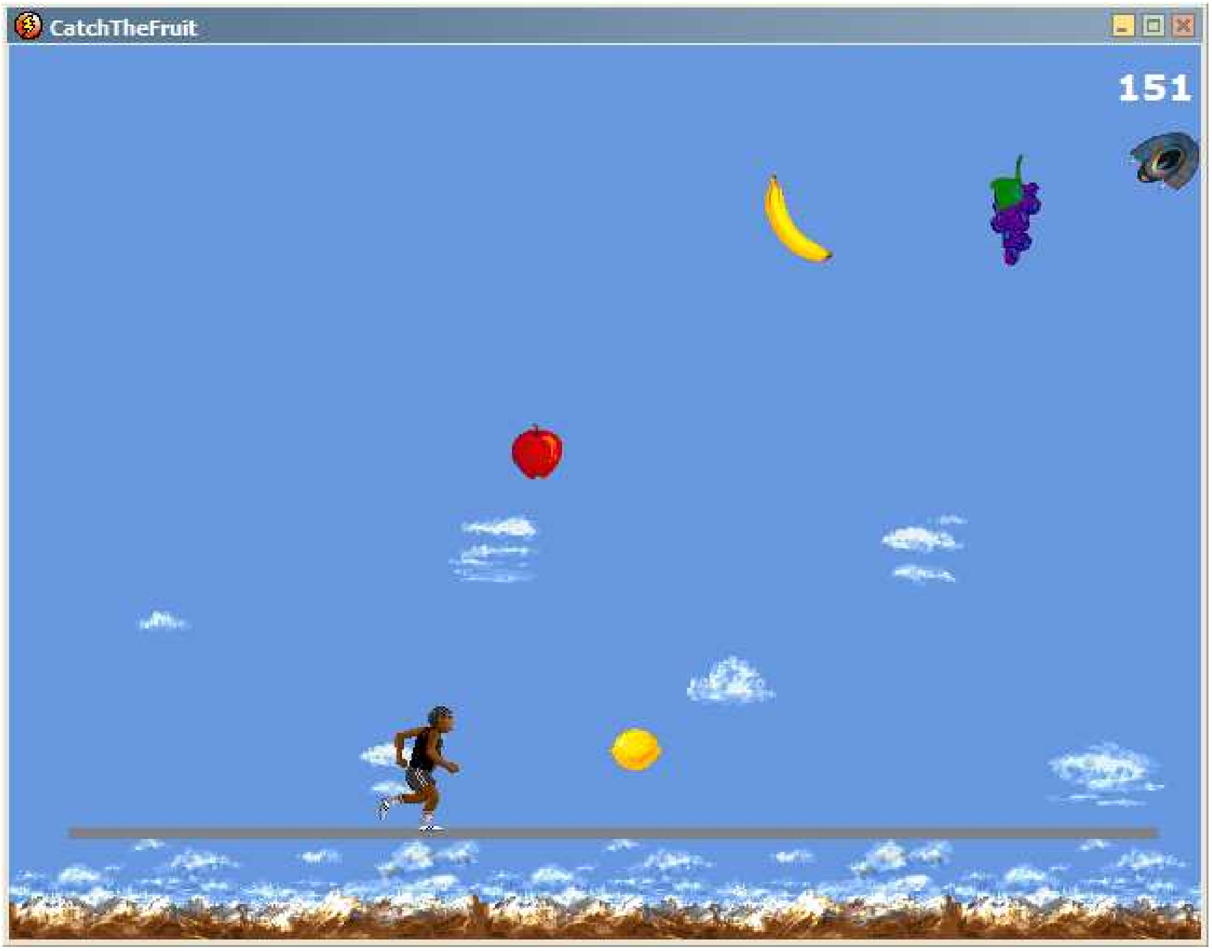
Making a 'Catch the fruit' game with MMF2



Making a ‘Catch the fruit’ game with MMF2

Table of contents

Welcome

This tutorial will guide you through the creation of a simple little game called ‘Catch the fruit’. This tutorial will cover basic game making techniques, collisions, movements, creation of objects and inserting objects from the MMF2 libraries.

“Catch the fruit” is a simple game that can be made in many variations. This version of the game is probably the simplest version of it.

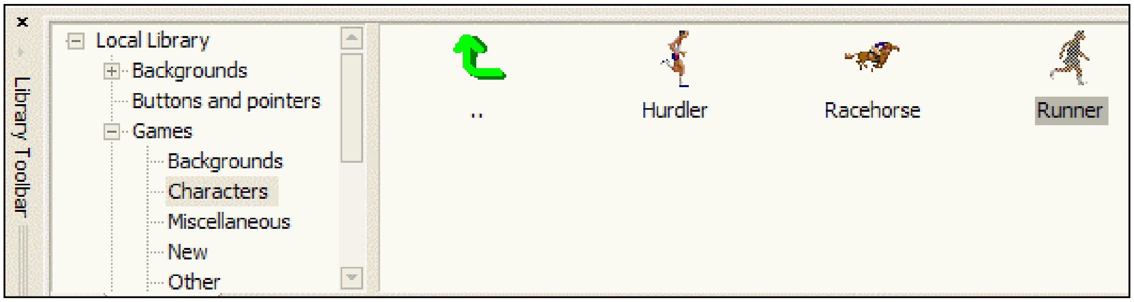
Things we want to see in our game

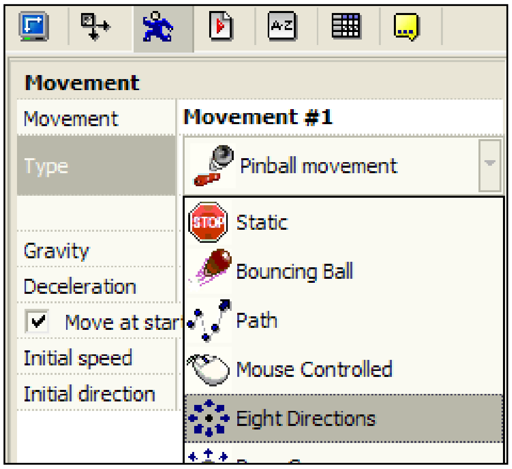
- Making a moveable character
- Occasionally created ‘fruits’ fall down from the top of the screen
- You gain points by collecting the fruit
- You loose points by missing a fruit
- More fruits drop over time to make the game harder

Making a controllable character

First we need to find a suitable character for our game that we want to control. A good place to look is the libraries that come bundled with MMF2.

A good character sprite can be found in the library “Games -> Characters -> 08sptpep” called “Runner”.



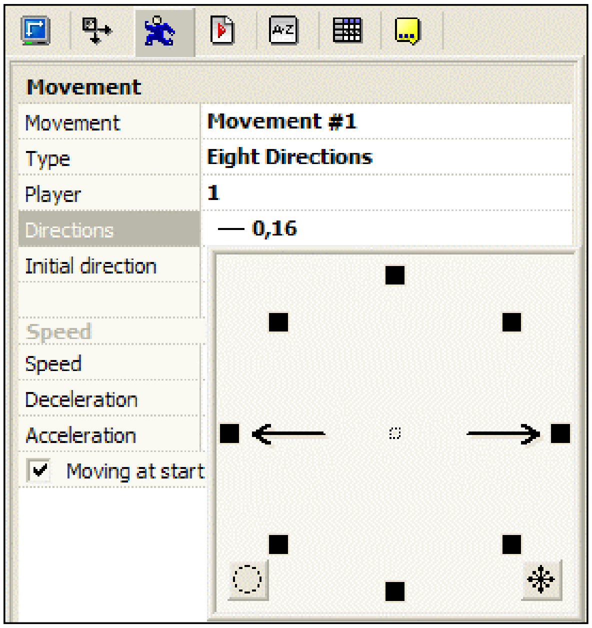


Movement: Movement #1

Type: Pinball movement

Drag the object into the bottom of our frame and click on it to select it.

We want our character to be controlled by the player of our game, so we will go to the ‘Movement’ tab in the properties sidebar. In the “Movement type” property, we select the ‘Eight Directions’ movement type. With this movement we can move our character around in a ‘Zelda’ like way. This isn’t suitable for our game, so we need to restrict the directions the player is allowed to move in.



Movement: Movement #1

Type: Eight Directions

Player: 1

Directions: 0,16

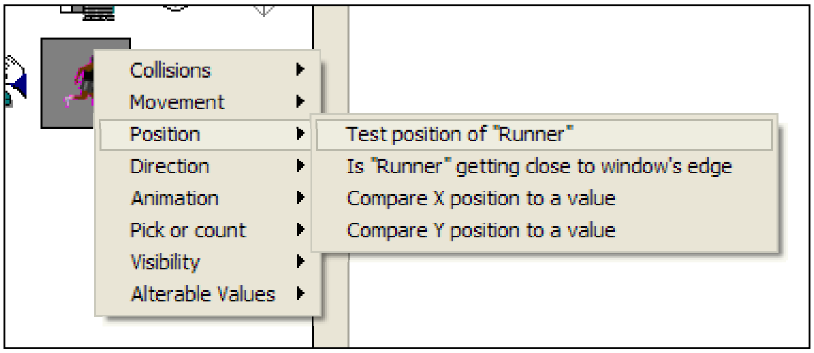
To do that, click the ‘Directions’ property and remove the arrow from all the directions except the arrows turning left and right. Now our character can only move left and right.

Also change the speed of the character to something around 32. We don’t want the game to be too easy do we?

If you open the animation editor for our “Runner”, you will notice that it already have walking animations for both left and right so we don’t need to modify this object’s animations to make it look right. You can test the application to try the controls for our runner sprite.

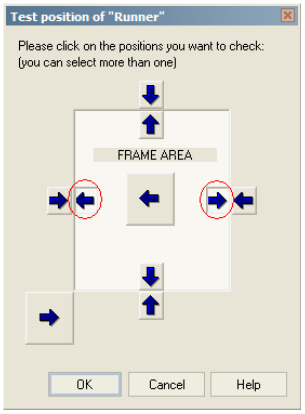
You might notice that our player can run outside the borders of the screen which shouldn’t be possible. So we simply open up the event editor and make an event to prevent it.

Create a new condition from the ‘Runner’ sprite called “Test position of ‘Runner’”:


Collisions
Movement
Position
Direction
Animation
Pick or count
Visibility
Alterable Values
Test position of "Runner"
Is "Runner" getting close to window's edge
Compare X position to a value
Compare Y position to a value

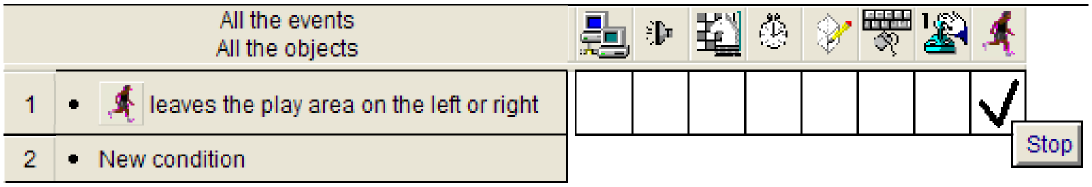
In the dialog that pops up, you simply click arrows that indicate a movement leaving the frame. In our case we click these arrows marked with a red circle:

Here the condition is: “Is our object leaving the frame at the left or right?” We can then simply make a “stop” action to prevent the object from leaving if the condition was true.


Test position of "Runner"

Please click on the positions you want to check: (you can select more than one)

We should now have an event that looks like this:


All the events
All the objects

1 • leaves the play area on the left or right ✓ Stop

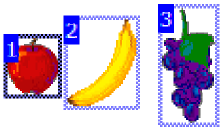
2 • New condition

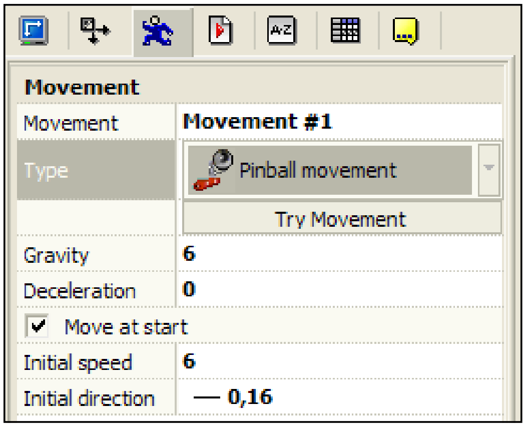
Adding the fruit

For our game we wanted some fruits to drop from the top of the screen once in a while. For this we can use more of the library graphics that comes with MMF2 used together with MMF2's inbuilt movement systems.

We can find some nice fruit sprites from the MMF2 libraries at:
"Games -> Miscellaneous -> 07fruit" (Or "Fruits") Drag them into the frame so we can use them in our game.

Now select all the fruits you have dragged into the frame. It should look like this:





Movement	
Movement	Movement #1
Type	Pinball movement
	Try Movement
Gravity	6
Deceleration	0
☑ Move at start	
Initial speed	6
Initial direction	— 0,16

With the inbuilt pinball movement, we can make our fruits fall down in a "realistic" way. When they are created, they fall slowly but will quickly accelerate like any other object with gravity in real life.

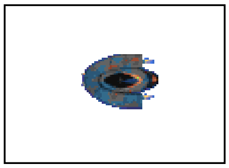
We then apply the pinball movement for all of the fruits by selecting it from the list.

If you test the application, they will drop really fast, making them almost impossible to catch. So either we could make the 'Runner' run faster or we could make the fruits fall slower. For this example we'll try the latter and make the fruits fall slower. We set the gravity and initial speed to 6. You don't need to change the fruit's Initial direction but I have set these fruit's initial direction to right (0) and left (16).

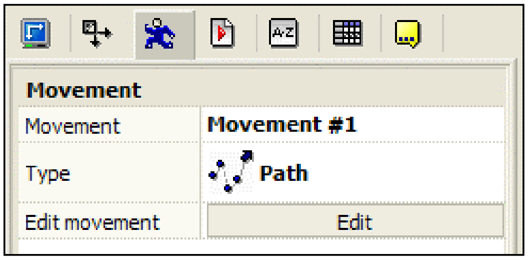
You can now test how the fruits fall down from the sky obeying gravity.

Making the fruits drop occasionally

For this little game, the fruits should drop from some flying object. We'll find a suitable graphic that can fly around in the top of our screen, randomly dropping some delicious fruits.



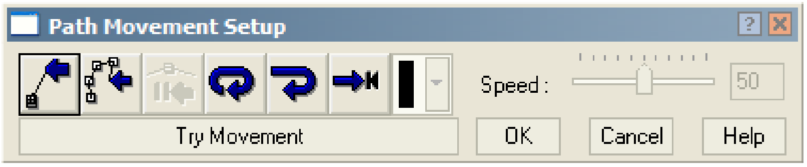
In the graphics library "Games -> Spacecraft -> Spacecraft" there is a sprite called "Small UFO" which we can use. Drop it at the top left corner of our frame. We need this little UFO to move around in some sort of pattern or following a path. For this purpose we can use the built-in "Path Movement".



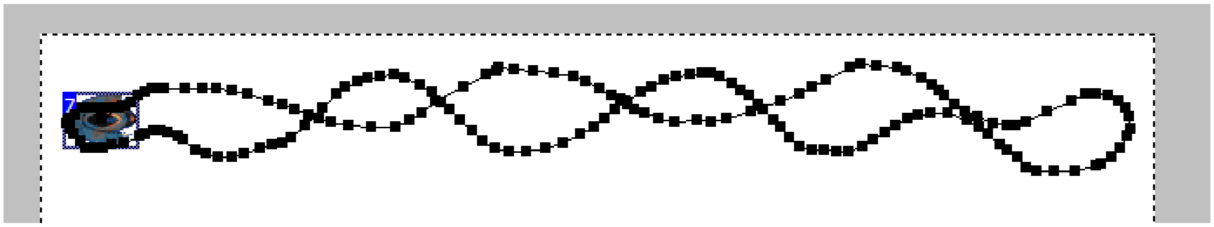
Movement	
Movement	Movement #1
Type	Path
Edit movement	Edit

Select the UFO, go to its movement properties and select 'Path Movement' for it. Then click the 'Edit' button to edit the UFO's path.

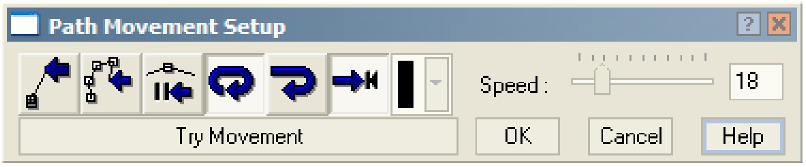
This is the path movement dialog box that will pop up after you clicked 'Edit'. With this we will draw the path we want the UFO to follow.



To begin drawing the path, you click the second button from the left. Now simply draw a curvy path at the top of the frame like if you would draw with a pencil tool in a paint program. Here is an example path.



If you test the movement now, the UFO will quickly go through the path and stop, but that is not what we want. We want it to slowly follow its path and continue in a seamless loop.
To change the speed, drag a selection box around all path nodes so they are selected. On the Path movement dialog box, you move the slider bar to a lower value than 50. Speed 18 is a good value. We also want the movement to repeat itself when it is finished, so we click the “Loop the movement” button. “⟲”
If we didn’t position the last node exactly where we started, the UFO will move slightly each time it finishes its movement. To fix this, we click the “Reposition object at end” button “⟲”to make sure it will never move away from its original position.
We should end up with a dialog box that looks like this:



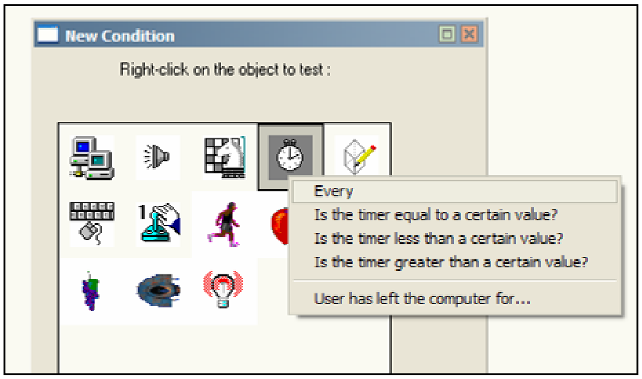
Speed : 18

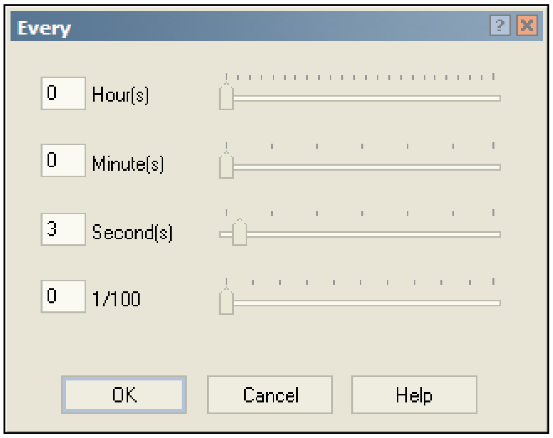
Try out the game and watch our little UFO move around the screen.

Now for the fruits:
MMF2 has simple timing functions which we can use for the fruit creation.
Let’s make some fruits drop more often than other:
Every 3rd second create an apple,
Every 5th second create a banana,
Every 7th second create some grapes.

MMF2 makes it much simpler to do these kinds of timings than other development systems where you constantly need to compare the current system time to some initial value. Here it is only necessary to tell MMF what you want to happen every three seconds and it will magically happen.

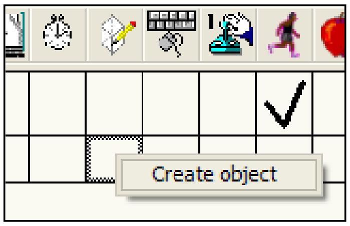
Make a new ‘Every’ condition. You can find it in the Timer object which is one of the default objects that is always included in every MMF2 application.
Change the “Seconds” slider bar to the value 3.


New Condition
Right-click on the object to test :
Every
Is the timer equal to a certain value?
Is the timer less than a certain value?
Is the timer greater than a certain value?
User has left the computer for...

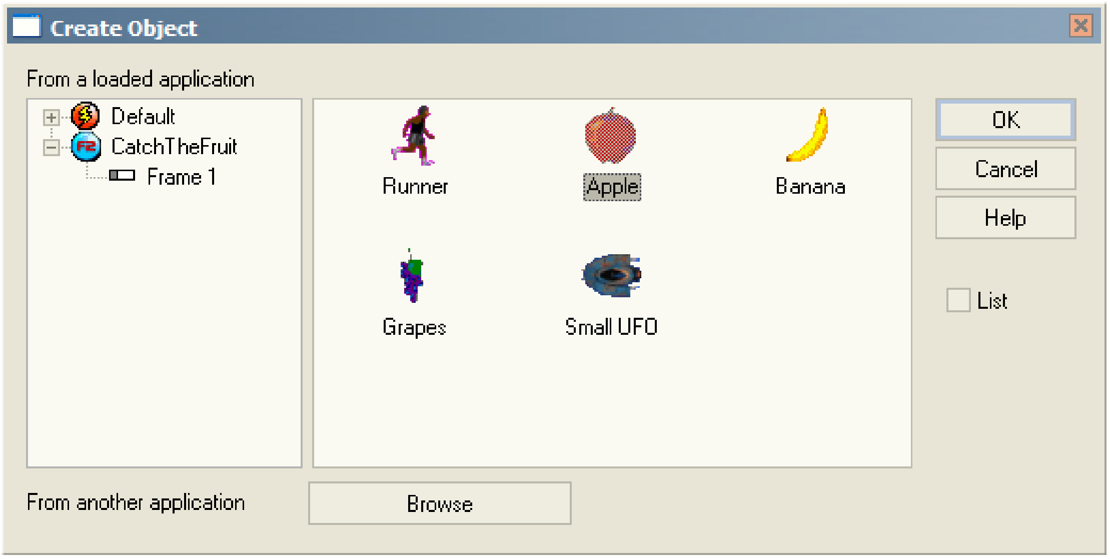

Every

0 Hour(s)
0 Minute(s)
3 Second(s)
0 1/100

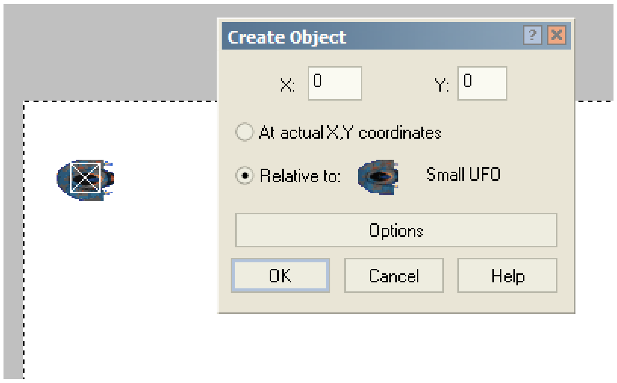
Every action you put into this new event will happen once every three seconds. So if we put a ‘create object’ action into our ‘Every 3 seconds’ event our objects will be created every third second.
At the ‘Create new objects’ object you find in the event editor, you right click and select the ‘Create object’ action.



In this dialog you are asked which object you want to create. Simply select the apple object and click ‘OK’.


Create Object

From a loaded application



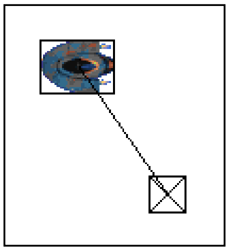
Create Object

X: 0 Y: 0

○ At actual X,Y coordinates

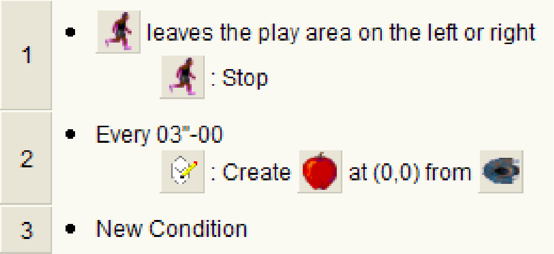
● Relative to: Small UFO

When this “Create Object” dialog opens, you can then choose the coordinates where your object should appear. If you drag around the ‘creation point’ you simply tell MMF to create it at that exact spot, but we want the apple to be created at the UFO, not at action coordinates. So simply click on the UFO object on the frame to create the apple relatively to the UFO.



You will notice that the ‘creation point’ will not overlap the UFO to begin with when you click it. If we leave it there, the apples will then be created at that position relative to the UFO. This would look odd as the apples would pop out of nowhere but close to the UFO. So drag the ‘creation point’ up to the centre of the UFO. Alternatively you can simply write the value ‘0’ into the X and Y boxes. Then it will look just the same as in the first picture. Press ‘OK’.

Test your game and watch how the UFO will drop the apples every three seconds.
To sum up what we have actually done in the event editor, here is a snapshot of our events as seen in the Event List Editor.



1 • leaves the play area on the left or right
: Stop

2 • Every 03"-00
: Create at (0,0) from

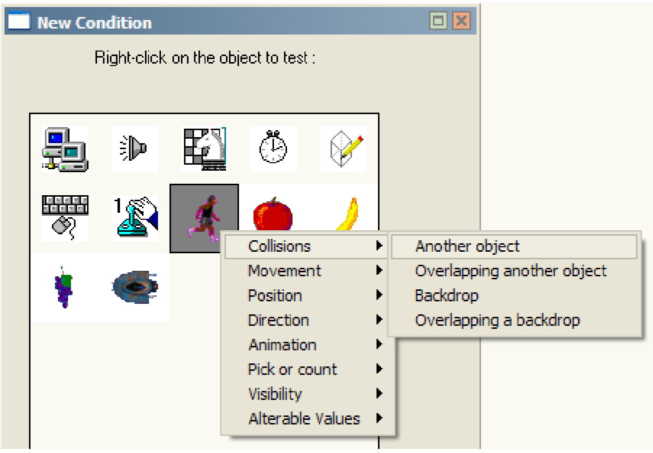
3 • New Condition

By using only two lines of “code”, we have achieved to make a UFO fly around the top of the screen dropping apples that falls downwards, make a controllable animated player that is prevented from exiting the screen at the left and right. Simple isn’t it?

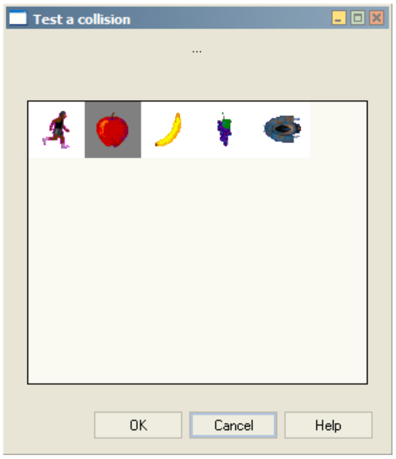
In the event editor, create a few more events that creates the fruits, but change the time between each fruit drop.

Collecting the fruit

Until now, the fruit would pass right through our player. We need to add an event for each fruit that we want to collide with our player.

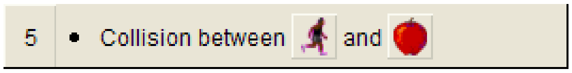


Make a new event and select the “Collision -> Another object” action as shown on the image.



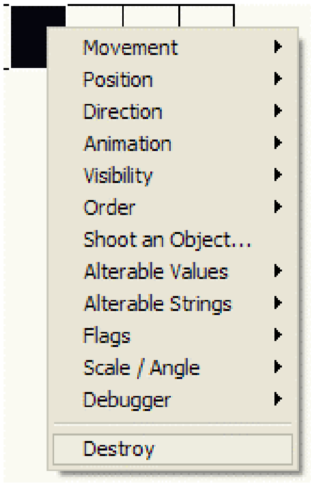
Here you select the object you want to test if collides with the runner object.

Your new event should look like this one:

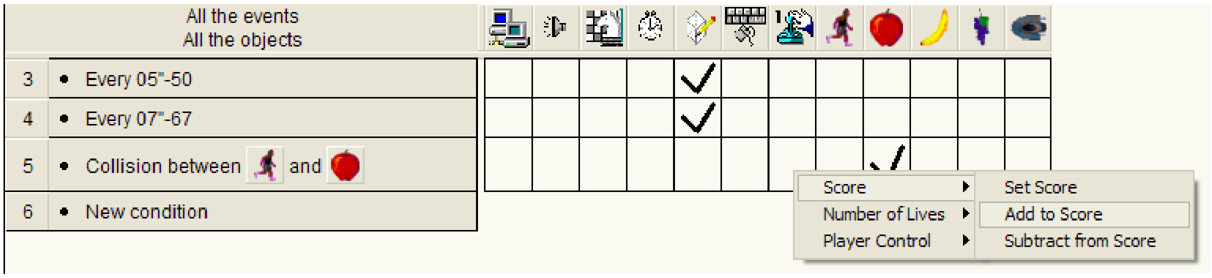

5 • Collision between and

This event basically asks MMF2 “Have our runner just collided with an apple?”
When the answer for that question is “yes”, it is our job to tell MMF2 what we then want to happen. In our case we want the apple to destroy and we want to give the player some points.

So in our new event, in the cell below the apple, we right click and select the ‘Destroy action’:



To give some score points to the player, we right click below the ‘Player 1’ object and select the “Score -> Add to score” action.



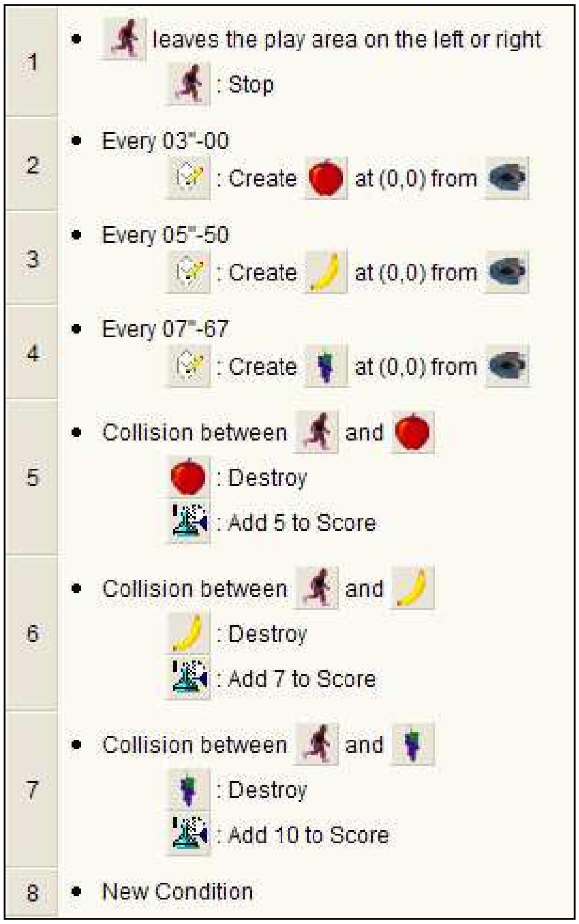
All the events
All the objects

3 • Every 05"-50
4 • Every 07"-67
5 • Collision between and
6 • New condition

You can decide how many points you want to award the player for collecting the apple. In this tutorial I decided to give the player 5 points.

Just like before, continue these steps to add collision test events for the other fruits.

After the other fruits where added, this is what the Event List Editor looks like: As you can see, I’ve made the banana and the grape fruit drop with bigger intervals so they don’t drop as often as the apple. I’ve also changed how many points the player will be awarded when they collide with the bananas and grapes.



1 • leaves the play area on the left or right
: Stop

2 • Every 03"-00
: Create at (0,0) from

3 • Every 05"-50
: Create at (0,0) from

4 • Every 07"-67
: Create at (0,0) from

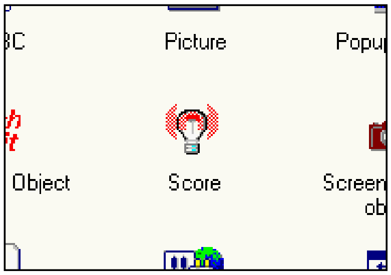
5 • Collision between and
: Destroy
: Add 5 to Score

6 • Collision between and
: Destroy
: Add 7 to Score

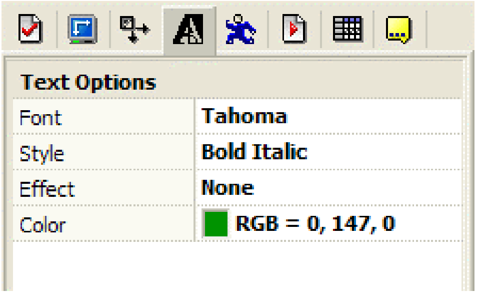
7 • Collision between and
: Destroy
: Add 10 to Score

8 • New Condition

Currently there is no way the player of our game can watch his/her score, so we head back to the Frame Editor and insert a Score object.



When you insert the new score object, it will display a small black ‘0’.
If you want to make the font bigger and change its colour, select the score object and go to the font properties in the Properties sidebar:

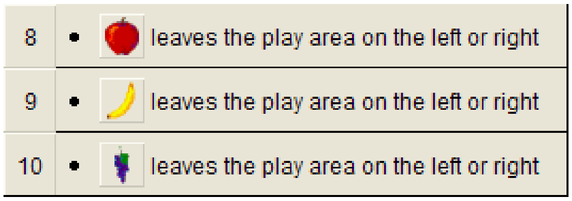


Text Options	
Font	Tahoma
Style	Bold Italic
Effect	None
Color	🟩 RGB = 0, 147, 0

Here I’ve changed the visible font and style of the score.

There are still a few things left to do before we are completely finished with our small game. If the UFO drops the fruits too close to the borders of the screen, the fruits can fall out of the screen making them impossible to collect.

For each fruit object we have, we make an event similar to the event that prevents the ‘Runner’ object from leaving the frame:



8	•	leaves the play area on the left or right
9	•	leaves the play area on the left or right
10	•	leaves the play area on the left or right

In these events you add a ‘bounce’ action below at the correct fruits. This will make the fruits bounce back into the frame if they are about to leave it.

Finalizing your game

Our game is pretty much finished; all we need is to spice up the game a little.
Here are some things for you to try:

- Change the frame background colour to something else than white
- Add some backdrops from the library to make the game more visual appealing

You can also try adding sounds that will play when you collect your fruits. See the documentation how to do this.

Done

Thank you for reading this tutorial. I hope you have gotten some insight in simple game creation with Multimedia Fusion 2. If you have any questions regarding this tutorial, you can always post a question on the Clickteam forums at www.clickteam.com