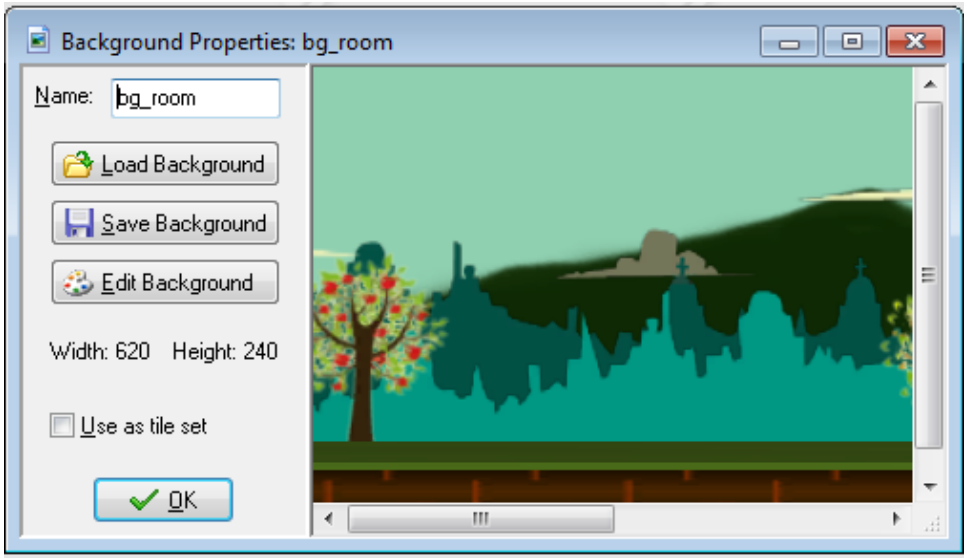
SCROLLING BACKGROUNDS

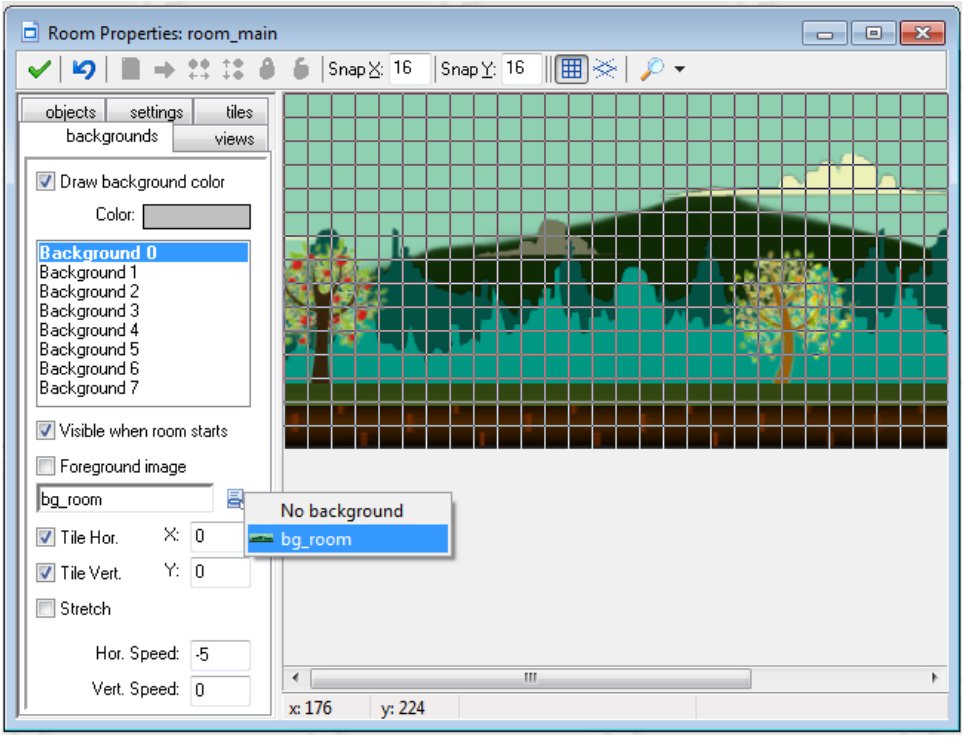
Scrolling backgrounds can often be useful when creating platform games (something we will be developing over the course of this unit).

Creating a scrolling background in GameMaker is actually very simple. The first thing you will need to do is find a background image that you want to use as your background. Doing a search for "scrolling background images" will give you numerous examples of scrolling backgrounds that you can use for a game. For this lesson, I will be using the following background image that you can find on the shared directory (the image is called **parallax.png**):

Once you have found an image you want to use, you will need to create a **Background** in your GameMaker project:



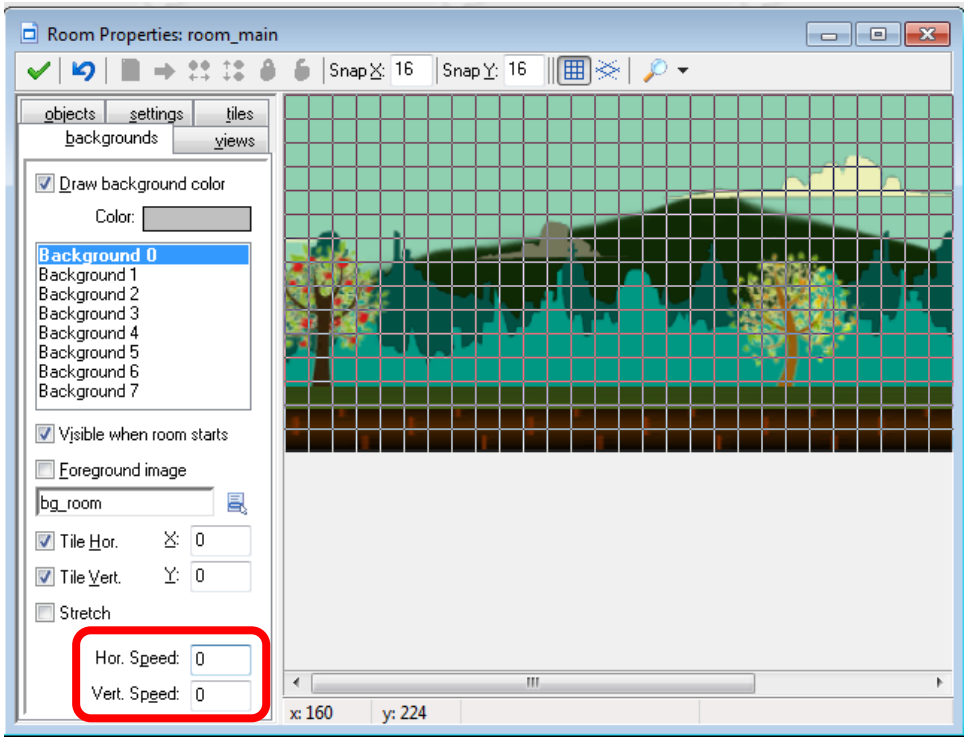
The next thing you will need to do is create a Room and add the background to your room:



Finally, you will need to determine whether you want the background image to scroll vertically or horizontally.

Scrolling horizontally means it will scroll left or right. If you want it scrolling right, you will need to set the **Horizontal Speed** to a positive integer value (the higher the number, the faster it will scroll). If you want it scrolling left, you will need to set the **Horizontal Speed** to a negative integer value (the lower the number, the faster it will scroll).

Scrolling vertically means it will scroll up or down. If you want the background scrolling up, you will need to set the **Vertical Speed** to a negative integer value (the smaller the number, the faster it will scroll). If you want it scrolling down, you will need to set the Vertical Speed to a positive integer value (the higher the number, the faster it will scroll).



Once you have established whether you want it scrolling horizontally or vertically and you have set the horizontal or vertical speed, save and run the program.

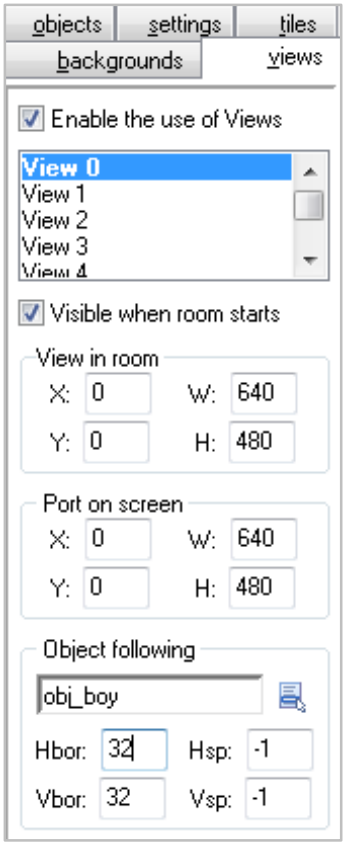
USING VIEWS

Views provides a mechanism for drawing different parts of your room at different places on the screen, or for drawing just a part of your room to cover the whole screen. For example, in most platform games, the view follows the main character, as if you could see the whole level on the screen your character would be too small to see and there would be no surprises for the player.

Views can also be used in two-player games, since they allow you to create a split-screen setup in which in one part of the screen you see one player and in another part you see the other player.

A third use is in games in which part of the room should scroll (e.g. with the main character) while another part is fixed (for example some status panel). This can all be easily achieved in using the Views menu.

THE VIEWS TAB



At the top of the views tab there is a box labelled **Enable the use of Views**, and this must be flagged before any of the views can be active in a game.

Below this you will see a list of available views (you can define a maximum of eight) with the information about where these views are to be drawn contained in the boxes below. First of all you must indicate whether the view should be visible when the room starts by checking the appropriate box, and if you wish to use views, make sure at least one view is visible at the start of every room. Visible views are shown in bold in the view list above this.

A view is defined by two different sets of values: the view itself and the port on the screen. This can sometimes cause confusion so let me explain this a bit before explaining how we define each of them.

The **View in room** is an area of the room that we are going to display on the screen.

The **Port on screen** is the area of the display where we are going to draw the view.

So, this means that you can have a 640 x 480 view in your room, and then set the port to 320 x 240, which will display the view scaled down to that sized port on the screen, and you can also do the same and set the view to a smaller value and the port to a larger value making the image scale up to fit the port size and be shown on the screen larger than it is. In this way you can maintain a screen (port) size while changing the view and display more or less of the room in the same area of the screen.

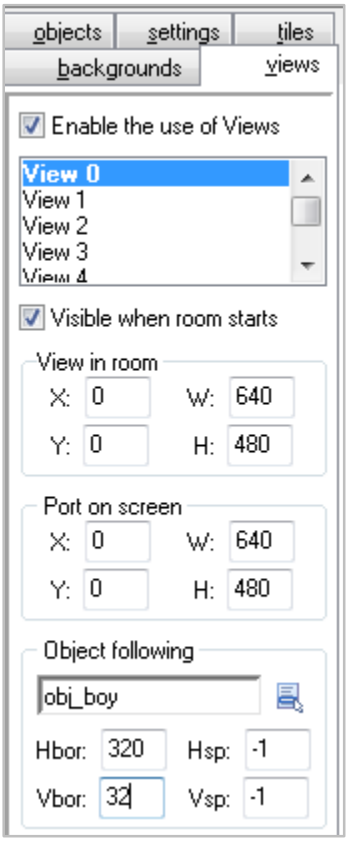
The view is always defined as a rectangular area in the room, where you specify the position of the top-left corner, the width and the height of this area. Then you must specify where this area is shown in the window on the screen by defining the view port, where again you specify the position of the top-left corner and the size (note that anything other than 0, 0 for the top left corner will produce strange results). You can have more than one port and they can overlap, in which case they are drawn in the indicated order one on top of the other. One important thing you need to remember is that the overall screen area is always defined as a rectangular area, so your ports, even when offset, will form a rectangle, with any empty spaces being filled in by the window colour.

The **Object following** option is used when you want the view to "follow" a certain object. To do this you must click on the menu icon and select an object from the list that pops up (if there are multiple instances of this object in the room, only one of them is followed by the view). The normal behaviour for a view is to only move when the instance being followed gets too close to a "buffer" zone that makes an invisible boundary around the edge of the view. This zone can be defined by you using the **Hbor** and **Vbor** values, where **Hbor** is the horizontal border zone, and **Vbor** is the vertical border zone. So, setting these values to, for example, 64, will mean that the view will not start to move and follow the character until he reaches 64 pixels from the edge of the view.

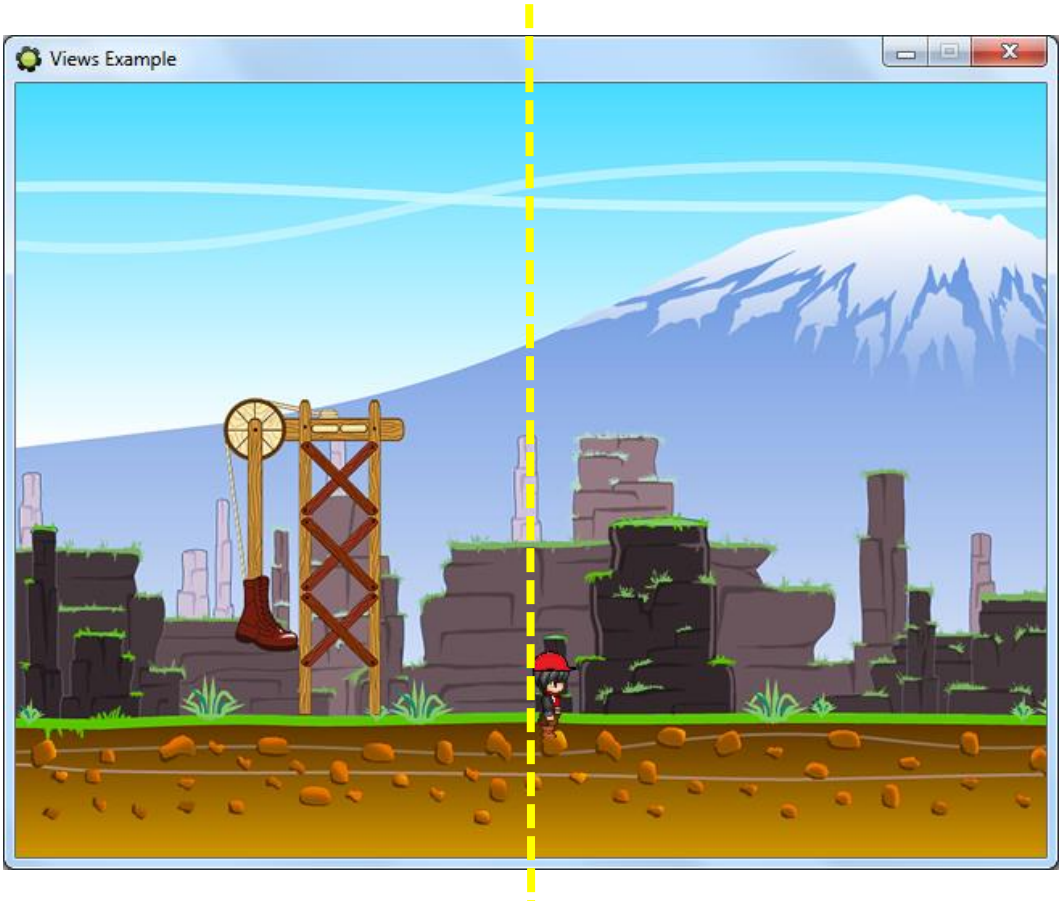
Finally you can indicate the speed at which the view moves when the character has reached the buffer zone, and this has a default value of -1. This default value is basically instantaneous and means that the moment the follow object is outside the **Hbor** or **Vbor** buffer zone, the view will skip to its current position. Now, this is not always what you want and so you can set the vertical and horizontal scrolling

speed for the view by setting the values for **Hsp** and **Vsp** to something other than -1. Note that a value of 0 will cause the view to not move at all, and any other positive value is how many pixels it will move in any step, so setting the **Hsp** to 5 will have the view follow the object at 5 pixels per step horizontally.

In the following example, I'm going to use a background image that is 1920 x 480 and set the Views to the following options:



Now when the player moves to the right, as soon as it reaches the halfway point of the view the background will begin scrolling to the left. This is because the **Hbor** value is set to 320 (which is half of the width of the view in the room).



Hbor = 320