This will insert your sprite and voila you are done.

To select an object just left click on it.

Most objects (sprites among them) can be scaled and rotated by holding right mouse click and middle mouse click, respectfully.

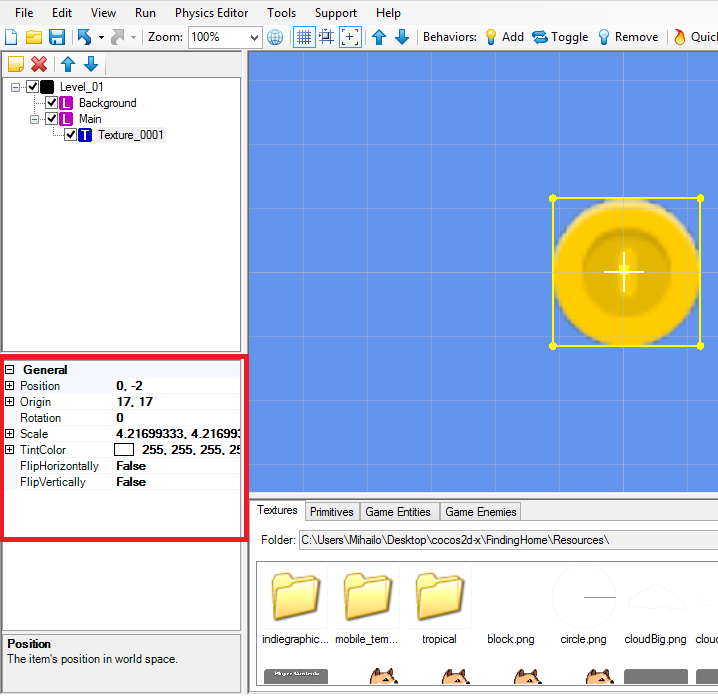
Here is the example of the same sprite just scaled a little bit.



You can see angle and boundary of the selected sprite so you know how big it will be once you release the right or middle click, depending on what are you trying to do.

## Where can I see properties for this object?

If the item is selected in the map editor or in layers portion of the editor on the left side you will see all properties that object has.



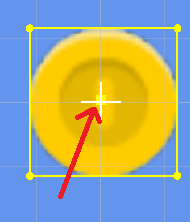
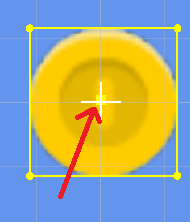
Not all entities have the same options and some, like a Player (see below) have almost none, but most of them will have similar number of similar properties.

Because of that I will explain only properties for Sprite object, and you can see for yourself properties for other specific entities.

### Position

Defines 2D coordinates (or vector) of the sprite.

Position of the sprite depends on where editor considers to be the anchor point, or point of origin is.



Anchor Point or Origin

In our editor point of origin is always in objects centre. This way the position stays the same even if object is scaled to different size.

### Origin

Has no effect.

As we previously covered anchor point can be any value, and different engines use different settings but we here use object centre or (0.5, 0.5) anchor point. Editor sees it differently and gives different values that the game so this setting in the end (for now at least) has no effect.

### Rotation

Item rotation in radians.

### Scale

Defines how many times the object is larger than original sprite.

Objects can be scaled by X axis or Y axis, or to keep original proportion of the item simultaneously.

For now only X axis is calculated when checking how to scale object. This will be fixed.

### Tint

Affects object colour and transparency.

Your regular RGBA model colour defines how sprite will be rendered and how much transparency it can have.



Same Coin sprite with R:0, G:192, B: 0, A:255

(Looks like the coin was in the water for too long ;)

### Transparency

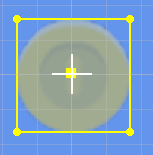
There is no separate transparency property of an object, but we can use Tint or Colour property of an object to make this happen, and this is how we do it.

Since white colour has no effect on the sprite we can use that with alpha set to what level of transparency we want.

We set color to white (R: 255, G: 255, B: 255).

If we set A channel to **255** (full white colour with full opacity) the item is fully visible and colour is not changed. This is the default setting for all objects. (Resulting 255, 255, 255, 255).

But if we set A channel **100** (partial transparency) we get semi-transparent image. (Resulting in 255, 255, 255, 100).



Partially transparent sprite

### Flip Horizontally

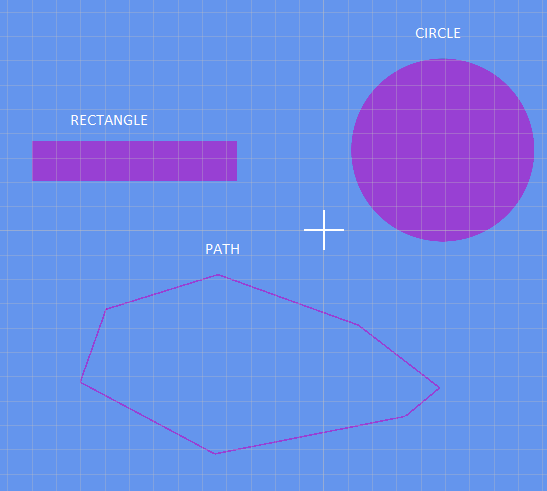
Makes sprite change direction. Try it out.

### Flip Vertically

Makes sprite upside down. Try it out.

## Primitives

Much like the sprite entities you can use primitives to build your level on, also they are more likely to be used for quickly prototyping the level when no textures are available.



Another primitive I planned to add is a text primitive, which could be used for 2 things:

* (visible) Posts in game, text on buildings, …. Probably not going to happen as assets will have all details on them already.
* (visible only in editor) So we can leave each other notes in editor, such elements are not visible in game.

Path is currently unsupported in the editor as I’m not sure what we need it for.

I was thinking that good use for it could be making camera move with path, for intro or outro animations in levels, or when some new entity is introduced, we could animate camera to move with path.

## Game entities

All items cover until now used in any games, but Game entities are specific to ours. I tried to name them so you won’t have any issues differentiating them, and if need arises I can make a full list of entities in separate document (or better help directly in editor) so you know what does what.

### Player

Player object represents player in the game but also an entry point. Can be placed only once.

Object has only position property, and wherever object is placed is where the level starts.

NOTE: I planned to add Direction property, so player can face left or right at the start.

### Exit

Represents exit points and can be placed multiple times.

If Player object walks into the Exit object engine will trigger scene transition and level will change.

For now Exit object has width/height/scale properties (how large the area of exit is).

It also has a **Level** property that contains the filename of next level to be loaded, without the full path (since everything is in the .\Resources\), so if you want to load Level5.xml when player enters this area you just enter that.

Exit object is NOT visible in the game, it has no visual representation.

If Exit object has empty Level property or file does not exists the same level will load again. (Can be changed, let me know).

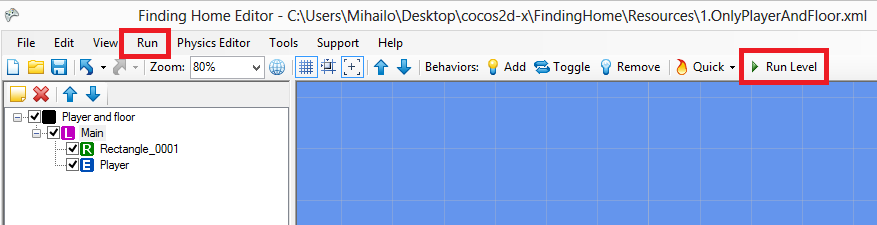
## Game Enemies

Let’s skip this for now since nothing is really implemented yet.

## How to run the game with the current level

After you have made your fist level and you saved it you can easily run the level by hitting **F5 key** or using **Run level** button in the toolbar.

In case the level was not saved you will be asked by the editor to do so then. If even then you don’t save the editor.



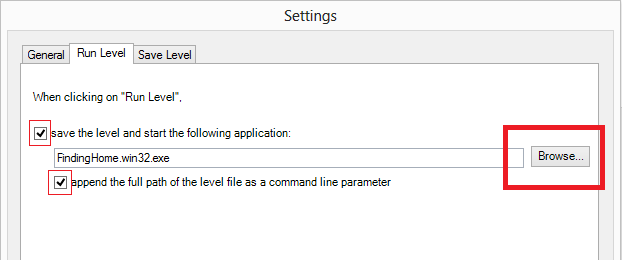
Please note that the game can have only one instance, or in plain words it can be run only once so if nothing happens when you hit that F5 check if it’s already running.

### Troubleshooting

If folder structure is correct and editor settings.XML file is in place you should not would have any issues running the game as everything is set up.

But in case you do and editor cannot find the game you can tell the editor where it is, using Tools/Settings menu item and **Run Level** tab.

Just check checkboxes as they are here and hit browse to find game executable (usually FindingHome.win32.exe) and click save, you should be fine.



# Game physics

Our engine uses Box2D for real physics simulation and has been used in many games and has been ported to many languages including desktop C++, hybrid python or full web JavaScript.

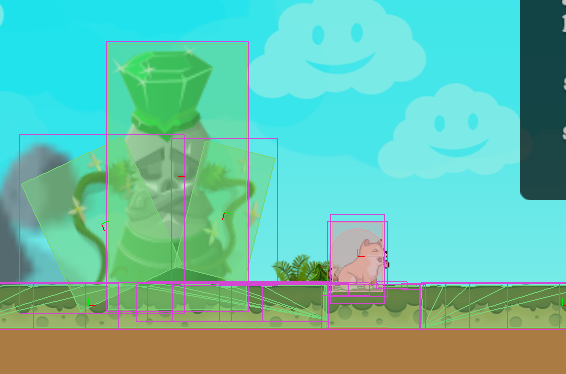
We use C++ implementation as it is the fastest one.

Many games today use this engine for rigid body simulations and Angry Birds is probably one you are familiar with the most.

That does not mean engine is all-possible, this is an excerpt from Box2D manual on what the engine is capable of and how it is tuned.

|  |
| --- |
| Box2D has been tuned to work well with moving objects between 0.1 and 10 meters. So this means objects between soup cans and buses in size should work well. Static objects may be up to 50 meters big without too much trouble. |

Engine and tools will shield you from ever doing any computations by hand with something easier to work with – Game Behaviours – but you need to know how engine processes items, and it is all about the shapes.



F1 key - Enables physics debugging

Note that Game Behaviours term is valid for non-physical behaviours as well.

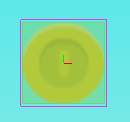
Also note that only basic physics features are implemented for now, and that for more complex puzzles with more movable parts and joints (no not those joints) this part might be heavily expanded, but I’ll try to make everything definable through behaviours, as much possible at least.

## Physical shapes

Box2D knows 3 distinctive types of physical shapes:

* **Rectangle shape** – used for every sprite by default
* **Circle shape** - used only for circle primitive
* **Polygon** (or irregular) **shape** – created using Shape Editor

Every sprite inserted into system has Rectangle shape by default. If you place object in Main layer (so it will get physical properties even without behaviour) you can see this easily.



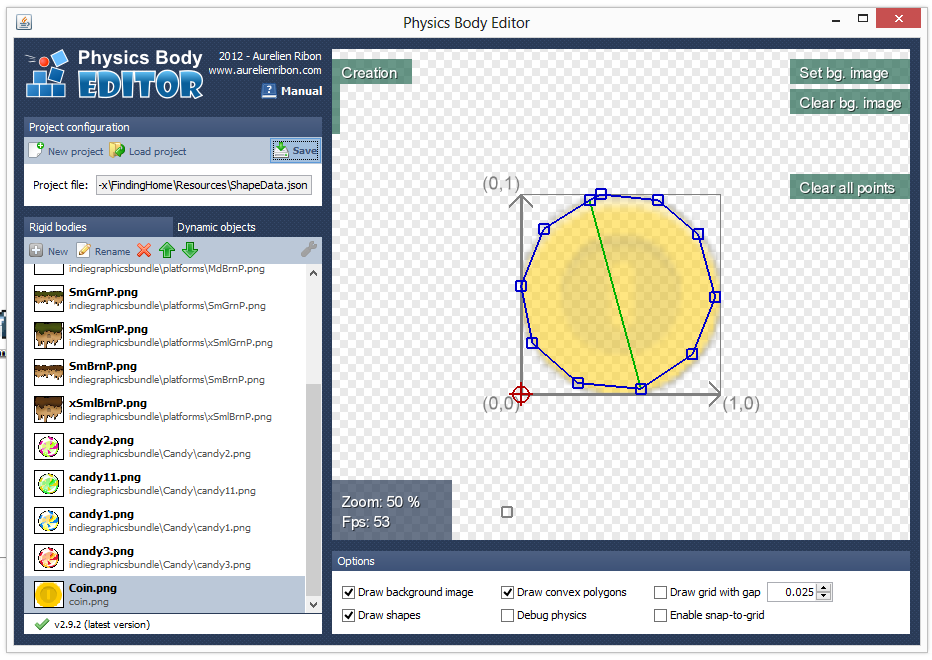
The same coin, although round in presentation it has rectangle shape by default.

Since it would be quite silly for a coin to be rectangle we need to define custom shape for a sprite we want to use that has physical properties (I will repeat, everything in Main + whatever has behaviours).

We define custom shape using a number of polygons to define it, and sometimes for complex shapes a very large number of polygons may be needed.

To be honest, even I would not know how to do that by hand. This is why we separate utility to create custom shapes for us.

## Making custom shapes and shapes database

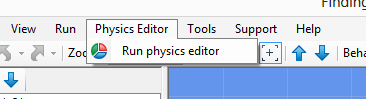


Physics Body Editor in action

Physics body editor is a tool written in Java for a game framework named libGDX by a guy with unusually cool and game like name of Aurelien. (If anyone knows the guy give him my thanks!)

The tool is quite simple to use and I made sure we have full support in the engine for it.

You can find editor in .\Utis\ directory (need Java), and I made a shortcut for it anyway directly in Level Editor.



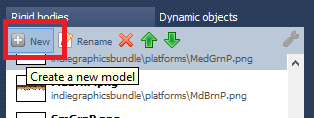
### How to load Custom Shapes Database

All custom shapes are contained in single JSON file named **ShapeData.json**.

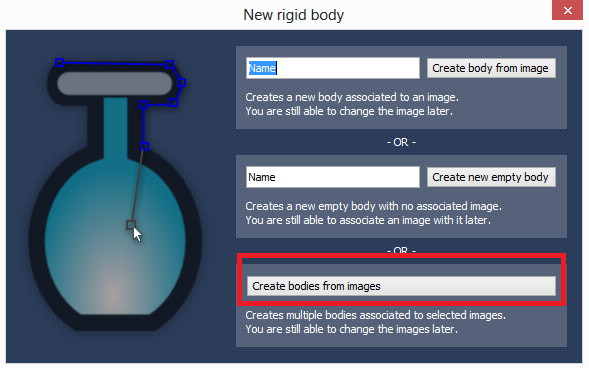
Contents of this file are not important, but think about it like a Project or database, if you load that file using **Load project** -> **.\Resources\ShapeData.json**, you will get nice list of all shapes available for you to edit, review or remove.

### Making new custom shape

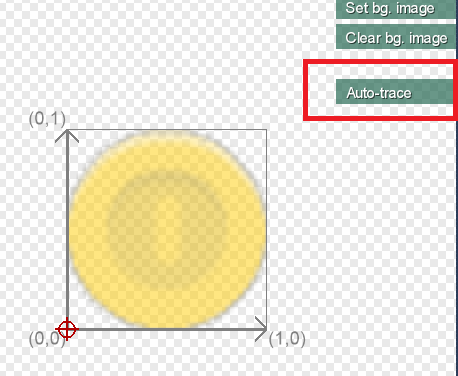
Easy as a pie! First make sure database is loaded from Resources directory. Then hit new.



Then create body from images.

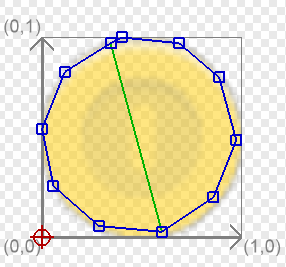


You should now have something like this (no shape defined just background).

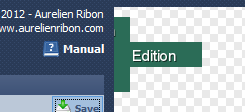


Then hit Auto Trace, or create your own shape by adding points that make custom shape.

After that you should have something like this:

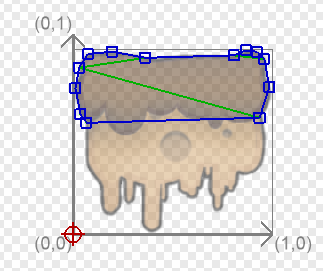


Now you can (if necessary) edit the shape to more precise points, using the Edition mode.



**Hit save button and you should be ready!!!**

### It’s good to define shape for only what you need



This is a great example of how shape can be only partially defined, and it is not also useful for level design but it also saves a lot of processing time when shapes are relatively simple.

|  |
| --- |
| **I cannot stress enough that you should keep your shapes as simple as possible, as complex shapes cause more overhead and can keep FPS low when they are many of them.** |

### Shape scaling

You may wonder OK, I define shape in Physics Editor, does my body in the Level editor has to be the same size, or unscaled?

No. Engine will automatically recognize what scale the object in editor is and will do necessary adjustments.

So go wild, but don’t forget there are some limitations of how big or small objects can be.

### One last thing

Physics Editor has nice documentation of its own so I won’t get into more details here, do check it out if you get stuck or let me know on forums or via Skype.

**It’s important to know that game engine recognizes shape by its name that has to be identical to filename of the asset.**

So if you are defining shape for image named .\Resources\Tropical\**Coin.png** shape name has to be **Coin.png**.

In case you used **Create Bodies from Images** button above that should be done automatically.

If not check if the filename is the same as the name of shape.

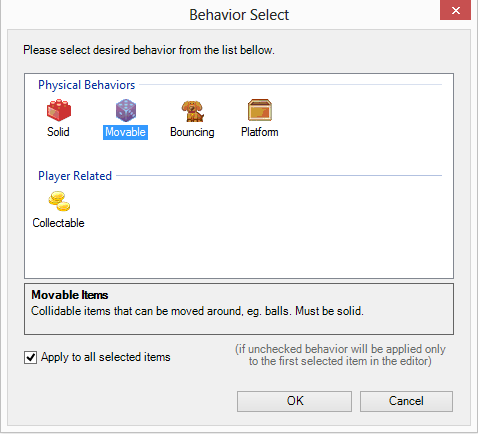


# Game behaviours

And we finally came to more interesting part of the editor, object behaviours.

Not to go purple here, Object behaviours define how game engine interprets objects and what player can do with them.

I think this image shows everything you need to know about the options available to you, at least since there are no complex behaviours that are not self-explanatory.



Try experimenting with them all, I tried to be detailed in how to use them in box bellow selection.

Also notice that in order for some behaviours depend on other behaviours in order to be enabled.

More noticeable example of this is Movable behaviour, in order to be enabled object has first to be Solid (or what are you going to move if it’s not) or has to be in special Main layer.

# Example levels and notes