# Tutorial

### *Page 1 of 15*

## Your First Game

So, you have gotten yourself a copy of *GameMaker: Studio* and want to make games? Well, this tutorial is the best place for you to start! Here we are going to introduce you to most of the main concepts behind the programme and hopefully show you how simple it is to get a game up and running, not just on your PC, but on your mobile device or web page too.   
  
  
  
This tutorial aims to teach you how to find your way around the *GameMaker* IDE, as well as how the basic components of a game are created, linked to resources, and then placed into rooms to form a final game that can then be distributed for others to play. The game itself that we are going to make is a very simple one called "Catch The Clown", and we are going to make it using the proprietary *GameMaker* Drag'n'Drop system (which will be shortened to D'n'D from now on), although further tutorials available to you will focus more on the code aspect of making a game.   
  
At various points in this tutorial you will be directed towards the "**Catch The Clown Assets**" folder. It can be found in [this folder](file:///C:\Users\Jyothi\AppData\Local\GameMaker-Studio\Tutorials\My_First_Game\Catch_The_Clown_Assets\#_blank), and you should have it open and ready as we will need a few things from it for our game.   
  
***NOTE:*** *If you can't see some of the images in this tutorial correctly, or they seem too small, you can expand the tutorial pane by dragging the left window border into the IDE.*

##### © Copyright YoYo Games Ltd. 2015 All Rights Reserved

# Tutorial

### *Page 2 of 15*

## The Game Idea

Before jumping into the action, we should take a moment to plan out our game. It's all too easy to sit down to start working on an idea and then later find that you have lost focus and that the game is not what you actually intended it to be! This in turn leads to an abandoned project and a sensation of having wasted your time, which is obviously not something you want to happen. Therefore, we should make a very simple **design document** which outlines all the important things we want in our game. Once the game has been completed following this document, we can go back and revise and change things.

## Design Documents

Design documents can take any form you wish, like a full blown essay with doodles and art clipped onto it, or a simple note with some bullet points. The important thing here is to give your project a definite direction and have a list of goals that you want to achieve. Here's our design document for "Catch The Clown":

### Description

"Catch the Clown" is a little action game. In this game a clown moves around in a playing field. The goal of the player is to catch the clown by clicking with the mouse on him. If the player progresses through the game the clown starts moving faster and it becomes more difficult to catch him. For each catch the score is raised and the goal is to get the highest possible score. Expected playing time is just a few minutes.

### Goals/Resources

* **Game objects** - There will be just two game objects: the clown and the wall. The wall object has a square like image. The wall surrounding the playing area is made out of these objects. The wall object does nothing. It just sits there to stop the clown from moving out of the area. The clown object has the image of a clown face. It moves with a fixed speed. Whenever it hits a wall object it bounces. When the player clicks on the clown with the mouse the score is raised with 10 points. The clown jumps to a random place and the speed is increased with a small amount.
* **Sounds** - We will use two sounds in this game. A bounce sound that is used when the clown hits a wall, and a click sound that is used when the player manages to click with the mouse on the clown.
* **Controls** - The only control the player has is the mouse. Clicking with the left mouse button on the clown will catch it.
* **Game Play** - At the start of the game the score is set to 0. The room with the moving clown is shown. The game immediately begins. When the player presses the key the game ends.
* **Levels** - There is just one level. The difficulty of the game increases because the speed of the clown increases after each successful catch.

That should be good enough for the moment, and we can now start creating the game!

##### © Copyright YoYo Games Ltd. 2015 All Rights Reserved

# Tutorial

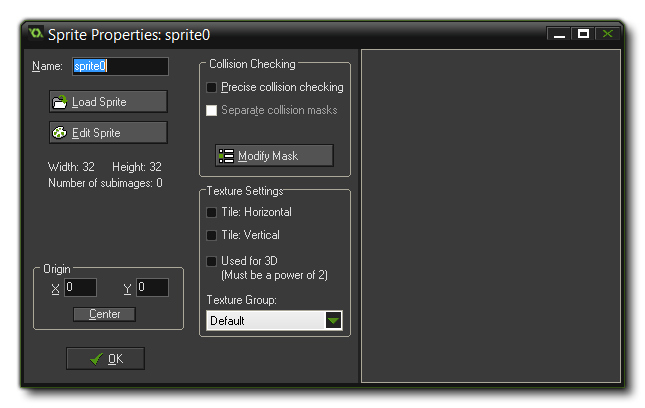
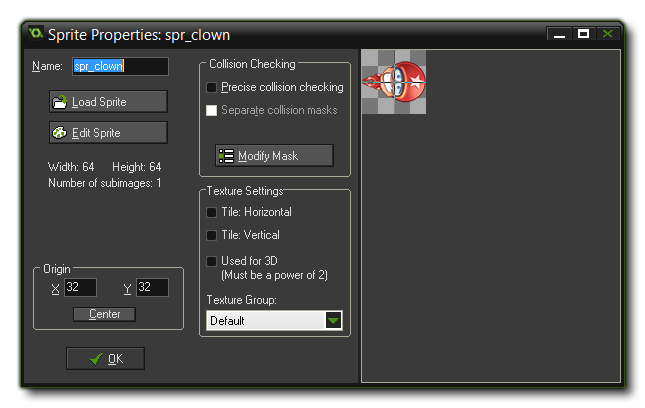
### *Page 3 of 15*

## Sprites

As the game design document describes we will need two images for the two game objects. Such images are called **sprites** in *GameMaker*. There is a lot to know about sprites but, for the moment, simply think of them as little images which we will display in our game.   
  
For making these images you can use any drawing program you like, for example the paint program that is part of any Windows system, but *GameMaker* also has a built-in sprite editor for this purpose. Creating nice-looking sprites is an art that requires a lot of practice, but fortunately there are large collections of images of all sorts available for free. By searching the web and you are bound to find images in large quantities, *but make sure you read and follow the licences that accompany them before including them in any of your games*.   
  
For our little game we use the following two sprites, which can be found in the [*Catch The Clown Assets*](file:///C:\Users\Jyothi\AppData\Local\GameMaker-Studio\Tutorials\My_First_Game\Catch%20The%20Clown%20Assets\) folder that comes with this tutorial:

The clown: The wall: 

To add these sprites to the game we proceed as follows:

1. From the Resources drop down menu at the top of the main *GameMaker* window, choose **Create Sprite**. The **Sprite Properties** form appears, like the one shown below.   
     
   
2. Click on the Name field where currently is says **sprite0** (the default name for the sprite). Rename it to "**spr\_clown**".
3. Click on the **Load Sprite** button. This opens a file browser.
4. Navigate to the *Catch The Clown Assets* folder that came with this tutorial and selected the image file clown.png. The sprite properties should now look like this:   
     
   
5. Press the OK button to close the form.

You should also now set the sprite **origin** to its center. The origin is the point at which *GameMaker: Studio* will draw the sprite in the game room, and all you have to do here is press the button marked *Center* to place the draw origin at the center of the sprite. You should also change the **collision mask** of the sprite to be the full image and a circle as we will want it to bounce later and this mask shape will give the best results. The collision mask is what *GameMaker: Studio* will use to base all the collisions between different instances off of and as such it enables you to give a complex sprite a simple collision shape, as you can see in this image: Next we will add the wall object in the same way:

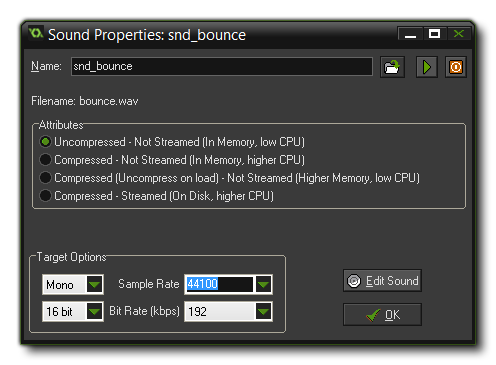
1. From the Resources menu, choose **Create Sprite**. Click on the Name field and rename it to "**spr\_wall**".
2. Click on the **Load Sprite** button and select the image file wall.png.
3. Press the OK button to close the form.

No need to change the origin, nor the collision mask for this sprite as they are fine as they are.   
  
As you might have noticed, the clown and wall sprite have now appeared in the list of resources at the left of the Game Maker window. Here you will always find all the sprites, sounds, objects, rooms, etc... that you have created for your game. Together we call them the *resources*, or *assets*, of the game and they form part of what is known as **Resource Tree**.   
  
You can select a resource from the tree by clicking on its name, and then you can use the **Edit** menu to change the resource, duplicate it, or delete it. *Right-clicking* on the resource name will show a similar menu, and is often quicker to use. Note that this overview of resources will become crucial when you are creating more complicated games.

##### © Copyright YoYo Games Ltd. 2015 All Rights Reserved

**Adding Sounds**

Now that we have created the sprites, we will need to create two sound effects as well. We want one to play when the clown hits a wall and the other to play when the clown is successfully "caught" with the mouse by the player. We will use two wave (\*.wav) format files for this as wave files are excellent for short sound effects.   
  
To create two sound resources:

1. From the Resources menu, choose **Create Sound**. The **Sound Properties** form appears, and you should click on the *Name* field and rename it to "**snd\_bounce**".
2. Click on the **Load Sound** button, navigate to the *Catch The Clown Assets* folder that came with the tutorial, and select the sound file bounce.wav. The form should now look like this:   
     
   
3. Click the "OK" button to close the form.
4. Create another sound resource and name it "snd\_click".
5. Click the **Load Sound** button and select the sound file click.wav.
6. Close the form.

Within the sound properties form you can use the play button, with the green triangle pointing to the right, to listen to the sound which is repeated constantly unless you press the red stop button. There are also a few other options relating to sound quality, but for now we can ignore these as the default values are fine for most games. Notice again that the two sounds are shown in the resource tree on the left of the main *GameMaker* window

# Tutorial

### *Page 8 of 15*

## The Collision Event

The next event we will define is a **Collision Event** with the wall object. The collision event is triggered once for every step of the game that two instances are touching, and an instance is considered as touching when a part of its *collision mask* (as defined in the **Sprite Properties** overlaps that of the other instance in a collision. In this case we want the clown to bounce against the wall, changing direction and playing the bounce sound effect at the same time.   
  
To do this, follow these steps:

1. Press the **Add Event** button. In the *Event Selector* click on the **Collision Event** button and select "**obj\_wall**". The collision event between "obj\_clown" and "obj\_wall" is now added to the list of events.
2. Include a **Bounce** action by dragging it from the page at the right (from the "*Move*" tab, "*Jump*" section).   
   C:\Users\Jyothi\AppData\Local\GameMaker-Studio\Tutorials\My_First_Game\8_1.png
3. Leave the default values for this action. We are not interested in precise bounces and we want to bounce against solid objects (since we flagged the "obj\_wall" object as *solid* in it's object properties).
4. Press OK to close the action window.
5. Select the tab **"Main1"**, and from it include the **Play Sound** action in the collision event action list.   
   C:\Users\Jyothi\AppData\Local\GameMaker-Studio\Tutorials\My_First_Game\8_2.png
6. Drag the **Play Sound** action below the **Bounce** action already present. In the action window, click on the icon to the right of the Sound property which will bring up a list of all available sound assets (from the resource tree), and from the list select "*snd\_bounce*". Leave the **Loop** property as false since we want to play the sound only once, not have it looping constantly.
7. Press OK to close the action.

That concludes the collision event with the wall object! It now has two actions to be performed (in the given order) when the collision occurs. If you have made a mistake, you can right-click with the mouse on an action you added and, for example, choose **Delete** to remove the action (or press the <Delete> key on the keyboard). You can also choose **Edit Values** to change the properties of the action (*double-clicking* on the action will do the same), and you can drag them up and down to change the order in which they are executed within the action list for the event.

##### © Copyright YoYo Games Ltd. 2015 All Rights Reserved

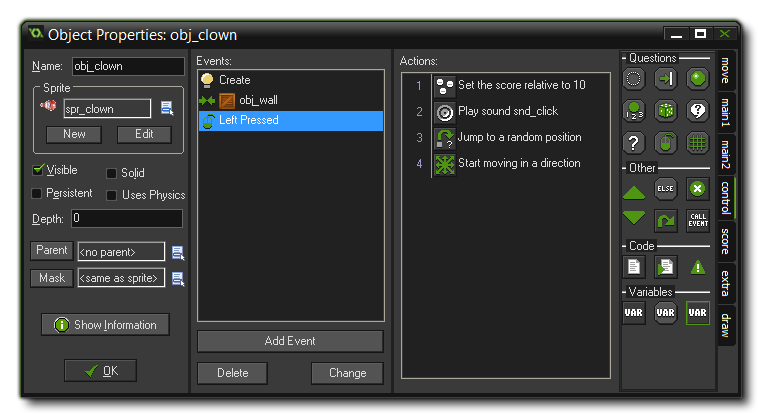
# Tutorial

### *Page 9 of 15*

## The Mouse Event

Finally we need to define what to do when the user clicks with the left mouse button when the cursor is over the clown. We are going to add four actions here...   
  
First of the actions that we will add to the event is to add 10 points to the score. This is easy as *GameMaker* has a built in variable that can automatically keep a score for you. The next action we want to play the click sound, and then, after this, we will jump the clown to a random position, setting a new random direction of motion with a slightly increased speed. Finally, the last two actions are added to gradually increase the difficulty of the game.   
  
The following steps explain how to create the mouse event that we require:

1. Press the **Add Event** button. In the *Event Selector* click on the **Mouse Event** and in the sub-menu that appears select **Left Pressed**. This event happens when the user *presses* the left mouse button while the mouse cursor is on top of the instance and will only be triggered *once* for each press. Note that, like collisions, a normal mouse event will only be triggered when the cursor is considered to be touching the collision mask of the instance, so if the instance has no sprite assigned, this event will not trigger!
2. From the tab labelled **Score** include the *Set Score* action.   
   C:\Users\Jyothi\AppData\Local\GameMaker-Studio\Tutorials\My_First_Game\9_1.png
3. As new score indicate a value of 10 and also click on the box next to the property **Relative** to enable it. When **Relative** is enabled the value is added to the current score, otherwise the score would be replaced by the value.
4. From the tab **Main1** include a **Play Sound**action and for the sound indicate "*snd\_click*". Leave Loop as false.   
   C:\Users\Jyothi\AppData\Local\GameMaker-Studio\Tutorials\My_First_Game\8_2.png
5. From the **Move** tab, include a **Jump to Random** action, which places the instance in a random, collision-free position. The parameters can be left unchanged for this action.   
   C:\Users\Jyothi\AppData\Local\GameMaker-Studio\Tutorials\My_First_Game\9_2.png
6. Finally we include a **Move Fixed** action.   
   C:\Users\Jyothi\AppData\Local\GameMaker-Studio\Tutorials\My_First_Game\7_2.png
7. Again select all eight arrows (and not the centre square) and for the speed indicate a value of 0.5 and enable the **Relative** property to *add* 0.5 to the current speed. If you don't flag the "relative" box, then the speed will be *set* to 0.5 (and if you keep relative ticked and use -0.5, the speed will *decrease* by the given amount).

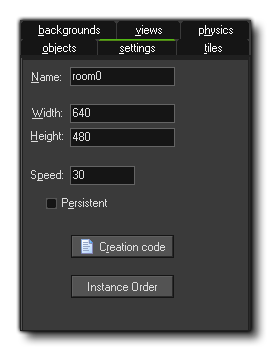
That is all the actions we need for the **Mouse Event** and the finished event list should look like this: We are now finished with the clown object. We have included actions for the three events that are important, so press the OK button to close the Object Properties window and get ready to actually see your game come to life as we create a room and place some instances into it!

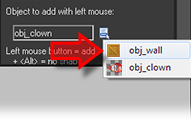
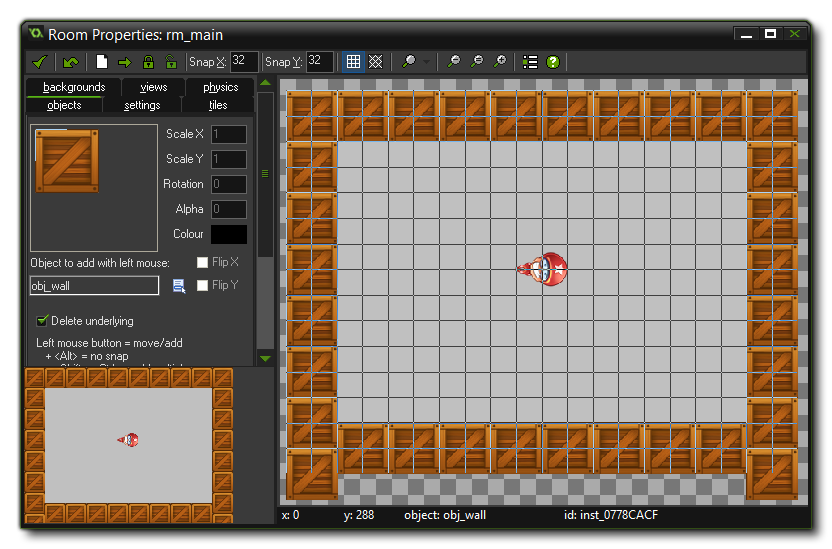
##### © Copyright YoYo Games Ltd. 2015 All Rights Reserved

# Tutorial

### *Page 10 of 15*

## Adding A Room

Now that we have created the game objects there is one more thing to do. We need to create the room in which the game takes place!   
  
For most games, designing effective rooms (which can be considered as generally the "levels" of the game) is a time-consuming task because here we must find the right balance and progression in the game. But for "Catch the Clown" the room is going to be very simple: a walled area with one instance of the clown object inside it, bouncing around.   
  
To start with, from the drop down **Resources** menu in the main *GameMaker* window, choose **Create Room**. This will create a room and open the **Room Editor** window (you can also *right click* on the "Rooms" folder in the resource tree and select *Create Room*). On the left you can see some tabbed pages, and here you should select the tab labelled **Settings**.   
  
  
  
These are the base settings for your room, with the room *width* and *height* (in pixels) and the *room speed* being the important things to note. The room speed is basically the number of times that a game step occurs in a second, so our game will run the step event 30 times each second. We don't need to change these values, so leave them as they are and follow these steps:

1. In the Name field type in "rm\_main". This is the name that will identify this resource in your game, much like the sprite names, sound names, and object names that we have already defined. Note that in programming these names are called **indexes**, so the room name is also the *room index*, as the sprite name, for example, is also known as the *sprite index*. This naming is not important for this tutorial, but as you progress further with *GameMaker* you will see it used more and more to refer to resources.
2. Select the **Objects** tab. Enlarge the window somewhat such that you can see the complete room (you can use the mouse wheel to zoom the game room in and out too).
3. At the top of the room editor, make sure the value for **Snap X** and **Snap Y** is 32, as the size of our sprites is 32x32 pixels and this makes it easier to place the sprites at the correct locations. These values are not used anywhere else except the Room Editor and are used simply as a guide for placing things.
4. On the left you can see the image of the an object, which is the currently selected object. it should be the clown, but if it is not, then click the image and select the "*obj\_clown*" object. Now, place one instance of it in the room by clicking with the mouse somewhere in the centre of the grey area. If you place more than one by mistake, you can *right-click* on it and select *Delete* to remove it again.
5. Click on the icon with the menu symbol next to the field "*obj\_clown*". Here you can also select which object to add, and you should now select "*obj\_wall*".   
     
   
6. Click on the different "cells" bordering the room to put instances there. To speed this up, you can press and hold the <Shift> and <Control> keys on the keyboard and drag the mouse with the mouse button pressed (it is recommended that the option **Delete Underlying** is switched on for this so as to avoid placing more than one instance in the same position). As mentioned previously, you can remove instances using the *right* mouse button and selecting **Delete** from the subsequent pop-up menu, or by holding down the <Control> key while right clicking.
7. Once you have placed the walls as shown in the image below, change objects and place an instance of the clown in the middle of the room.   
     
   

When you are happy with the results, you should click on the green tick at the top left of the window to close and save your room.

##### © Copyright YoYo Games Ltd. 2015 All Rights Reserved

# Tutorial

### *Page 11 of 15*

## Saving and Testing

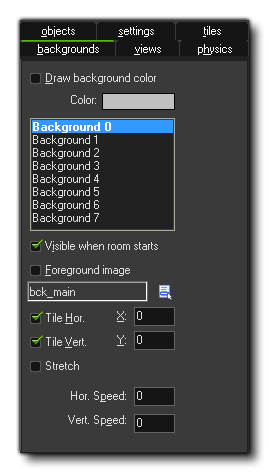
You might not have realized it but our game is ready now to play! The sprites and sounds have been added, the game objects have been designed and the first (and only!) room in which the game takes place has been created. Now it is time to save the game and to test it.   
  
Saving the games works as in almost any other Windows program - just click the **Save** icon and *GameMaker* will save your project. You can also go to the **File** menu and choose *Save As* to rename your game and save it in a different folder. Also note that *GameMaker: Studio* will automatically save your game *every time you press play*, so if you have made changes and are not sure whether they are going to be kept after testing or not, it may be a good idea to save the project under a different name before making the changes and testing.   
  
Next we need to test the game. Testing is a crucial component of creating a game, and although you can test it yourself, you should also ask others to test it when you are nearing completion of the project. Testing (or running the game in general) is simple! Choose the command **Run Normally** from the drop down **Run** menu in the main *GameMaker* window, or press the green **Play** button at the top of the IDE. The game will be saved, and then run, and if you did not make any mistakes, the room will appear on the screen with the clown moving inside it.   
  
  
  
Try testing it now and see whether the game behaves as expected. You should hear the correct sounds and the speed of the clown should increase every time you click on it. To end the game click on the "close" button at the top right of the game window.   
  
Now it is time to fine tune the game. You should ask yourself, for example, the following questions: Is the initial speed correct? Is the increase in speed correct? Is the room size correct? Did we pick effective sprites and sounds for the game? If you are not happy, change these aspects in the game and test again. Remember that you should also let somebody else test the game, because since you designed the game it might be easier for you than for other people.   
  
Once you are happy with your game you should create a stand-alone executable for the game. This is a version of the game that can run without the need for *GameMaker*. This is very simple to do, but first you must choose a "target" for your game from the drop-down menu at the top of t he *GameMaker* window. I suggest choosing "Windows" just now, as that is what this game is designed for. In the **File Menu** choose the command **Create Application**. You have to indicate the place to save the installer too, as well as give the file a name. Once you have done this, you'll see the compile window showing you the progress and when it reads "Finished!" you're all set. You can now close *GameMaker* and still play the game by installing it on your machine, or distribute it for friends to play.

##### © Copyright YoYo Games Ltd. 2015 All Rights Reserved

# Tutorial

### *Page 12 of 15*

## Final Polish - Backgrounds

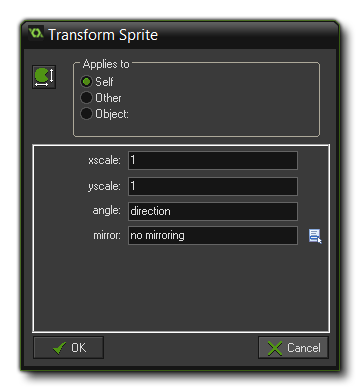
Our first game is ready but it needs some finishing touches to make it a bit nicer. For example, some music, a score display, a nicer background... These are all things that will improve the game and make it a nicer experience for those that play. To start with, let's add a background...   
  
The grey background of the room is rather boring, and so we are going to use a new type of resource, the **Background** resource. To add one, go to the **Resources Menu** and choose **Create Background**. The **Background Properties** window will appear, and you should click on the "*Name*" field and rename it to "*bck\_main*". Now, click on the **Load Background** button, navigate to the *Catch the Clown Assets* folder and select the image file background.png.   
  
We need to assign this background image to the room, so double click on the game room in the resource tree to open it up, then click on the "*Backgrounds*" tab. You then need to de-select the property **Draw Background Colour** (as we do not need to) then click on the little menu icon in the middle and pick the "*bck\_main*" in the pop-up menu.   
  
As you will see, in the room we suddenly have a nice background. Note the two properties **Tile Hor.** and **Tile Vert.** that are available on this tab. They indicate that the background must be tiled horizontally and vertically, that is, repeated to fill the whole room. For this to work correctly the background image must be made such that it nicely fits against itself without showing cracks.   
  
The room editor tab should now look like this:   
  
  
  
Play the game again and you should notice that this simple change gives a big improvement to the final "look" of your game!

##### © Copyright YoYo Games Ltd. 2015 All Rights Reserved

## Final Polish - The Clown

Even though the clown speeds up every click, it is still quite easy to click on it as the direction it moves in is always a straight line. To make it more difficult we will let the clown change its direction of motion from time to time. To this end we are going to use an **Alarm**. Each instance can have multiple alarms (up to 12, numbered from 0 to 11) and these alarms tick down every step (game tick) and at the moment they reach 0 the associated **Alarm Event** happens.   
  
In the Create Event of the clown we will set the alarm, and in the alarm event we will change the direction of motion and set the alarm again. To do this, follow these steps:

1. Re-open the clown object by double clicking on it in the resource list at the left of the window.
2. Select the **Create** event and from the **Main2** tab add a *Set Alarm* action.   
   C:\Users\Jyothi\AppData\Local\GameMaker-Studio\Tutorials\My_First_Game\13_1.png
3. In a game, time is split into *steps* and the steps are defined by the *room speed*, which is the number of steps that the game has to complete per second. The default is 30, so the game completes 30 steps in every second of real time. The alarm is calculated in steps too and we want it to run every two seconds, so there we should have a value of 60 (two times the current room speed). The alarm number should be left as 0, as it is the **Alarm[0] Event** that we are wanting to trigger.
4. Click on **Add Event** and choose the button **Alarm**, then from the pop-up menu select *Alarm 0*. In the event include the **Move Fixed** action (from the move tab), select all eight arrows and set the speed to 0 *but*, unlike previous uses of this action, enable the **Relative** property. In this way 0 is added to the speed, that is, it does not change.
5. To set the alarm clock again, include another **Set Alarm** action, and set it to 60 steps for *Alarm[0]*, exactly as before. This will cause a *loop* in the alarm and it will now run every 2 seconds of your game.

Test your game again now and you'll find that it presents much more of a challenge than previously!   
  
We can also make the clown face the direction that it is moving in. As you can see, the sprite is currently facing to the right, and in *GameMaker: Studio*, this represents 0 degrees. So, what we want to do is add in an action to change the angle the sprite is drawn at every time it changes direction. So, we need to go to **Main1** tab and select the **Transform Sprite** action: C:\Users\Jyothi\AppData\Local\GameMaker-Studio\Tutorials\My_First_Game\13_2.pngIn this action, you can scale the sprite, rotate the sprite and also mirror the sprite around the horizontal or vertical axis, but for now we just need to use the *angle*. In the space provided write the word "**direction**". This is a special variable (like the "score" variable, as we will see) that *GameMaker: Studio* uses to get the angle at which an instance is moving, and placing it here will now make the sprite angle rotate to face the direction of movement. Go ahead and place copies of this new action in each event of the clown object, after the actions that change it's direction (the **Create Event**, the **Collision Event** and the **Alarm[0] Event**), then test your game again to see how this changes things.

##### © Copyright YoYo Games Ltd. 2015 All Rights Reserved

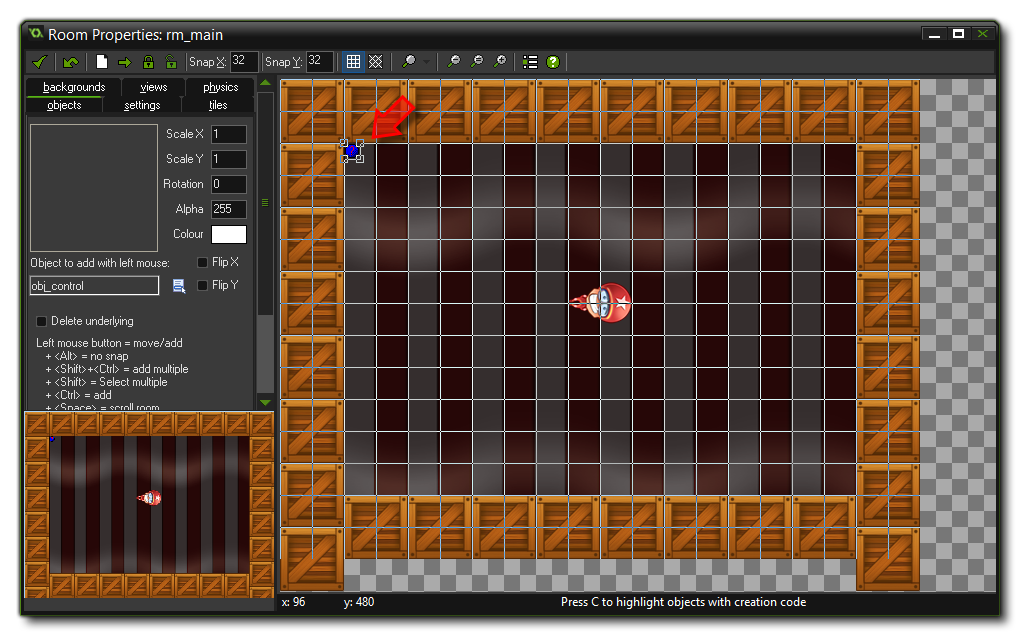
# Tutorial

### *Page 14 of 15*

## Final Polish - Control

The final thing we are going to add to the game is a *controller object*. Many games have special objects with no sprites assigned to them that are used to control elements of the game that are not directly related to the gameplay. In this case, our controller will show the score and play some background music.   
  
First lets add some music to the resources for the game:

1. From the **Resources** drop down menu, choose **Create Sound**, which will open a new **Sound Properties** window. Click on the *Name* field and rename it to "*snd\_music*".
2. Click on the **Load Sound** button, navigate to the Resources folder and select the sound file music.mp3.
3. We don't need to change anything else here as *GameMaker* will detect that it is an \*.mp3 file and make the necessary changes. Note that this sound file uses the \*.mp3 format as it keeps the file size smaller, and this follows the general rule of thumb that \*.wav files are for sound effects and \*.mp3 files are for music. Now press the "OK" button to close the window.

Create a new object by opening the **Resources** drop down menu and selecting **Create Object**, or by *right clicking* on the "Objects" folder in the resource tree and selecting "*New Object*. Give this object the name "*obj\_control*" and add a **Create Event** to it. In this event, drag a **Play Sound** action into it and for a sound indicate "snd\_music" and set **Loop** to true because we want the music to repeat itself forever while playing.   
  
We want to display the player score with this object too, so you need to add a **Draw Event** to your object. In this **Draw Event** you should go to the **Score** tab and drag the **Draw Score** action into the action list.   
C:\Users\Jyothi\AppData\Local\GameMaker-Studio\Tutorials\My_First_Game\14_1.png  
The position for drawing the score is defined by the x and y values that you input here, so set them both to 64, and leave the "Caption" as it is, since it is the score we are showing and not some other value.   
  
The last thing we need to do is to add this object into the room, so double click the game room to open it again a,d then in the **Objects** tab, select "*obj\_control*" and place it in the room anywhere. Since the object does not have a sprite, you will see that *GameMaker* shows you it's position with a little blue ball marker with a question mark. this will not be shown in the game and is only there as a reminder to you that you have placed a spriteless instance in the room.   
  
  
Run the game now and see how everything works together!

##### © Copyright YoYo Games Ltd. 2015 All Rights Reserved

# Tutorial

### *Page 15 of 15*

## Summary

Your first game is now complete! it may not seem like much, but thanks to this short tutorial you have learned the basics of creating games with *GameMaker: Studio*... You now know about resources and how to create them. You know how to create objects and the difference between them and instances. You know how to make a room, and add instances into it. These things are the building blocks on which you will construct better games in the future!   
  
However, before going off to start work on your first real masterpiece, it is strongly recommended that you run through at least a couple of the other tutorials that come with *GameMaker*, as each one introduces different aspects and techniques, all of which you will need to know to get the most out of the software. It is also recommended that you have the manual open at all times as it provides a valuable source of reference material for all aspects of *GameMaker*.   
  
With that said, this first tutorial is now complete and we hope that you have had fun making your first game!

##### © Copyright YoYo Games Ltd. 2015 All Rights Reserved

.

# Tutorial

### *Page 5 of 15*

## Objects, Events and Actions

Having created the sprites and sounds does not mean that anything is going to happen in your game. Sprites are only the images for game *objects* and we have not yet defined any game objects. Similarly, sounds will only play if we tell them to be played, and we can only tell them to be played from a game object too. So we need to create our two game objects next...

## Instances

However, before we do this you will have to understand the basic way in which *GameMaker: Studio* operates. As we have indicated before, in a game we have a number of different game **objects** and during the running of the game one or more **instances** of these game objects will be present on the screen or, more general, in the game world. Note that there can be *multiple instances of the same game object*. So for example, in our "Catch the Clown" game there will be one wall object, but a large number of instances of that object surrounding the playing field, and there will be just one instance of the clown object. Think of objects as a blueprint, and instances as something created from that blueprint.

## Events

Instances of game objects don’t do anything unless you tell them how to act, and you do this by indicating how the instances of the object must react to *events* that happen. There are many different events that can happen as your game progresses, but the first (and often most important) event that we are going to look at is when the instance is created. This event is called the **Create Event**, and more often than not some action is required here, for example we must tell the instance of the clown object that it should start moving in a particular direction.   
  
Another important event happens when two instances collide with each other and is called a **Collision Event**. For example, when the instance of the clown collides with an instance of the wall, the clown must react and change its direction of motion.   
  
Other events happen when the player presses a key on the keyboard or clicks with mouse on an instance etc... For the clown in our game we will use a **Mouse Event** to make it react to a press of the mouse on it.

## Actions

To indicate what must happen in any given event, you must specify *D'n'D actions*. There are many useful actions for you to choose from, for example there in an action that sets the instance in motion in a particular direction, or there is an action to change the score, or an action to play sounds.   
So, to sum up the above, defining a game object that we can create instances of consists of:

* Giving the object a name and assigning a sprite (if necessary)
* Setting the initial object properties
* Adding events which will be triggered by specific things in our game
* Adding actions to the events to have our instances of the object act/react in different ways

Before continuing, just lets revise a very important distinction that you should always keep in mind, and that is the distinction between **objects** and **instances** of those objects. An object defines a particular game object with its behaviour (that is, its reaction to events). Of this object there can be one or more instances in the game and these instances will act according to the defined behaviour. Stated differently, an object is an abstract thing - like in normal life, we can talk about a chair as an abstract object that you can sit on, but we can also talk about a particular chair, that is an instance of the chair object, which actually exists in our home.

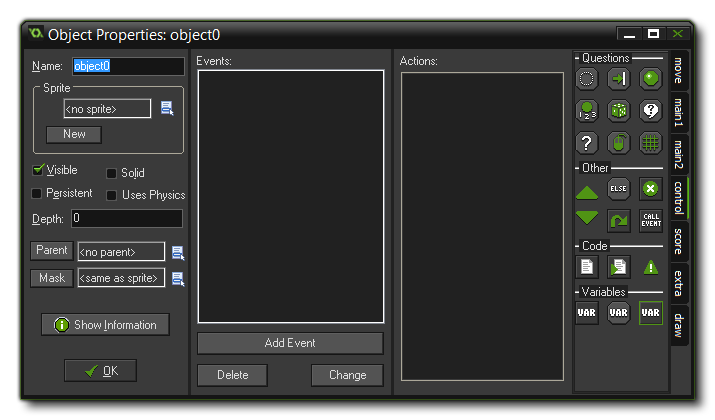
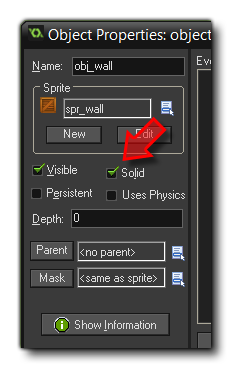
##### © Copyright YoYo Games Ltd. 2015 All Rights Reserved

# Tutorial

### *Page 6 of 15*

## Adding Objects

So how does this all this come together for the game we are making? To start with, we will need two objects. Let's first create the very simple wall object, as this object needs no behaviour at all since it will not react to any events in the game world.   
  
To create the wall object you must follow these steps:

1. From the drop down *Resources* menu in the main *GameMaker* window choose **Create Object**. The Object Properties form appears, as is shown below:   
     
   
2. Click on the *Name* field and rename the object to "**obj\_wall**".
3. Click on the *Menu icon* at the end of the *Sprite* field and in the list of available sprites select the "**spr\_wall**" sprite.
4. Instances of the wall object must be "flagged" as solid, that is, no other instances should be allowed to penetrate them. To this end click on the box next to the **Solid** property to enable it.   
     
   
5. Press OK to close the form.

For the clown object we start in the same way:

1. From the *Resources* menu, choose **Create Object**.
2. Click on the *Name* field and rename the object to "**obj\_clown**".
3. Click on the icon at the end of the Sprite field and select the "**spr\_clown**" sprite.

Note that we do not make the clown object solid! Moving instances should *never* be flagged as solid in their object properties as that can cause some serious issues with collisions, so reserve this option for wall objects, platform objects and other things that shouldn't move around the room when added as instances. Now, for the clown there is a lot more that needs to be done as we have to specify its behaviour, which we will do on the next page...

##### © Copyright YoYo Games Ltd. 2015 All Rights Reserved

## The Create Event

Appart from the object details, the rest of the **Object Properties** window is taken up by two sub-windows: one for adding events, and one for adding actions to those events. With the buttons below the Events window you can add events, delete events or change events, of which there are a large number of different ones you can use.   
  
Next to the events there is an empty list of actions that must be performed for the selected event (if any), and at the right of this list that are a number of tabbed pages with little icons. These icons represent the different actions and in total there are close to 100 different actions you can choose from. If you hold your mouse above one of the icons a short description of the corresponding action is given in a tooltip. You can drag actions from the tabbed pages at the right to the action list to make them happen when the event occurs. This is called the **Drag and Drop** system of programming (D'n'D for short).   
  
So, to get things started with programming our game, we are now going to define what should happen when an instance of the clown object is first created. In this case we want the clown to start moving in an arbitrary direction at a fixed speed so proceed with the following:

1. Press the **Add Event** button. The *Event Selector*, as shown below, will appear.   
     
   
2. Click on the **Create Event** button. The Create event is now added to the list of events, and it is automatically selected (you should see it highlighted in the IDE).
3. Next you need to include a **Move Fixed** action in the list of actions. To this end, open the *Move* tab and then press and hold the mouse on the action image with the eight green arrows in the page at the right, drag it to the empty actions list, and release the mouse to drop it into the action list for the event. An action window will now open asking for information about the action.   
   C:\Users\Jyothi\AppData\Local\GameMaker-Studio\Tutorials\My_First_Game\7_2.png
4. In the action window for the **Move Fixed** action you can indicate in which direction the instance should start moving. Select all eight directions (not the middle one which corresponds to *no* motion), and note that the selected directions turn red. When multiple directions are selected one is chosen at random when the event is run. Also set the *Speed* to 4. The "speed" of an instance is calculated as "pixels per step", so 4 means that the instance will move 4 pixels in the given direction every game step.   
     
   
5. Press OK to indicate that we are ready with this action.

##### You have now specified behaviour that must be executed when an instance of the clown object is created, by adding the event, including an action, and setting the action properties. The object properties window for the clown object should now look like this: C:\Users\Jyothi\AppData\Local\GameMaker-Studio\Tutorials\My_First_Game\7_4.png Copyright YoYo Games Ltd. 2015 All Rights Reserved

##### ©

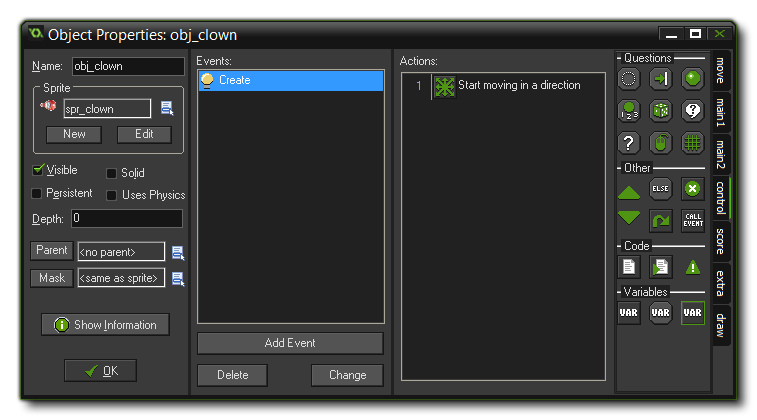
# Tutorial

### *Page 7 of 15*

## The Create Event

Appart from the object details, the rest of the **Object Properties** window is taken up by two sub-windows: one for adding events, and one for adding actions to those events. With the buttons below the Events window you can add events, delete events or change events, of which there are a large number of different ones you can use.   
  
Next to the events there is an empty list of actions that must be performed for the selected event (if any), and at the right of this list that are a number of tabbed pages with little icons. These icons represent the different actions and in total there are close to 100 different actions you can choose from. If you hold your mouse above one of the icons a short description of the corresponding action is given in a tooltip. You can drag actions from the tabbed pages at the right to the action list to make them happen when the event occurs. This is called the **Drag and Drop** system of programming (D'n'D for short).   
  
So, to get things started with programming our game, we are now going to define what should happen when an instance of the clown object is first created. In this case we want the clown to start moving in an arbitrary direction at a fixed speed so proceed with the following:

1. Press the **Add Event** button. The *Event Selector*, as shown below, will appear.   
     
   
2. Click on the **Create Event** button. The Create event is now added to the list of events, and it is automatically selected (you should see it highlighted in the IDE).
3. Next you need to include a **Move Fixed** action in the list of actions. To this end, open the *Move* tab and then press and hold the mouse on the action image with the eight green arrows in the page at the right, drag it to the empty actions list, and release the mouse to drop it into the action list for the event. An action window will now open asking for information about the action.   
   C:\Users\Jyothi\AppData\Local\GameMaker-Studio\Tutorials\My_First_Game\7_2.png
4. In the action window for the **Move Fixed** action you can indicate in which direction the instance should start moving. Select all eight directions (not the middle one which corresponds to *no* motion), and note that the selected directions turn red. When multiple directions are selected one is chosen at random when the event is run. Also set the *Speed* to 4. The "speed" of an instance is calculated as "pixels per step", so 4 means that the instance will move 4 pixels in the given direction every game step.   
     
   
5. Press OK to indicate that we are ready with this action.

You have now specified behaviour that must be executed when an instance of the clown object is created, by adding the event, including an action, and setting the action properties. The object properties window for the clown object should now look like this:   
  


##### © Copyright YoYo Games Ltd. 2015 All Rights Reserved