IGM Tutorial III: Basic Platformer

by Kain Vinosec

In this tutorial, we're going to cover the very basics of creating a jumping/platformer style game which will include creating a hero gadget, creating tiles with and without walls, and creating an enemy gadget for our hero to defeat. At the bottom of this guide you'll find a link that will lead you to a flash example of what the finished project should look like. You can also click an of the Screenshots to see a full-sized version for clarity. You will also need the resources in the following archive to do this tutorial (right click => save as to save them to your computer)

Resources

http://blog.rpgmakerweb.com/wp-content/uploads/2012/02/Grass.png

http://blog.rpgmakerweb.com/wp-content/uploads/2012/02/Hero.png

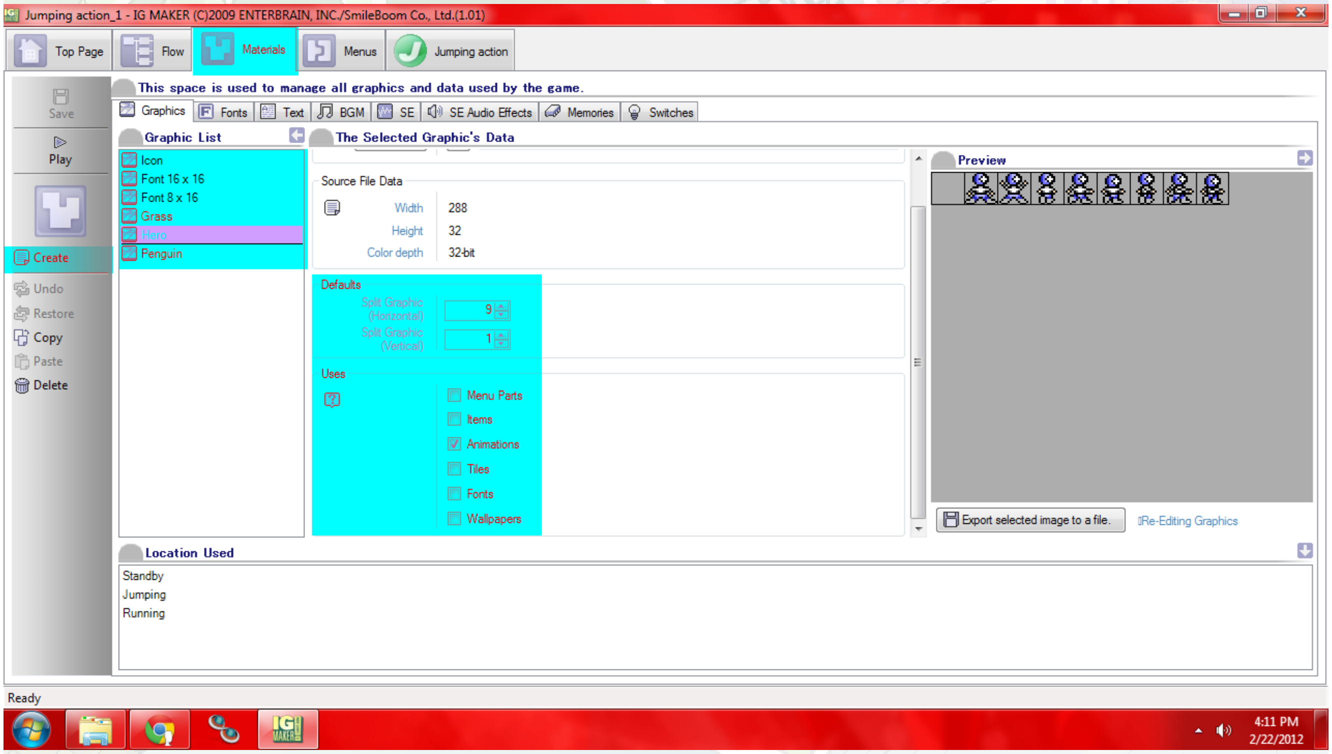
http://blog.rpgmakerweb.com/wp-content/uploads/2012/02/Penguin.png

Open up IG Maker and on the left menu under the "Top Page" tab select New Game. This opens up a guide to creating new projects. Read it if you like, then hit next. For this game's genre we're going to select Jumping Action (New Game). Hit next again and we get to pick the starting screen resolution (this can be changed later by selecting the "Flow" tab and then the "Overall Settings" sub-tab). Select 640×480 under the VGA 4:3 option. This option seems to be the best for flash, and it works fine for PC games as well (though I find that Wide 16:9 1280×720 works best for the Xbox 360). Hitting next again allows you to name your game. This can be changed later so just use the default for now and hit finish.

Now you've been taken to the "Flow" Tab. We're not going to do anything here right now, so let's head over to the "Materials" tab and import our three files. When you select "Materials" you're taken to another sub-tab called "Graphics". In the "Graphic List" you'll see three files; Icon, Font 16×16, and Font 8×16. These files cannot be deleted, but they can be changed. Icon is worthy of note as this lets you change the icon for your game's executable on the PC or the icon used on the Xbox 360's menus. For now we're going to select the "Create" option in the menu to the left.

Go ahead and import Grass.png first. For "Split Graphic (Horizontal)" we're going to change the number from 1 to 3. You'll notice to the right that this splits the graphic into three equal parts, defining the tiles we have. There is no need to change "Split Graphic (Vertical)" because there is only one line of tiles. Under "Uses" you only need the "Tiles" check box selected. You can leave the others blank. Now use the "Create" option again to import the Hero and Penguin files. For the Hero's Split

Graphic (Horizontal) you'll want to set it to 9, and the Penguin's to 6. Both of their Uses can be set to Animations only.



This space is used to manage all graphics and data used by the game.

Graphic List: Icon, Font 16 x 16, Font 8 x 16, Grass, Hero, Penguin

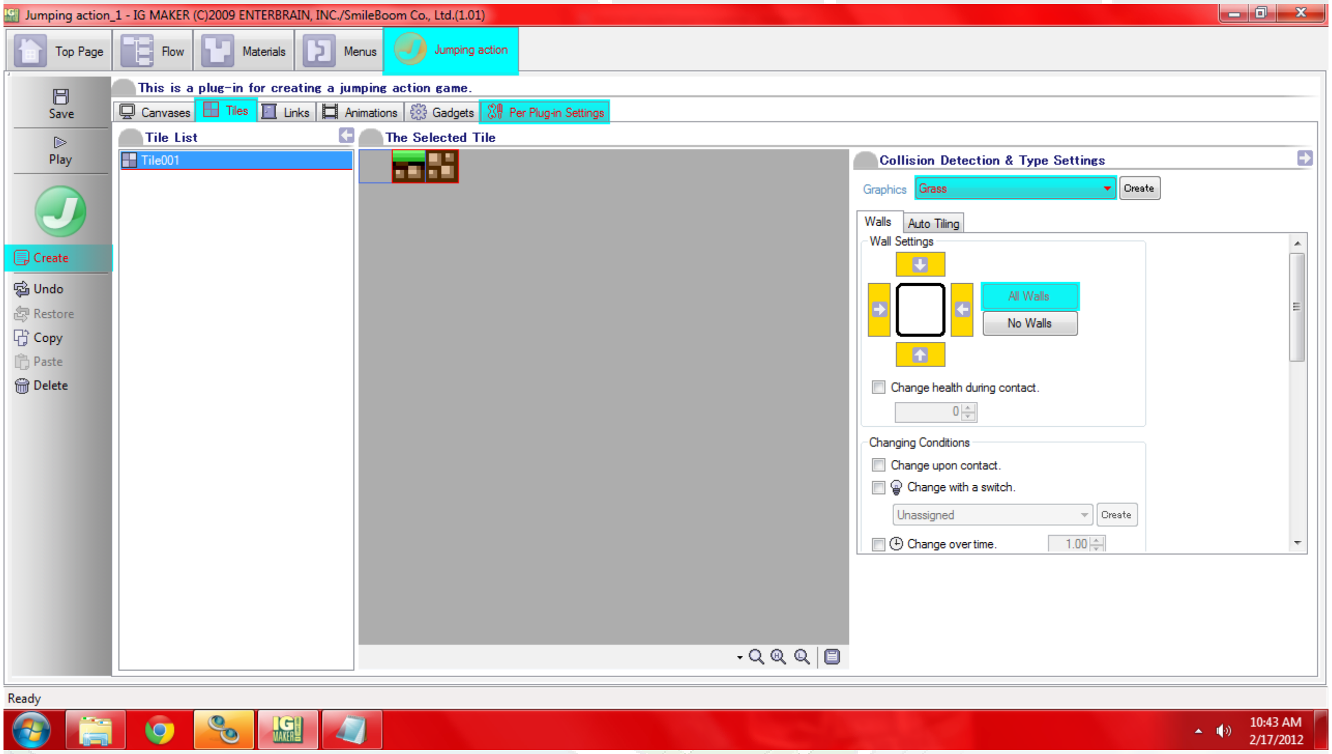
Source File Data
Width 288
Height 32
Color depth 32-bit

Defaults
Split Graphic (Horizontal) 9
Split Graphic (Vertical) 1

Uses
□ Menu Parts
□ Items
☑ Animations
□ Tiles
□ Fonts
□ Wallpapers

Location Used
Standby
Jumping
Running

Now that we've got our graphics loaded as materials, we can set up the tiles for the first canvas but first we have to set the "Basic Tile Size" for our game. So select the "Jumping Action" tab and then select the "Per Plug-In Settings" sub-tab. The "Basic Tile Size" we want is 32×32 so select that and then select the "Tiles" sub-tab. Hit the "Create" option and it'll make a new blank tile set called Tile001. To the right is a drop down menu for graphics. Select that and choose the Grass option. You'll notice that it loads the tiles with a blue border around them. The size of the blue border is what we changed when we switched the "Basic Tile Size". Setting the size smaller than the tiles you're using will give you more control over how each part of the tiles react but for now we just want both of the visible tiles to be complete walls. Select both tiles (you can select both by holding down the left mouse button and dragging it from one tile to the next) and to the right we're going to hit the "All Walls" button. You'll notice the boxes with arrows telling you where walls exist will all become yellow for both of our tiles.



Jumping action_1 - IG MAKER (C)2009 ENTERBRAIN, INC./SmileBoom Co., Ltd.(1.01)

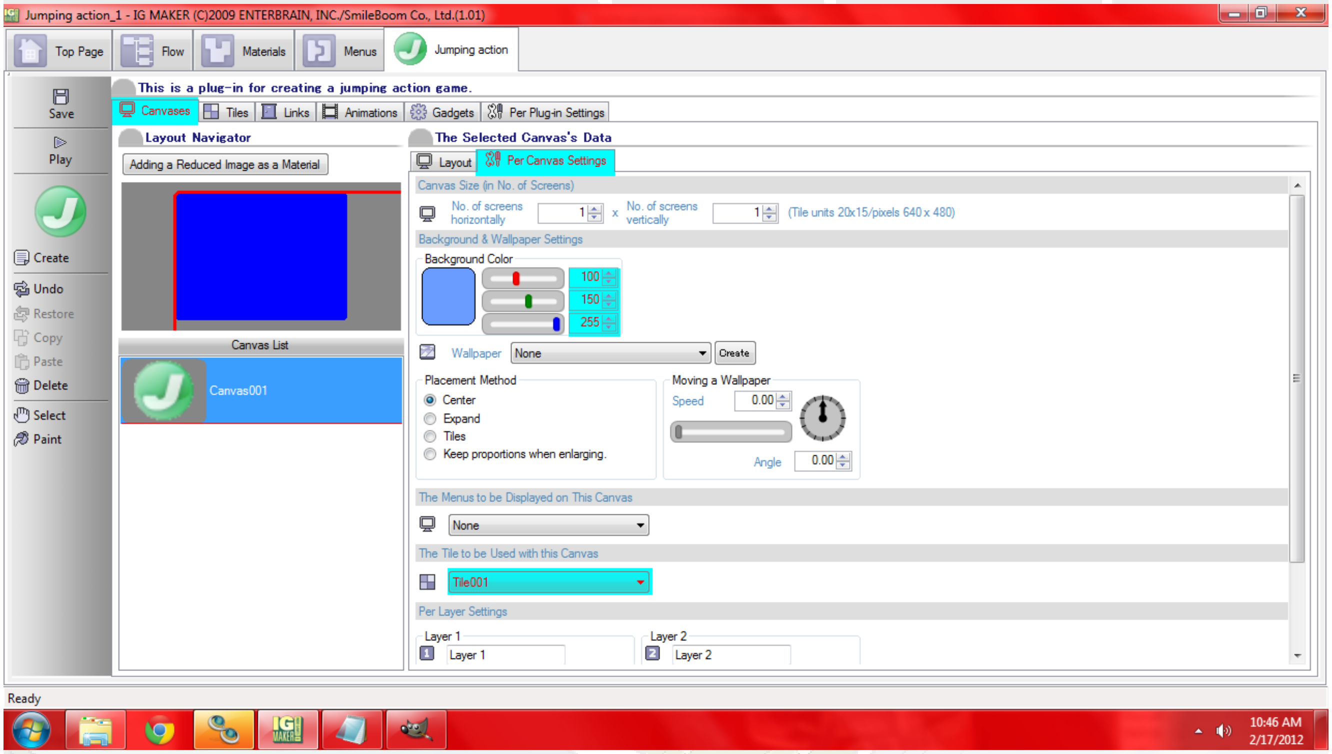
This is a plug-in for creating a jumping action game.

Graphics: Grass

Tile List: Tile001

Ready

Now let's fill in our first canvas. Select the "Canvases" sub-tab, and then select the "Per Canvas Settings" sub-sub-tab. For "The Tile To Be Used With This Canvas" select Tile001 from the drop down menu. Now in order to create a sky background for our tiles, we're going to use the "Background Color" option. Try these settings for a day-blue sky: Red 100, Green 150, Blue 255.



Jumping action_1 - IG MAKER (C)2009 ENTERBRAIN, INC./SmileBoom Co., Ltd.(1.01)

This is a plug-in for creating a jumping action game.

Canvas Size (in No. of Screens)
No. of screens horizontally 1 x No. of screens vertically 1 (Tile units 20x15/pixels 640 x 480)

Background & Wallpaper Settings
Background Color 100 150 255
Wallpaper None

Placement Method
● Center
○ Expand
○ Tiles
○ Keep proportions when enlarging.

Moving a Wallpaper
Speed 0.00
Angle 0.00

The Menus to be Displayed on This Canvas
None

The Tile to be Used with this Canvas
Tile001

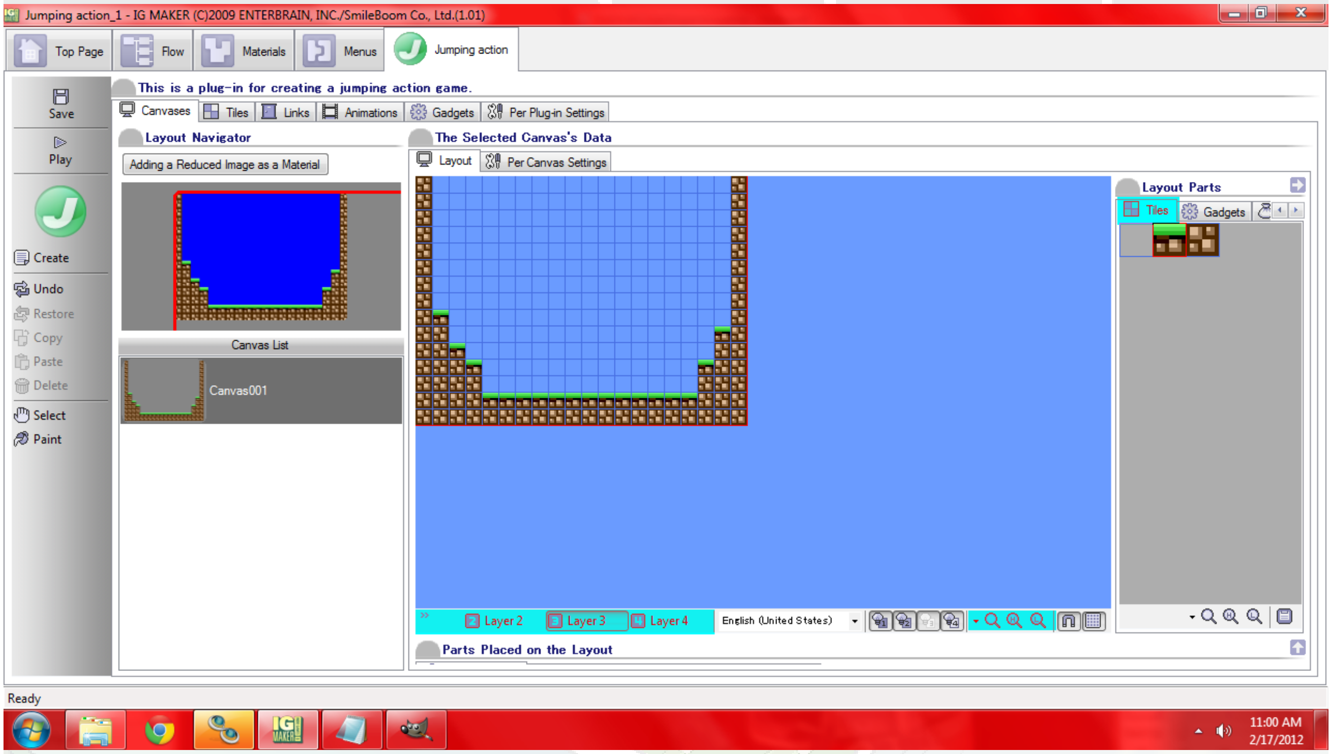
Per Layer Settings
Layer 1 Layer 1
Layer 2 Layer 2

Ready

10:46 AM
2/17/2012

Now we'll switch back over to the "Layout" sub-sub-tab. You'll see that it's all very blue and to the right you'll see the tiles we set up. Switch over to your preferred layer (layer options are located just below the grid on the left), and you can probably go ahead and zoom the grid out (you can zoom in and out using the magnifying glass icons or the drop down menu located below and to the right of the grid). Before I continue with placing tiles let me explain how the four layers work.
The layers represent the depth of on-screen items. Putting things on Layer 1 will put them at the very front of the screen and Layer 4 would be the very back. Objects cannot interact across layers so you have to choose one layer to build off of. I like Layer 3 as I don't usually have very much stuff behind my character but it is fun to put things infront of them like bushes or buildings and things that you can run behind.

So now let's place some tiles. Choose whatever layer you like and place your tiles however you want. Remeber they don't have to just form the ground. You can have them float in the air, or be various sizes. Here's what I came up with.



You'll notice I've got walls going up either side. This helps with gadgets as you can have them alter course or actions based on the locations of walls. You don't need walls that go all the way up the sides of the screen though. Simply making them one tile higher than the ground would be fine for this tutorial.

Now we're going to set up our Hero and Penguin gadgets, but first we'll need to set up their animations. Select the "Animations" sub-tab and let's get started. Like before the first thing you'll need to do is hit the "Create" option which creates Animation001, and its first action called Action001 which has two displays called Right and Left. Let's rename Animation001 to Hero and Action001 to Standby. Under the "Per Action Settings" sub-sub-tab, you'll want to check the box next to "Set Origin At Character's Feet". This allows for easier placing of the Hero gadget later on. Under "Graphic Data" you'll need to select the drop down menu and choose the Hero graphic. Then you'll want to switch to the "Per Frame Settings" sub-sub-tab and notice down at the bottom there is a gray bar with the word "Graphic" in it. If you put your mouse around the top of this bar you'll be given the option to resize the window. Click and drag the window up so that you're able to see the graphics for the Hero.

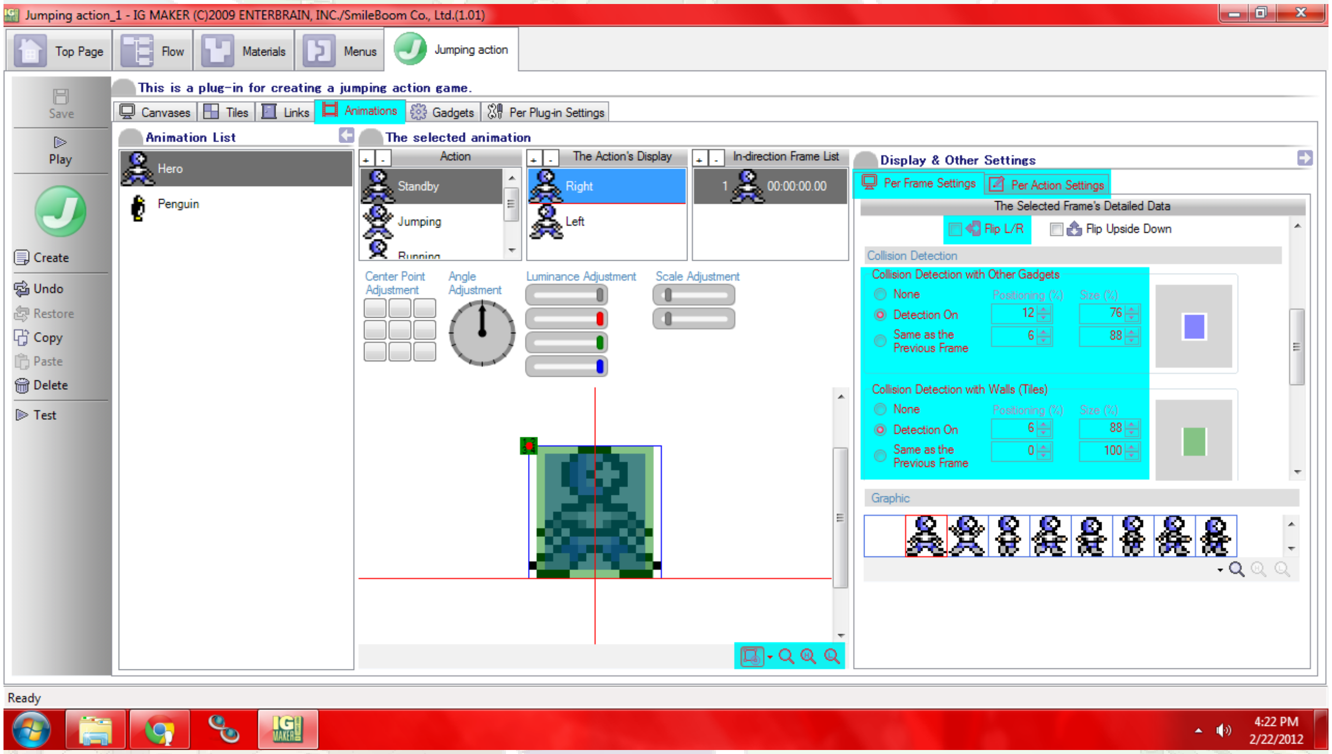
For Standby > Right > 1st Frame select the first tile with the hero in it. Do the same for Left, but this time under "Display Settings" there is a check box for "Flip L/R" that will mirror the tile so it looks like he is facing left. Next we'll need to set up our hero's collision detection. He does not need collision detection for attacks at the moment. Here are the Collision Detection numbers that I used.

Collision Detection with Other Gadgets
Horizontal Positioning % (top left box) = 12
Horizontal Size % (top right box) = 76

Vertical Positioning % (bottom left box) = 6
Vertical Size % (bottom right box) = 88

Collision Detection with Walls (Tiles)
Horizontal Position % = 6
Horizontal Size % = 88
Vertical POsitioning % = 0
Vertical Size % = 100

Now if you'll look below the image of our character on the grid you'll notice three zoom-altering magnifying glass icons and an odd box icon to the left of those. If you click that you'll see the collision detection we've just set up. It should look like a thin green bar in the middle of our character from head-to-toe and a blue bar inside of the green one.


Jumping action_1 - IG MAKER (C)2009 ENTERBRAIN, INC./SmileBoom Co., Ltd.(1.01)

This is a plug-in for creating a jumping action game.

The Selected Frame's Detailed Data

Collision Detection with Other Gadgets
○ None
● Detection On
○ Same as the Previous Frame

	Positioning (%)	Size (%)
	12	76
	6	88

Collision Detection with Walls (Tiles)
○ None
● Detection On
○ Same as the Previous Frame

	Positioning (%)	Size (%)
	6	88
	0	100

Now we're going to make our character jump. Click the + button next to "Action" and it'll create a new action below Standby. Rename this new action Jumping. Follow the steps that we did for Standby for "Per Action Settings". For the graphic select the second tile (where his arms are spread out). Remember for the Left direction you'll need to flip the image. When you're done with this set up your collision detection the same as before.

By now you should be getting comfortable with creating new actions. So create another new one and rename it Running. Follow the same steps as before. Now we need to add five new frames for each direction of running so click the + button next to "In-Direction Frame List" and it'll create a new blank frame. Set the graphics for the running animation. Don't forget your collision detection (take note of the “Same as the Previous Frame” option under “Collision Detection” as this enables you to only need to set the detection for the first frame in a series and have all the other frames copy it). Next go to “Per Action Settings” and change your “Frames Displayed per Second” from 2 to 20. You do this after you’ve got everything else set up because it doesn’t update frames created after this setting has been changed, only those created before. After you set the frames per second to 20 click the checkbox that enables infinite looping.

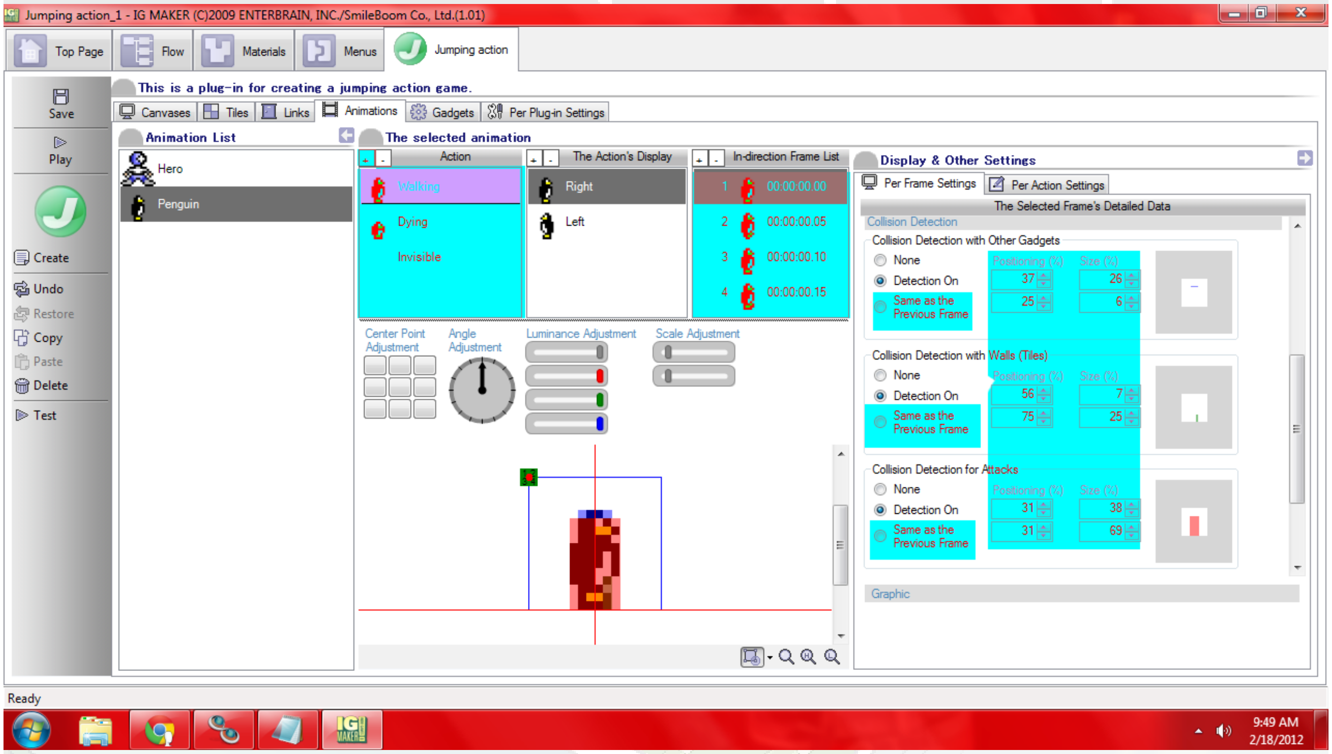
Now you’ll need to create a new animation for the Penguin enemy. The penguin doesn’t ever stand still so you don’t need to create a standby animation. For each direction of walking you’ll need four frames. The tiles you want will be the first, second, first, and third. This will make it appear as if he’s taking steps. He also needs a new action to die so you’ll need a left and right two-frame animation for his death (without infinite looping). Use the far right tiles in the penguin graphic where he looks smooshed. You’ll also need a third action that we’re going to call Invisible. Set it up the same as before with no collision detection and use the blank space on the far left of our penguin graphic. This allows our penguin to disappear once he is killed. Here is what you’ll need to set the collision detection for the penguin’s walking animation.

Collision Detection with Other Gadgets
Horizontal Positioning % (top left box) = 37
Horizontal Size % (top right box) = 26
Vertical Positioning % (bottom left box) = 25
Vertical Size % (bottom right box) = 6

Collision Detection with Walls (Tiles)
Horizontal Position % = 37
Horizontal Size % = 26
Vertical POsitioning % = 75
Vertical Size % = 25

Collision Detection for Attacks
Horizontal Position % = 31
Horizontal Size % = 38
Vertical POsitioning % = 31
Vertical Size % = 69

You don’t need any collision detection for the dying action as he should be harmless once stepped on. Here’s a screenshot that might help with some of the things I’ve just covered.



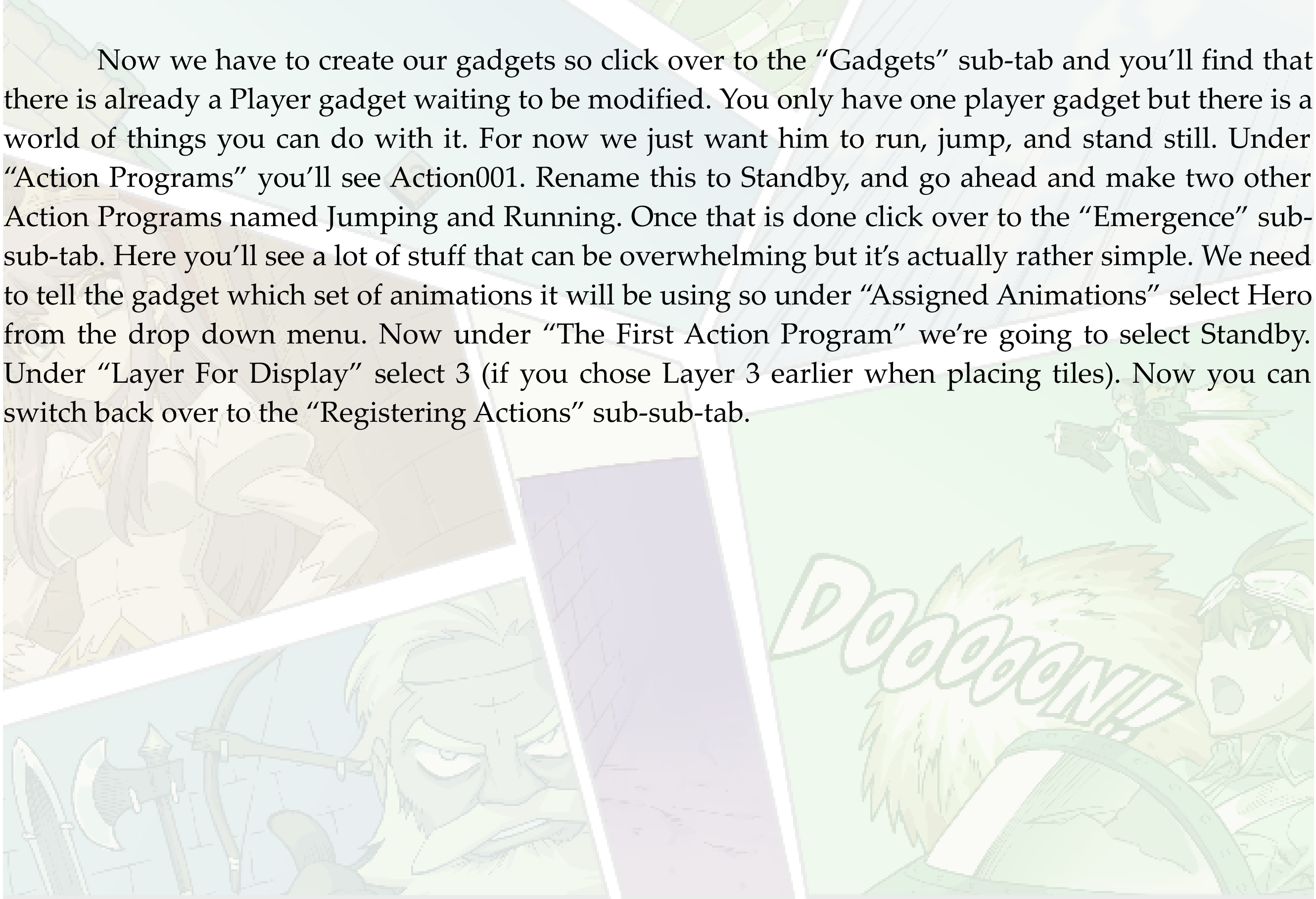
This is a plug-in for creating a jumping action game.

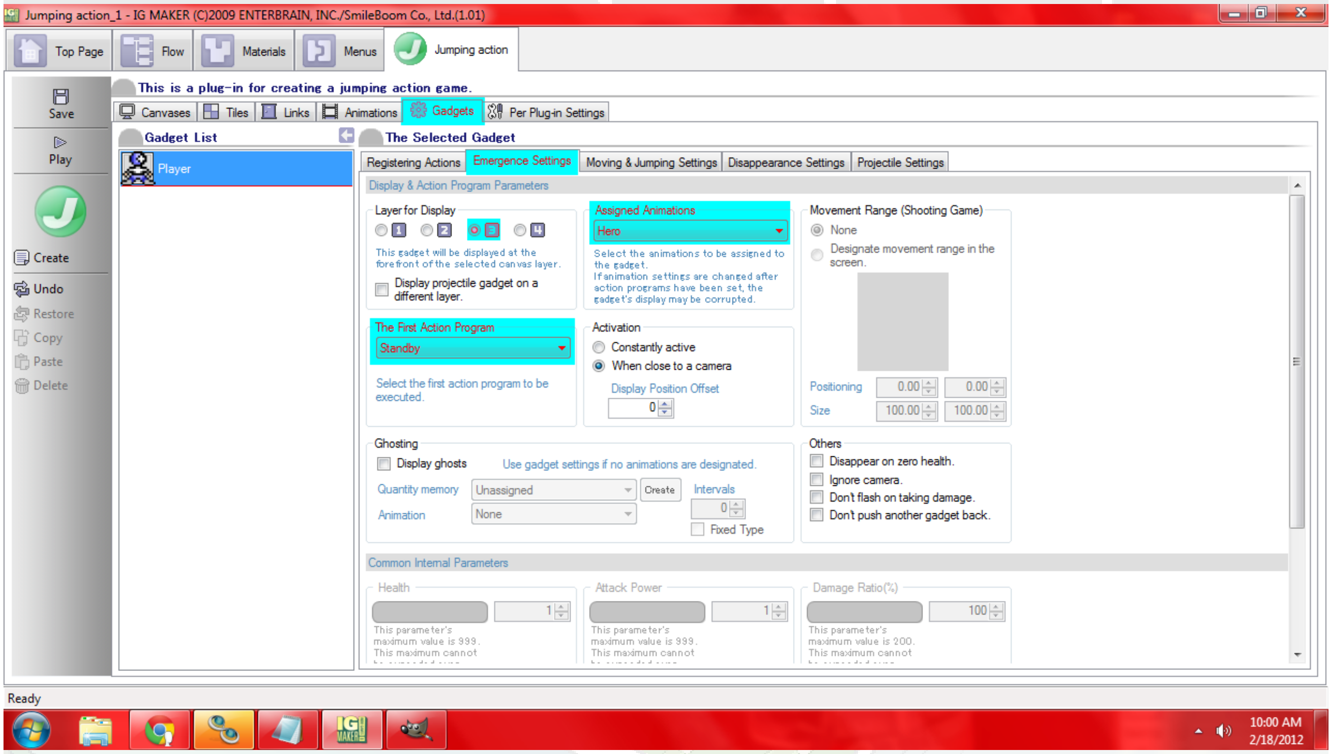
Animation List: Hero, Penguin

Action	The Action's Display	In-direction Frame List
Walking	Right	1 00:00:00.00
Dying	Left	2 00:00:00.05
Invisible		3 00:00:00.10
		4 00:00:00.15

Collision Detection	Setting	Positioning (%)	Size (%)
Collision Detection with Other Gadgets	○ None ● Detection On ○ Same as the Previous Frame	37 25	26 6
Collision Detection with Walls (Tiles)	○ None ● Detection On ○ Same as the Previous Frame	56 75	7 25
Collision Detection for Attacks	○ None ● Detection On ○ Same as the Previous Frame	31 31	38 69

Now we have to create our gadgets so click over to the "Gadgets" sub-tab and you'll find that there is already a Player gadget waiting to be modified. You only have one player gadget but there is a world of things you can do with it. For now we just want him to run, jump, and stand still. Under "Action Programs" you'll see Action001. Rename this to Standby, and go ahead and make two other Action Programs named Jumping and Running. Once that is done click over to the "Emergence" sub-sub-tab. Here you'll see a lot of stuff that can be overwhelming but it's actually rather simple. We need to tell the gadget which set of animations it will be using so under "Assigned Animations" select Hero from the drop down menu. Now under "The First Action Program" we're going to select Standby. Under "Layer For Display" select 3 (if you chose Layer 3 earlier when placing tiles). Now you can switch back over to the "Registering Actions" sub-sub-tab.





Jumping action_1 - IG MAKER (C)2009 ENTERBRAIN, INC./SmileBoom Co., Ltd.(1.01)

This is a plug-in for creating a jumping action game.

Gadget List: Player

The Selected Gadget — Emergence Settings

Display & Action Program Parameters

Layer for Display: ○ 1 ○ 2 ● 3 ○ 4

This gadget will be displayed at the forefront of the selected canvas layer.

□ Display projectile gadget on a different layer.

Assigned Animations: Hero

Select the animations to be assigned to the gadget. If animation settings are changed after action programs have been set, the gadget's display may be corrupted.

Movement Range (Shooting Game): ● None ○ Designate movement range in the screen.

Positioning 0.00 0.00

Size 100.00 100.00

The First Action Program: Standby

Select the first action program to be executed.

Activation: ○ Constantly active ● When close to a camera

Display Position Offset: 0

Ghosting: □ Display ghosts — Use gadget settings if no animations are designated.

Quantity memory: Unassigned — Intervals: 0

Animation: None — □ Fixed Type

Others: □ Disappear on zero health. □ Ignore camera. □ Don't flash on taking damage. □ Don't push another gadget back.

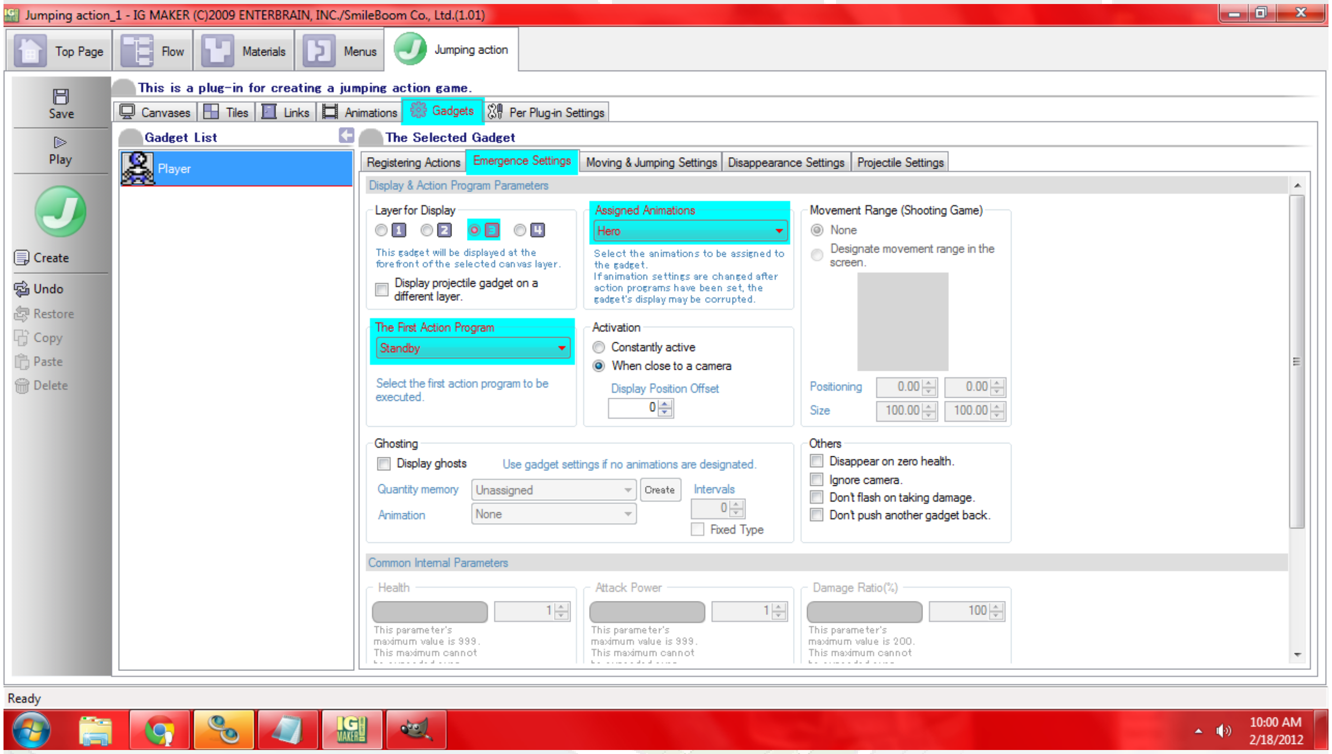
Common Internal Parameters

Health: 1 — This parameter's maximum value is 999. This maximum cannot

Attack Power: 1 — This parameter's maximum value is 999. This maximum cannot

Damage Ratio(%): 100 — This parameter's maximum value is 200. This maximum cannot

Now we need to set up how each "Action Program" acts when displayed on the screen. For all three you'll want to check the box for "Don't Designate A Direction Of Movement" and for Standby you'll want the "Movement Speed" to be changed to 0. For Jumping you'll want to make sure to check the box for "Jump". Now you need to set the animations under "Action To Be Assigned". The names should match up. For the directions just select right for now on all three.



This is a plug-in for creating a jumping action game.

Display & Action Program Parameters

Layer for Display: ○ 1 ○ 2 ● 3 ○ 4

This gadget will be displayed at the forefront of the selected canvas layer.

□ Display projectile gadget on a different layer.

Assigned Animations: Hero

Select the animations to be assigned to the gadget.
If animation settings are changed after action programs have been set, the gadget's display may be corrupted.

Movement Range (Shooting Game)

● None

○ Designate movement range in the screen.

Positioning 0.00 0.00

Size 100.00 100.00

The First Action Program: Standby

Select the first action program to be executed.

Activation

○ Constantly active

● When close to a camera

Display Position Offset 0

Ghosting

□ Display ghosts

Use gadget settings if no animations are designated.

Quantity memory: Unassigned Create

Intervals 0

Animation: None

□ Fixed Type

Others

□ Disappear on zero health.

□ Ignore camera.

□ Don't flash on taking damage.

□ Don't push another gadget back.

Common Internal Parameters

Health 1

This parameter's maximum value is 999. This maximum cannot

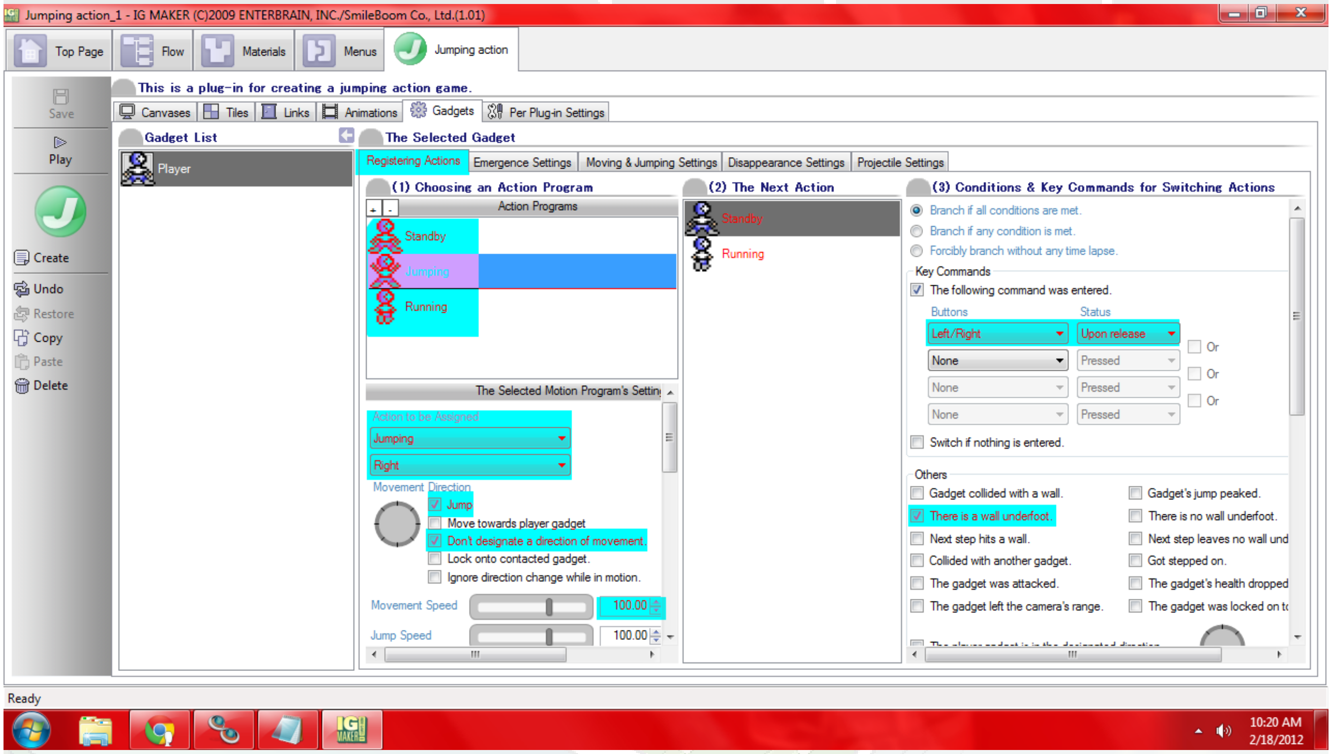
Attack Power 1

This parameter's maximum value is 999. This maximum cannot

Damage Ratio(%) 100

This parameter's maximum value is 200. This maximum cannot

Now we have to tell the game how each action interacts with the others. So select Standby for the "Action Program", and select Jumping as "The Next Action". Under "Conditions & Key Commands for Switching Actions" you'll want to set it up so the character jumps when a button is pressed (I used B). So check the box next to "The following command was entered" and select your choice of button for the action. For the "Status" choose "Upon Pressing". Now select Running as "The Next Action" and do the same, but for the "Buttons" select "Left/Right" and for "Status" select "Pressed". Now select Jumping as the "Action Program" and Standby as "The Next Action". Because we don't want Jumping to go straight to Running or Standby randomly, we'll want to set it up so that when "Left/Right" is released ("Upon release") and "There is a wall underfoot" your hero will go to standby. For Jumping to transfer to Running, you'd want "There is a wall underfoot" checked and you'd want "Left/Right" to be "Pressed". This allows you to keep running when you hit the ground if left or right is held down otherwise you'll simply land and stop. Now select Running as the "Action Program" and Standby for "The Next Action" and use "Left/Right" – "Upon Release". Then for Jumping you'll want it to be "B" (or whatever button you chose) – "Upon Pressing". This allows us to jump while running.



This is a plug-in for creating a jumping action game.

Gadget List: Player

The Selected Gadget

(1) Choosing an Action Program

Action Programs: Standby, Jumping, Running

The Selected Motion Program's Settings

Action to be Assigned: Jumping

Right

Movement Direction

☑ Jump

☐ Move towards player gadget

☑ Don't designate a direction of movement.

☐ Lock onto contacted gadget.

☐ Ignore direction change while in motion.

Movement Speed 100.00

Jump Speed 100.00

(2) The Next Action: Standby, Running

(3) Conditions & Key Commands for Switching Actions

● Branch if all conditions are met.

○ Branch if any condition is met.

○ Forcibly branch without any time lapse.

Key Commands

☑ The following command was entered.

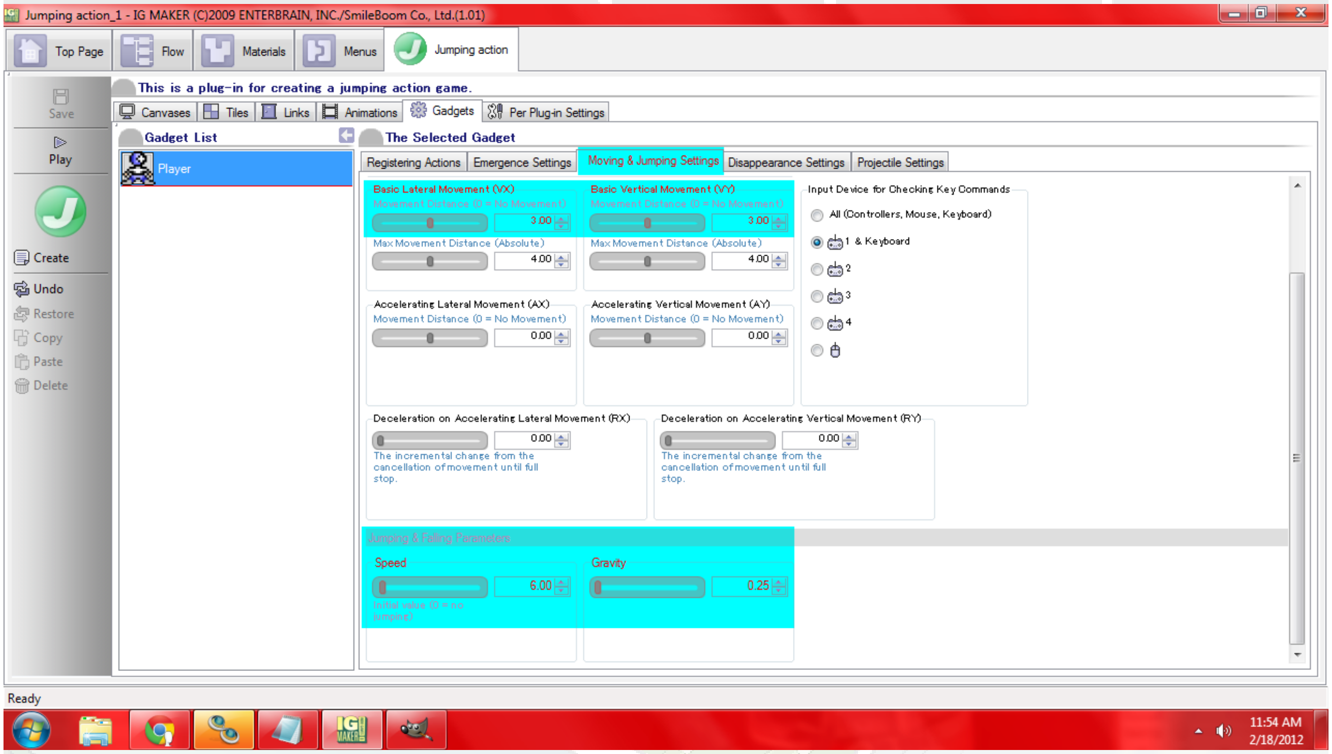
Buttons	Status	
Left/Right	Upon release	
None	Pressed	☐ Or
None	Pressed	☐ Or
None	Pressed	☐ Or

☐ Switch if nothing is entered.

Others

☐ Gadget collided with a wall.	☐ Gadget's jump peaked.
☑ There is a wall underfoot.	☐ There is no wall underfoot.
☐ Next step hits a wall.	☐ Next step leaves no wall und
☐ Collided with another gadget.	☐ Got stepped on.
☐ The gadget was attacked.	☐ The gadget's health dropped
☐ The gadget left the camera's range.	☐ The gadget was locked on to

Now we need to make our hero move with a bit more fluidity. If you tested the game now he wouldn't even be able to jump! So click the "Moving & Jumping Settings" sub-sub-tab. Under "Basic Lateral Movement (VX)" and under "Movement Distance" change it from 1 to 3. Do the same with "Basic Vertical Movement (VY)". Now scroll down and under "Jumping & Falling Parameters" adjust "Speed" to 6, and "Gravity" to 0.25. This will allow our hero to jump and run at a comfortable speed.



This is a plug-in for creating a jumping action game.

The Selected Gadget

Basic Lateral Movement (VX)
Movement Distance (0 = No Movement): 3.00
Max Movement Distance (Absolute): 4.00

Basic Vertical Movement (VY)
Movement Distance (0 = No Movement): 3.00
Max Movement Distance (Absolute): 4.00

Accelerating Lateral Movement (AX)
Movement Distance (0 = No Movement): 0.00

Accelerating Vertical Movement (AY)
Movement Distance (0 = No Movement): 0.00

Input Device for Checking Key Commands
○ All (Controllers, Mouse, Keyboard)
● 1 & Keyboard
○ 2
○ 3
○ 4
○

Deceleration on Accelerating Lateral Movement (RX): 0.00
The incremental change from the cancellation of movement until full stop.

Deceleration on Accelerating Vertical Movement (RY): 0.00
The incremental change from the cancellation of movement until full stop.

Jumping & Falling Parameters
Speed: 6.00
Initial value (0 = no jumping)
Gravity: 0.25

Now we need to create our Penguin gadget. Hit that “Create” option one more time! This will pop-up a new window where you can select the “Animation Assignments”. Obviously we have to choose Penguin. This creates Gadget001 with an Action001 which you can rename to Penguin and Walking Right respectively. Go ahead and make new “Action Programs” for walking left, dying right, dying left, and invisible. Walking right should already have the items under “Action to be Assigned” set properly, so just change the “Movement Direction” dial to the right. Set up Walking Left the same way, only of course the dial should be to the left. Then set up the animations for “Action to be Assigned” for the rest of the “Action Programs”. For both dying directions and invisible you won’t need to designate a direction of movement so check that box, and set your “Movement Speed” to 0 for all three.

Now head back to the “Emergence Settings” sub-sub-tab and choose layer 3, make “The First Action Program” walking right, and under “Activation” select “Constantly Active”. This will allow the penguin to exist even when it isn’t on the screen. Under “Others” check the box for “Don’t Flash On Taking Damage”. Go back to the “Registering Actions” sub-sub-tab and we’ll need to set up how the penguin moves.

Because we’ve done this before with the Player gadget I’m going to give a short-form version of how to make each Action Program interact with each other.

Walking Right > Walking Left IF “Gadget Collided With A Wall”
Walking Right > Dying Right IF “Got Stepped On”
Walking Left > Walking Right IF “Gadget Collided With A Wall”
Walking Left > Dying Left IF “Got Stepped On”

Dying Right > Invisible IF "Gadget Played All Animations"
Dying Left > Invisible IF "Gadget Played All Animations"

So our penguin will switch directions if it runs into a wall, and it dies if you jump on its head. Simple right? There are just a couple more things we need to fix before we're done. Select dying right or dying left and under "The Selected Motion Program's Settings" scroll down until you see a check box next to "Switch memories". Check that and in the drop-down menu find the memory named "Forced Player Jump". Just below the drop box select the "Add" option, and in the number field just below and to the left change it from 0 to 3 (in this case the number determines the height of the jump). Do this for both dying animations Now select invisible and scroll down until you find a check box next to "Disappear" and make sure it gets checked.

That's it. We now have a Player gadget that can stomp on a Penguin's head to make it die and disappear. Let's head back over to the "Canvases" sub-tab real quick to place the new gadgets! Once you're on the canvas select layer 3 (or whatever layer you're using) and select the "Gadgets" tab under "Layout Parts". You'll see a list containing Text, Camera, Course, Player, and Penguin. Put the player anywhere you like and throw down a few penguins by selecting the gadget and simply placing it inside an open spot on the grid. Save your progress if you haven't done so already and hit play.

When you test play your keys will be different than the "B" button we set up earlier. Here's a quick guide to what keys correspond to which buttons by default.

Left Trigger = Ctrl(L)
Left Bumper = Shift(L)
Back/Select = Tab
Start = Space
Right Trigger = Ctrl(R)
Right Bumper = Shift(R)
X = A
Y = W
A = X
B = D

Up/Down/Left/Right are simply the arrow keys. This can be confusing at first but it is easy to get the hang of with a little play testing. If all goes well you should be able to use our Export To Flash tutorial - http://blog.rpgmakerweb.com/tutorials/igm-export-flash/ - to turn this project into this: http://blog.rpgmakerweb.com/wp-content/uploads/2012/02/Jumping-action_1.swf

That's it for now! Feel free to experiment and try different things. If you have any ideas or questions for future tutorials feel free to leave comments or send me emails. I would love to hear everyone's thoughts.

Ready to get started with Indie Game Maker? Use the coupon code "**igmakerohyeah**" at check out to get 10% off your purchase of IGM.