PLATFORMER TUTORIAL

Start by **loading sprites** for the character you will be using in your game. The ones for this tutorial are in the resources folder provided with the example. Click the red pac-man icon in the top toolbar and load the appropriate sprite. The character and enemy sprites must be the same size (here they are 16x16px) so that when the sprite object changes or animates, it will not shift to a different location. Also, this allows collision detection between the characters to be smooth. When creating your sprites, make sure you **CENTER** each sprite to X:8 Y:8. Now create the following sprites:

- ♦ Playable character
- ♦ Playable character's death sequence
- ♦ Enemy
- ♦ Enemy death sequence
- ♦ Platforms (BLOCKS)
- ♦ Phantom BLOCKS (for invisible setup objects)
  ~ Note: to avoid confusion with blocks of actions/code, all references to BLOCKS as platforms or invisible objects will be capitalized as such: BLOCKS

Now **create an object** for each of these sprites by clicking on the blue ball icon. Name your objects so that they are easy to understand.

- ♦ Create the character object.
- ♦ Create the BLOCK object.
- ♦ Create the dying character object.
- ♦ Create the enemy object.
ADDING GRAVITY

To start, we will be working on the core of all platformers, gravity. First, click “Add Event” and add a new “Step” event. In this event, go to the “Controls” tab and add a new “Start of a block” and create a “Check Object” if-statement. Navigate the first dropdown menu titled “Object” and select BLOCK. This will check if the player sprite is colliding with the ground. Now change the Y value to 4, keeping X at 0. Since the player’s sprite is 8px, the sprite will be offset from the BLOCK by 4px so that they player looks like they are standing on the BLOCK instead of walking through it. Make sure you then check “Relative” and “NOT,” meaning the player is not colliding with a BLOCK. Anyone familiar with programming will recognize the check object action (as well as other checks like the test variable for example) as if-statements at their core.

Following our check object action, add the action “Set the gravity.” In the options, set the direction to 270 (meaning straight down) and the gravity to 0.5. This will pull the player toward the bottom of the screen if they are not colliding with a BLOCK. Now add an “Else” & within it, add another “Set the gravity” action with the direction 270 and the gravity at 0. Now the player will stop when they collide with a BLOCK. Now end the check by adding an “End of a block”.
ADD & IMPLEMENT TERMINAL VELOCITY

Start a new block with “Start of a block.” Under the step event in the character, add an “If an expression is true” statement located in controls. For the expression, check if $v_{speed}>6$. If it is, “Set the vertical speed” to 6. This makes the maximum vertical velocity that the player can achieve to 6. Now end the check by adding an “End of a block.”
Start a new block with “Start of a block.” Add the check, “If there is an object at a position.” In its options, navigate to the BLOCK object and set the Y value to -8. Check the box that says “Relative.”

Start a new block with “Start of a block.” Add the check, “If an expression is true.” Under the expression text area, type vspeed<-1. This will test if the player is trying to jump up through platforms. If the player jumps, their vertical speed will be negative and the test will be true. Add a “Reverse vertical direction” action. This will now prevent the player from jumping up through BLOCKS by stopping their vertical motion. Add two “End of a block” actions to end the blocks of actions.
COLLIDING WITH A PLATFORM / BLOCK

We will now change the vertical velocity of the player when they collide with a platform. This will stop the player so that they do not fall through. Add the new event “Collision” and select the platform BLOCK object. Under the actions, add a new “Jump to position” action (located in move), keeping the X value at 0 and changing the Y value to -4. Check the “Relative” box. This means that the player will move up 4px when they collide with the BLOCK so that they are not standing inside of the BLOCK. Now add an “Align to a grid” action with values: Snap Hor: 1 and Snap vert: 8. This will snap the player object to...
the grid (above the BLOCK) so that they do not become stuck. Now “Set the vertical speed” to 0 so that the player stops moving downward. You are now ready to test!

Now create a room. At the top of the Room Properties should be two boxes that say “Snap X” and “Snap Y.” Change the values to 32. Now add a line of BLOCKS at the bottom. Place a character object somewhere near the top. SAVE the game and RUN. You should see the character fall and stop at the BLOCK.
MOVEMENT

We now want to move the player. Add the events <Left>, <Up>, and <Right> located under “Keyboard.”

LEFT: In the <Left> event, add the check action “If there is an object at a position” and navigate the object value to BLOCK. Change the X value to -4 and keep the Y value at 0. Check “Relative” and “Not”. Start a new block with “Start of a block.” Add the action “Jump to position” and change the X value to -4 and keep the Y value at 0. Make sure you check “Relative”. This is important because you want the player to move -4 pixels to the left relative to their current position. Otherwise the player will not move. Now end the check by adding an “End of a block.”
RIGHT: In the <Right> event, add the check action “If there is an object at a position” and navigate the object value to BLOCK. Change the X value to 4 and keep the Y value at 0. Check “Relative” and “Not”. Start a new block with “Start of a block.” Add the action “Jump to position” and change the X value to 4 and keep the Y value at 0. Make sure you check “Relative”. Now end the check by adding an “End of a block.”
UP: In the <UP> event, add the check action “If there is an object at a position” and navigate the object value to BLOCK. Keep the X value at 0 and keep the Y value at 4. Check “Relative” and “Not”. Start a new block with “Start of a block.” Add the action “Jump to position” and keep the X value to 0 and change the Y value at -4. Make sure you check “Relative”. Add the action “Start moving in a direction,” click the vertical direction and set the speed to 9. Now end the check by adding an “End of a block.” Now add a “Jump to Position” and set the Y value to -1. Keep it “Relative.”

In the room, add a few more BLOCKS 2 grid spaces away from the bottom. SAVE the game and RUN. You should now be able to move the player back and forth and collide with the BLOCKS when you jump.

INTERACTION with ENEMIES

We will now allow a player to have a dying action. First, edit the properties of the DEAD CHAR object to create the animation effect for when the player dies. Create on “Add Event” and make a new “Animation End” event. Now go to the Main2 tab and make a “Restart the current room” action. Change the transition mode to “Blend.” Click okay to save this object.
In the Character object properties, add a new event “Destroy.” In the actions, add “Create moving instance of…” located in Main1. Navigate the object dropdown to “DEAD CHAR”, leave the X and Y values at 0, set speed to “speed” and direction to “direction.” Check the “Relative” box. This will play the destroyed character animation when the player dies.
Now create the enemy’s dying object. Edit the properties of **ENEMY KILLED**. Create on “Add Event” and make a new “**Animation End**” event. Add a new “**Destroy the instance**” action located in Main1. This will destroy the instance of the enemy, so make sure the “Applies to” has “**Self**” selected.

Modify the Character object by adding a new “**Collision**” event with an “**ENEMY**”. Begin by adding, “**If an expression is true**,” setting the expression to “y<other.y”. This checks if the y position of the player is less than the y position of another object. Start a new block with “**Start of a block.**” Add “**Destroy the instance**” applying to “Other.” Add “**Reverse vertical direction**” applying to “Self.” Add a new alarm applying to **Object: ENEMY COUNTER**. Set the **number of steps to 32** and the alarm number to **Alarm 0**. This checks if the player is colliding with the enemy and if they are above the enemy (instead of next to them). Then the player crushes the enemy, destroying that instance of the enemy.
Open the **ENEMY** object properties and create the new event “**Create.**” In this event, add the action “**Start moving in a direction.**” Click the left arrow and give it a **speed of 1**. Relative is not checked.
Now create a “Destroy” event. Add “Create instance of object” located in Main1 and navigate to object ENEMY KILLED. Make sure to click “Relative.” This will play the dying enemy animation at its current location and destroy that enemy.

Now we want to add gravity to the enemy. Create a “Step” event. Start a new block in the actions panel and check “If there is an object at a position.” The object should be set to BLOCK, the Y value should
be 4, and “Relative” and “Not” should be checked. **Set the gravity to 0.5 at 270 degrees.** Add an “Else” statement, which **sets the gravity to 0.** Now end the check with an **“End of a block.”** This chunk of code adds gravity to the enemies so that they remain in contact with the platforms.

Add the new event **“Collision”** and select the platform BLOCK object. Under the actions, add a new **“Jump to position”** action (located in move), keeping the **X value at 0** and changing the **Y value to -4.** Check the **“Relative”** box. This means that the enemy will move up 4px when they collide with the BLOCK so that they are not standing inside of the BLOCK. Now add an **“Align to a grid”** action with values: **Snap Hor: 1** and **Snap vert: 8.** This will snap the enemy object to the grid (above the BLOCK) so that they do not become stuck. Now **“Set the vertical speed”** to 0 so that the player stops moving downward. You are now ready to test!

![Game Maker interface showing object properties and event actions](image_url)

Place an enemy near the bottom platform. **SAVE** and **RUN** the game. Check that you can jump on the enemy and kill it as well as die when you run into the enemy.

**SMARTER ENEMIES**

In the **“Step”** event for the ENEMY object, start a new block with **“Start of a block”** action. Add a new check: **“If an expression is true”** checking if the vspeed\(>6\). If so, add a new action with **vertical speed = 6.** End the block. This sets the terminal velocity for the enemy.
Now we want the enemies to wander around. Start another block and check “if there is an object at a position.” Check if the object is a BLOCK at the x-value of -1 (relative). This checks if the enemy walks into a platform. “Set the horizontal speed” to 1, making the enemy turn around if they walk into a wall. Now check if the enemy walks into a platform in the other direction by checking at x = 1. “Set the horizontal speed” to -1.

Next, we check if the enemies jump and hit a platform. We start a new block and check if there is a BLOCK at that position. In here, we want to check if the vspeed<-1 and if so, reverse the vertical direction. End this block of actions.

Start another block. We will now give the enemy a random jump action: a very basic AI feature. Add a check “If there is an object at a position.” Check with object BLOCK at y-value: 1 (relative). Create a new block. Add the random action, “With chance 1 out of perform next.” Change sides to 160. This should change the action to say “With chance 1 out of 160 perform next.” Add “Set the vertical speed” and set the speed to -8. Now end both of the blocks. This will make the enemies jump randomly.
Place some enemies near the bottom platform. **SAVE** and **RUN** the game. Enemies should jump around randomly.

**ENEMY COLLISIONS**

As you noticed, the enemies fell off the edge. We now want to make a barrier to keep them in frame. In
the object titled **ENEMY MARKER**, make sure that nothing is added to events and that the object is **NOT visible**. This object will act as an invisible barrier so that the enemies will not fall off the screen.

Add a **collision** event with the ENEMY MARKER. Add the action “**Reverse horizontal direction**” that applies to “**Self**.” This will make the enemy turn around when they hit one of these invisible BLOCKS. **Outline the walls of the room** with these markers so that the enemies are contained.
Now add a collision event with an ENEMY. Add the action “**Bounce against all objects.**” Set precise to “**not precisely**” and against to “**all objects.**” This will make the enemy bounce and seem like it’s acting on its own.

Finally design a level, **save**, and **run** the game.

**WINNING CONDITION**

In the **ENEMY COUNTER** properties, add a “Create” event. In this event, “**Set Alarm 0**” to **160** steps. Add event “**Alarm 0**.” Here, “**Set Alarm 0**” to **160**. Add the check, “**If the number of instances is a value.**” Check if the object **ENEMY is equal to 0**. Start a code block with “**Start of a block.**” **Display the message** “Enemies destroyed. Congratulations!” Add a “**Restart the game**” action and end the code block.
Congratulations! You have completed the tutorial!