

# THE TECHNOLOGY AND CRAFT OF COMPUTER GAME DESIGN

An introductory course in computer game design

TUTORIALS, GRAPHICS, AND COURSEWARE BY:

**MR. FRANCIS KNOBLAUCH**

TECHNOLOGY EDUCATION TEACHER  
CONWAY MIDDLE SCHOOL  
ORLANDO, FLORIDA

## GAME BUILDING TUTORIAL FOR THE GAME

# LAZARUS

\*BASED ON THE GAME CREATED BY:

**JACOB HABGOOD & MARK OVERMARS**

\*INCLUSIVE OF GAMEPLAY, GRAPHICAL/SOUND ASSETS, AND PROGRAMMING CONCEPTS

GRAPHIC ASSET ILLUSTRATIONS BY:

**KEV CROSSLEY**



**TUTORIAL AND COURSEWARE DOCUMENTS INCLUDE:**

*Lazarus: Stages 1 & 2*

*Lazarus: Stages 3 & 4*

*Lazarus: Stages 5, 6 & 7*

**STUDENT RECORD DOCUMENTS INCLUDE:**

*Lazarus: Galactic Mail: Stages 1 & 2*

*Lazarus: Galactic Mail: Stages 3 & 4*

*Lazarus: Galactic Mail: Stages 5, 6, & 7*

**COMPANION MATERIALS INCLUDE:**

*Glossary*

*Concepts Explained*

**REQUIRED SOFTWARE OR GAME ENGINE:**

*Game Maker 8.1 or Game Maker Studio*

**REQUIRED DIGITAL ASSETS:**

\*Lazarus Assets

ALL TUTORIALS AND REFERENCE RESOURCES FOR THIS COURSE ARE THE PROPERTY OF THE AUTHOR.

USE OF THIS MATERIAL WITHOUT PERMISSION FROM THE AUTHOR IS PROHIBITED.

THIS COURSE IS IN AN ALPHA STAGE.

TUTORIALS AND COURSEWARE ARE PENDING COPYRIGHT.

## Lazarus: Stages 1 & 2

The basic vocabulary and concept of *animation* using *image* and *sub-image* assets was introduced in *Galactic Mail*. How do we apply these concepts in a creative sense to add value to an animation?

"When people laugh at Mickey Mouse, it's because he's so human; and that is the secret of his popularity."

-Walt Disney

This observation by Walt Disney expresses a core concept of animation that is shared in a variety of entertainment industries, including film, television, and video games. Does a mouse really sing and dance? Do bugs have big eyes and toothy smiles? If you believed in ghosts, would you expect them to be friendly and childlike? Animation can also make objects that exist in the real world take on life, like dancing cups and saucers. Events and actions that could never happen in the real world can become believable. Imagine a coyote running off the edge of cliff, pausing to hover in mid-air long enough to show expression of fear. Before falling and waving goodbye, before stretching and falling hundreds of feet to his demise.

Walt Disney called this artistic theory the "*plausible impossible*". He referred to the appearance of reality within situations and characters that would never exist in real life. The viewer becomes comfortable watching characters and situations that would never exist naturally. In other words, unbelievable things seem believable, and impossible things seem possible. The word **plausibility** applies here, as to be **plausible** means that things have an appearance of being believable, or true. When an animation is **plausible**, people will be comfortable with it, understand it, and connect to it, even though it is not possible in a real sense. Hence, the importance of Disney's concept of the "*plausible impossible*".

In *Lazarus*, you will be deepening your understanding of animation by creating properties for a character (named Lazarus) that give him lifelike qualities in both his movement and expression. In creating your animated events and actions, Lazarus the grape will behave in a way that makes sense. He will be interacting with realities of the physical world as well, as he will have to avoid the hazards of falling boxes in the warehouse where he dwells. He will need to look, or "check" for empty spaces to dodge into, where he will be safe from catastrophic events that will crush him. Even though he is a grape, his movements will appear to be natural in the way he leans and jumps. He will express joy in being safe, and fear when it is obvious that there is no escaping the falling boxes in the warehouse. The boxes themselves will take on the physical properties of real objects, as they fall due to gravity. Lighter boxes will stack onto boxes that are the same weight or heavier, and of course, a heavy box will crush lighter box. Hence, everything that happens in the gameplay of *Lazarus* will make sense, even though grapes don't really have eyes, facial expressions, and natural movement. Typically, boxes do not fall from the top of a warehouse. The blend of real physical properties, familiar facial expression, and movement all come together to make the gameplay and animation **plausible**.

As you work through STAGES 1 & 2, you can read the concepts (gold box) to learn about how plausibility is achieved through techniques like **squash and stretch**, **anticipation**, **staging**, and **exaggeration**. In STAGES 1 & 2, properties in this game build will include the use of new *conditional actions* including **check collision** and **check empty**, and the continued use of sub-procedures in blocks.

Complete the definitions for the vocabulary on the tutorial guide

**BE SURE TO PAUSE TO COMPLETE HYPOTHESIS AND EVALUATIONS WHEN PROMPTED.**



**DO NOT USE THE TEST BUTTON IN THIS ACTIVITY UNTIL YOU ARE PROMPTED.**

**THERE ARE NO SAVE PROMPTS IN THIS TUTORIAL, SO SAVE REGULARLY.**

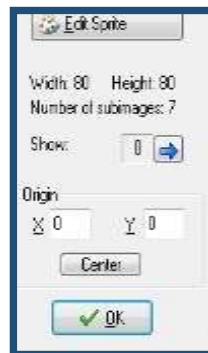
**START HERE BEFORE CONTINUING TO NEXT PAGE:**

Set up Game Maker in **advanced mode**. Save and name your file **initials\_lazarus**.

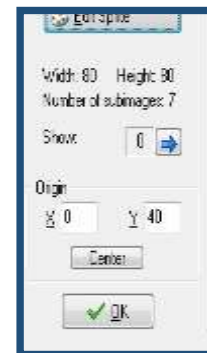
## STAGE 1A:

### Creating the sprite resources with specific origins

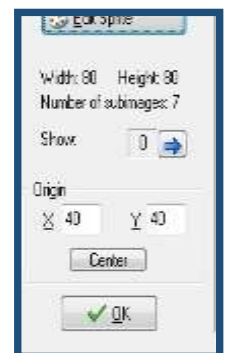
- 1- Create a new sprite called **spr\_laz\_stand** using **Lazarus\_stand.gif** from the **Lazarus Assets** folder on your computer desktop. Click the **OK** button to close the form.
- 2- Create another sprite called **spr\_laz\_right** using **Lazarus\_right.gif**. Under origin, set the **Y** value to **40** (so it shows **X** as **0** and **Y** as **40**).
- 3- Create a **spr\_laz\_jump\_right** with the **Lazarus\_jump\_right.gif**. You also leave **X** value of **0** and set a **Y** value of **40** just as you did with the last sprite.
- 4- Create a **spr\_laz\_left** sprite using **Lazarus\_left.gif**. On this sprite, set both the **X** and **Y** values to **40** (so it shows **X** as **40** and **Y** as **40**) then close the form by clicking **OK**.
- 5- Create a **spr\_laz\_jump\_left** sprite in exactly the same way using **Lazarus\_jump\_left.gif**. Again on this sprite, set both the **X** and **Y** values to **40** (so it shows **X** as **40** and **Y** as **40**) then close the form by clicking **OK**.
- 6- Create two more sprites called **spr\_laz\_afraid** and **spr\_laz\_squished** using **Lazarus\_afraid.gif** and **Lazarus\_squished.gif**. These sprites are the same size as the standing sprite (40 x 40 pixels) so the origin can be left at **0** for **X** and **0** for **Y**.



spr\_laz\_stand  
spr\_laz\_afraid  
spr\_laz\_squished



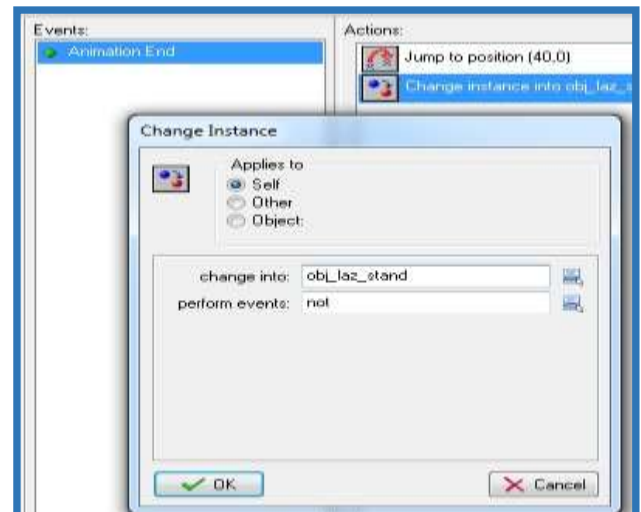
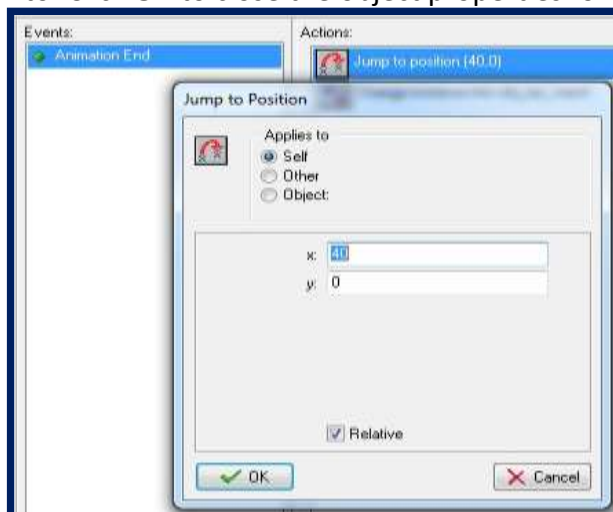
spr\_laz\_right  
spr\_laz\_jump\_right



spr\_laz\_left  
spr\_laz\_jump\_left

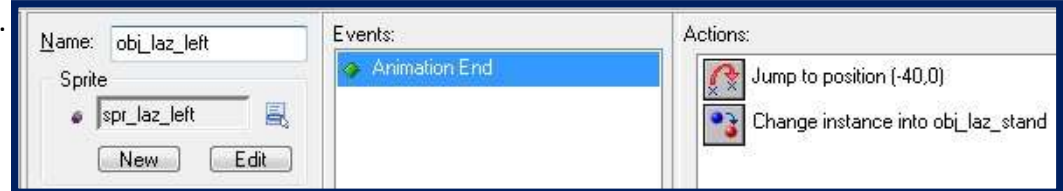
### Creating object resources

- 1- Create a new object called **obj\_laz\_stand**. Assign it **spr\_laz\_stand**, then click **OK**.
- 2- Create a new object called **obj\_laz\_right** and give it the sprite named **spr\_laz\_right**. Don't close the form because you are about to add events and actions.
- 3- **Click Add Event** and select **Other**, then select **Animation End** event. This is the event that will stop the animation when it reaches the last sub-image.
- 4- Drag and drop a **Jump to Position** action in this event from the **move** tab.
- 5- In the Jump to position window, set **X** to **40** and **Y** to **0**, and make sure that the **Relative** option is enabled.
- 6- Add a **Change Instance** action (**main1** tab) below this and select **obj\_laz\_stand** as the object to change back into. Click **OK** to close the object properties form.



**NOTE:** In your first Hypothesis, be ready to anticipate behavior based on these **X** and **Y** settings for all of these Lazarus objects (**hint:** the left and right moves and jumps occur with boxes that are all 40 x 40 pixels. You will need to explain what Lazarus will look like when he is moving, and what he will look like at the end of the movement).

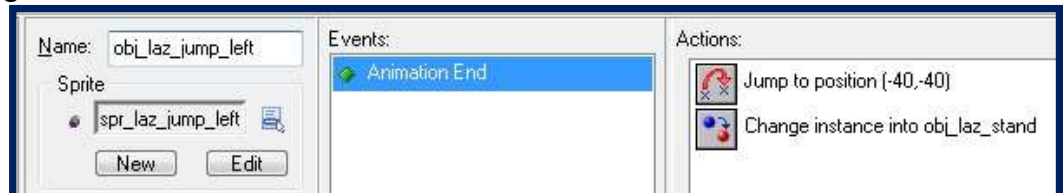
- 7- Create another object called **obj\_laz\_left** and assign it **spr\_laz\_left**. Again, leave the form open because you about to add some events with action.
- 8- Repeat the events and actions as before (steps 4–6), but this time set **X** to **-40** for the **Jump to Position** action in **obj\_laz\_left**.



- 9- Create another object called **obj\_laz\_jump\_right** and assign it **spr\_laz\_jump\_right**. Repeat the steps 4 through 6 process again, but now setting the Jump to Position coordinates to **X** to **40** and **Y** to **-40** for **obj\_laz\_jump\_right**.

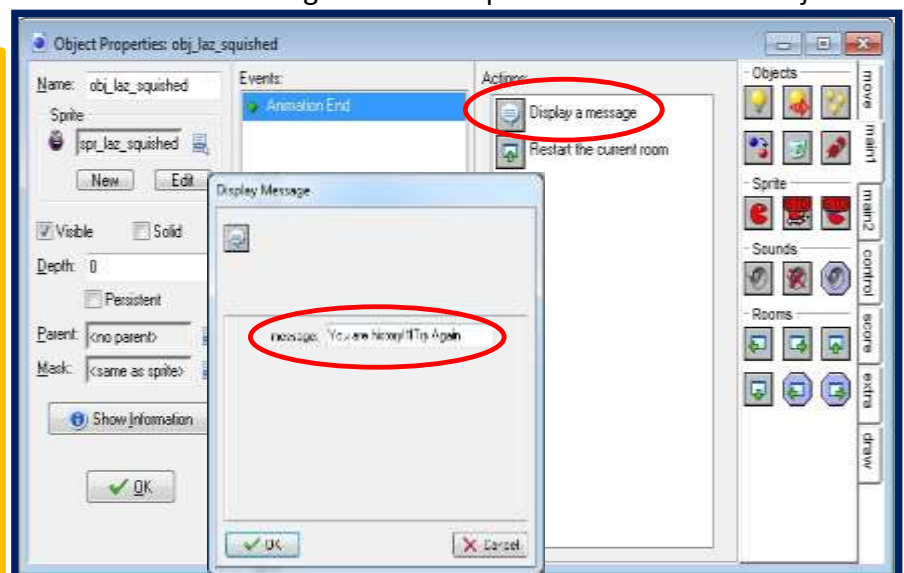


- 10- Create one more Lazarus with **obj\_laz\_jump\_left** and assign it **spr\_laz\_jump\_left**. Do steps 4 through 7 again, this time setting **X** to **-40** and **Y** to **-40**.



## Creating the squished Lazarus object resource

- 1- Create an object called **obj\_laz\_squished** and it **spr\_laz\_squished**.
- 2- Click Add Event and choose **Other**, then **Animation End** event from the list. Drag and drop the **Display Message** action found in the **main2** tab.
- 3- In the message properties, you can type something like **"YOU'RE HISTORY!#Better luck next time"** into the message properties. Don't use the quotation marks. **You should type the #** symbol in the middle of the message so it starts a new line from word. Try to come up with something fun. Another text might be **"BUMMER! YOU'RE SQUISHED# YOU SHOULD TRY AGAIN!"** Just keep it positive.
- 4- Drag in the **Restart Room** action from **main1** after the message action and press **OK** to close the object properties form.



## CONDITIONAL STATEMENTS

### FOR LAZARUS OBJECTS:

**LAZARUS MOVING RIGHT:** IF the animation ends at (40, 0), THEN change into Lazarus standing.

**LAZARUS MOVING LEFT:** IF the animation ends at (-40, 0), THEN change into Lazarus standing.

**LAZARUS JUMPING RIGHT:** IF the animation ends at (40, -40), THEN change into Lazarus standing.

**WRITE MORE CONDITIONAL STATEMENTS  
IN NEXT TUTORIAL GUIDE STEP**



## DO THIS ON YOUR TUTORIAL GUIDE

**STAGE 1A CONDITIONAL STATEMENTS:** Write two conditional statements for the two *Animation End* events not represented in the gold box above. Use x and y coordinates (x, y) in your first statement.

- The first statement is for step 10 under “Creating Object Resources”.
- Note that the second IF/THEN statement applies to Lazarus “squished”.

### STAGE 1B:

#### Adding a right key event for the standing Lazarus object

- 1- Reopen the properties form for the *obj\_laz\_stand* object by double-clicking it in the objects folder or the resource explorer.
- 2- Create a **Key Press, <Right>** then drag and drop **Check collision** action from the control tab.

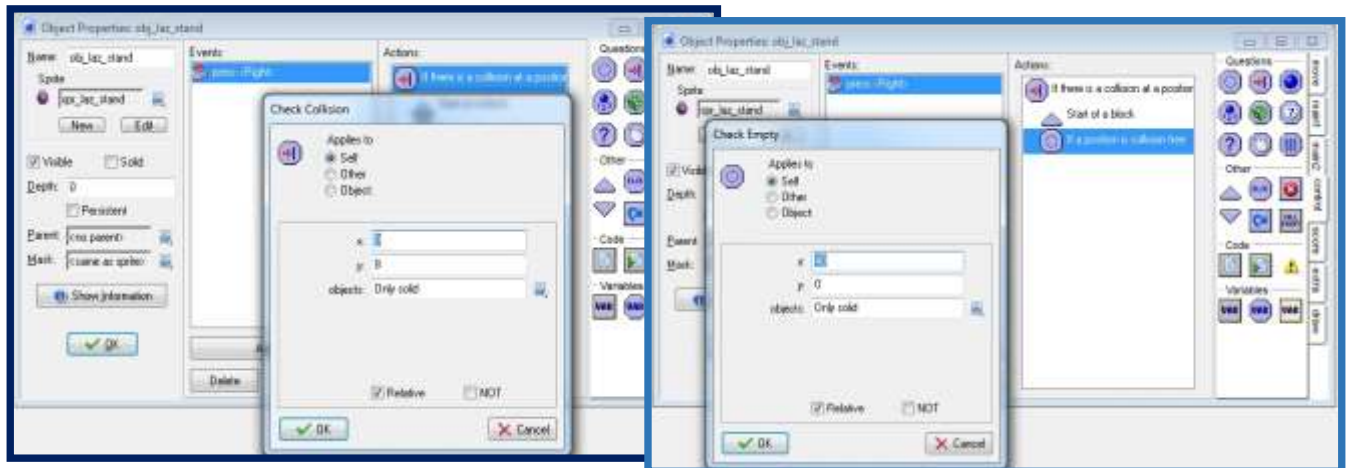
**NOTE:** **Check collision** is a conditional action that will not allow the next action to occur unless it checks as “true”. If the object collides with another object at a specified coordinate, it will check “true”.

- 3- Set **X** to **0** and **Y** to **8** and enable the **Relative** option by checking the box.

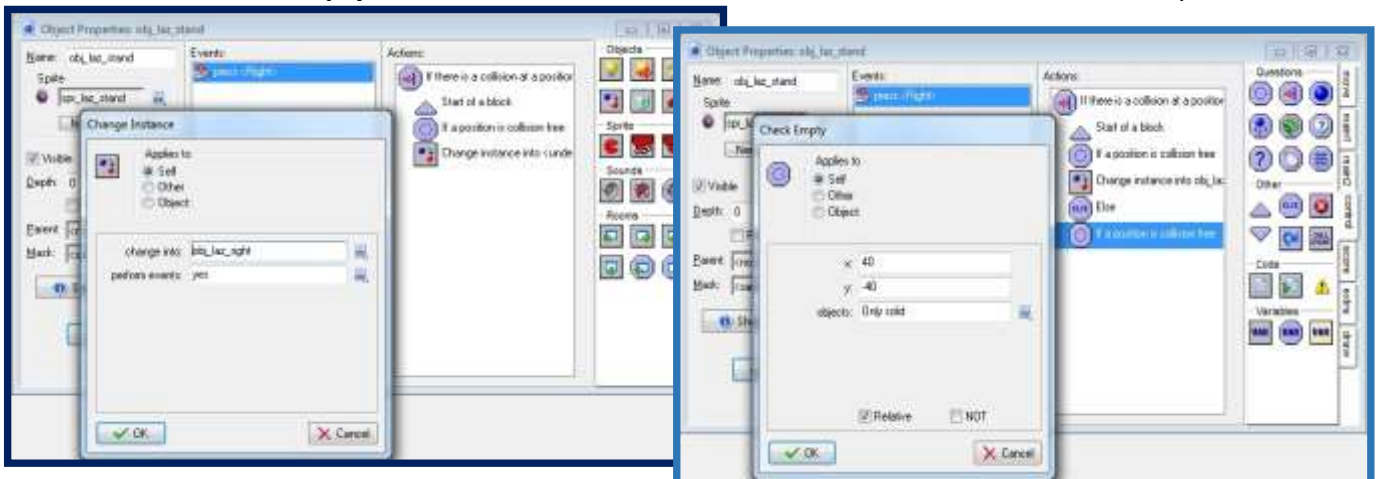
**NOTE:** Coordinates (0,8) move Lazarus slightly so he can sense what is under him. The next block will run as a sub-procedure only when the collision checks as *True*. He'll need to look for somewhere else to go!

- 4- Drag and drop a **Start Block** action.
- 5- Add a **Check Empty** conditional action. Set **X** to **40** set **Y** to **0**, and check the **Relative** option.

**NOTE:** **Check empty** is a conditional action that will not allow the next action to occur unless it checks as “true”. If the object does not collide with another object at a specified coordinate (if the space is “empty”), it will check “true”.



- 6- Add a **Change Instance** action and select the *obj\_laz\_right* object. Select **yes** to **Perform Events**.
- 7- Add an **Else** action from the control tab.
- 8- Add another **Check Empty** conditional. Set **X** to **40** and **Y** to **-40**, and check the **Relative** option.



9- Add a **Change Instance** action and select the **obj\_laz\_jump\_right** object. Select **yes** to **Perform Events**.

10- Finish the block with an **End Block** action.

### LOGIC SYSTEM FOR **CHECK COLLISION** & **CHECK EMPTY**

IF the right key is pressed, THEN check for a collision on right side of Lazarus.

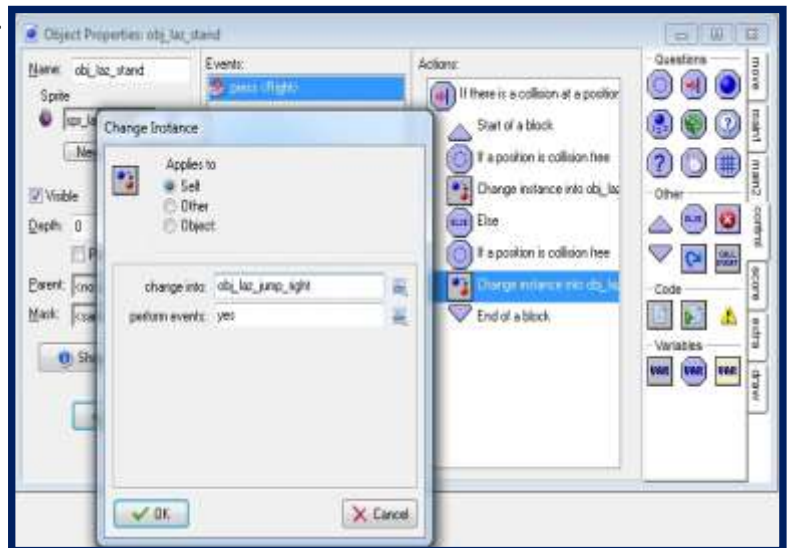
-WHEN THERE IS A COLLISION THE BLOCK STARTS

IF position on right (40, 0) is open (collision free), THEN change into Lazarus right.

-WHEN THE SPACE IS NOT OPEN CONTINUE WITH ELSE

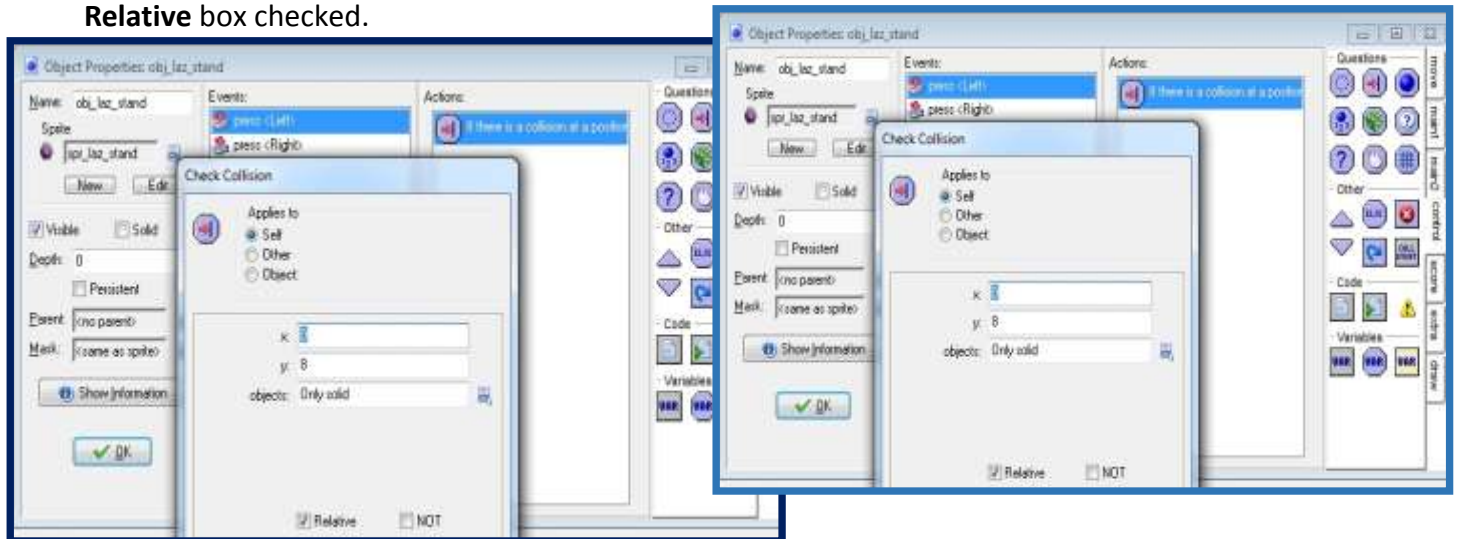
ELSE check for a collision one space above the space to the right (40, -40).

IF that position is open, THEN change into Lazarus jumping right.



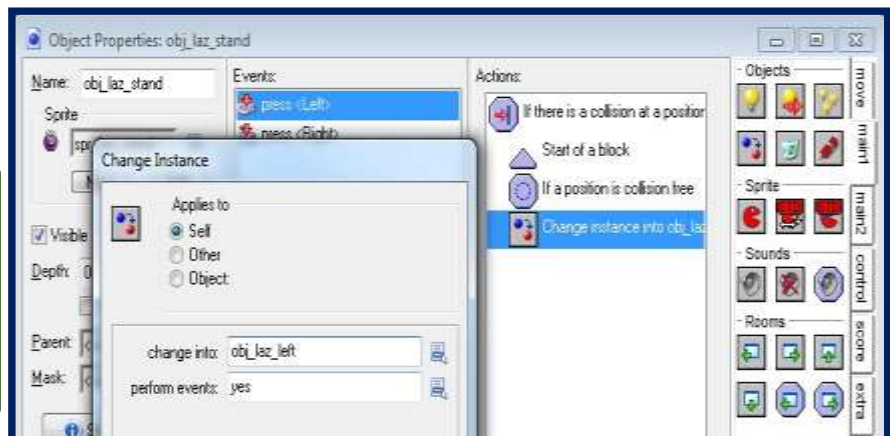
### Adding a left key press event to the standing Lazarus object

- 1- If you closed **obj\_laz\_stand**, reopen it because you are about to add more properties. Add a **Key Press**, **<Left>** event then drag and drop a **Check Collision** action. Set **X** to **0** and **Y** to **8**, and check the **Relative** option.
- 2- Drag a **Start Block** adding it to the actions.
- 3- Add a **Check Empty** conditional action from the **control** tab with **X** set to **-40**, **Y** set to **0**, and the **Relative** box checked.

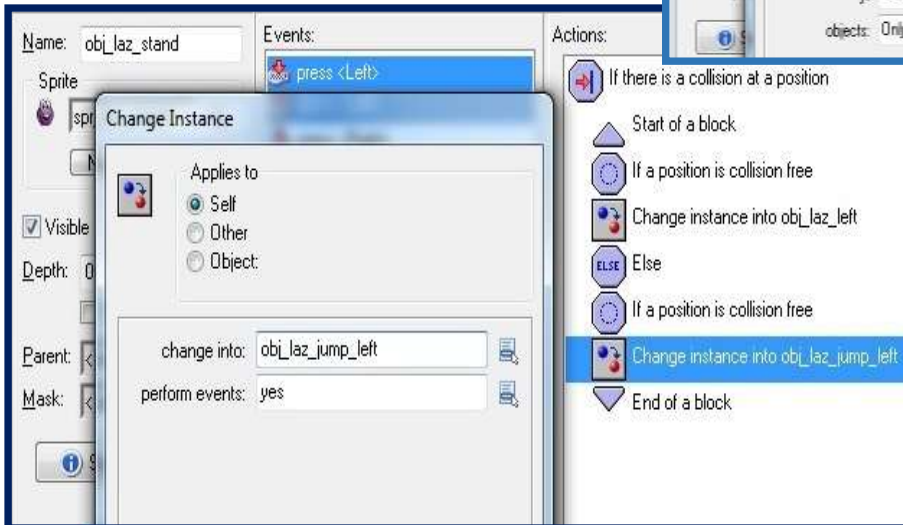


- 4- Add a **Change Instance** action from the **main1** tab. Choose **obj\_laz\_left** from the change into menu. Select **yes** to **Perform Events**.

Pay close attention to properties that you are adding for the left key event. Soon, **you will be instructed to write conditional statements for this logic system.**

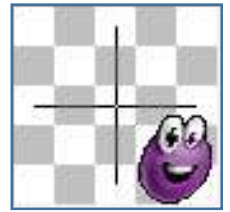


- 5- Add an **Else** action from the **control** tab. Add a **Check Empty** action with **X** as **-40**, **Y** as **-40**, and **Relative** enabled this checks diagonally left. Add a **Change Instance**, choosing **obj\_laz\_jump\_left** and select **yes** to **Perform Events**.
- 6- Add an **End Block** action to finish the block of actions. Do not close this properties form.



### Anticipation

This is the first frame of the Lazarus assets that move to the left or jump to the left. He is leaning to the right, in a position is preparing him to spring back to the left. This creates an element of **anticipation**, so the player gets the sense that the object is getting ready to move either straight to the left, or up to the left. **Anticipation** prepares the position of an object for the action that is about to happen. A dancer bends his knees before jumping. A pitcher takes a stance before winding up to pitch. A character may look off to the direction where it is going to move. These are all examples of moves or expressions that give the player an expectation of natural motion, even if the character and the situation is unnatural.



**Staging** Preparing a scene is a crucial element of creating animation in a gameplay. Putting the focus on the main character and preparing it for action is called **staging**. In the *Lazarus* game, it is clear that the focus of attention is on the most important object in the scene. Lazarus is colorful, has a joyful smile, and expressive eyes. The Lazarus standing object **stages** the character for events and action that will occur, putting him in center of gameplay.



### DO THIS ON YOUR TUTORIAL GUIDE

**STAGE 1B CONDITIONAL STATEMENTS:** Write a series of conditional statements showing the **logic system** for the left key events. You should use **IF**, **THEN**, and **ELSE** in the statements. Use **x** and **y** coordinates in statements (**x**, **y**).

- You should refer to the gold concept box **LOGIC SYSTEM FOR CHECK COLLISION & CHECK EMPTY** as a reference. This is found on the previous page.

### DO THIS ON YOUR TUTORIAL GUIDE

**STAGE 1A & 1B HYPOTHESIS:** Now it's time to predict the behaviors of the Lazarus objects based on the properties applied to the sprite and objects.

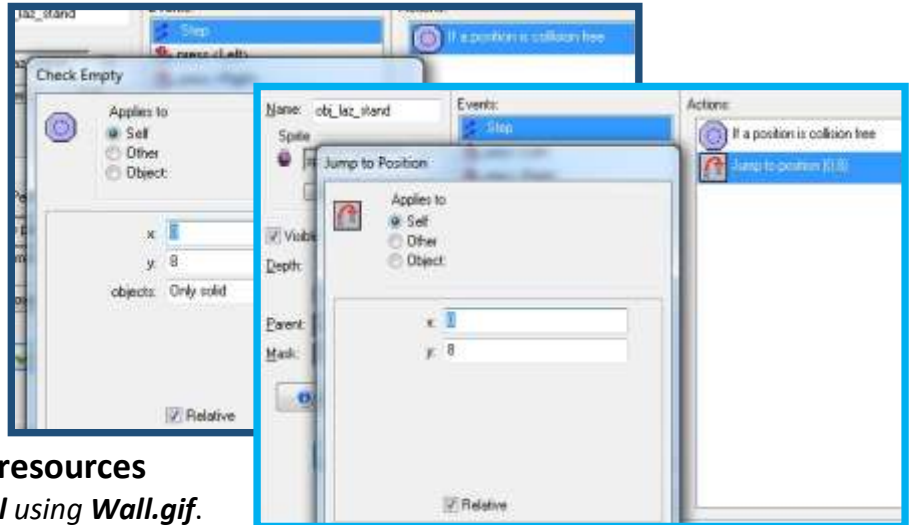
- Consider where Lazarus will move based on the presence of objects to his left or right.
- Note that you will not be able to test the validity of your hypothesis until the end of STAGE 2.



## STAGE 2:

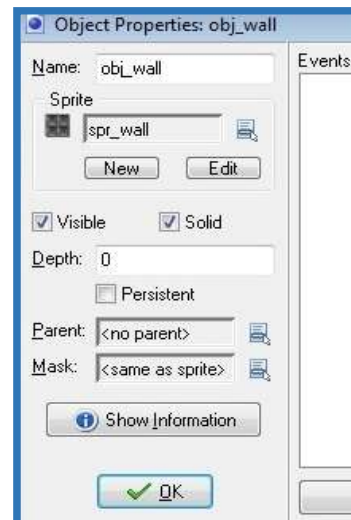
### Adding a step event to the standing Lazarus object to make it fall

- 1- Create a **Step** event to **obj\_laz\_stand**, selecting **Step** from the drop down that appears.
- 2- Add a **Check Empty** action in the **Step** event, setting **X** to **0** and **Y** to **8**, and checking the **Relative** option.
- 3- Add a **Jump to Position**. Set **X** to **0** and
- 4- **Y** to **8**, and check the **Relative** option.
- 5- Click **OK** to close the properties form.



### Creating the wall sprite and object resources

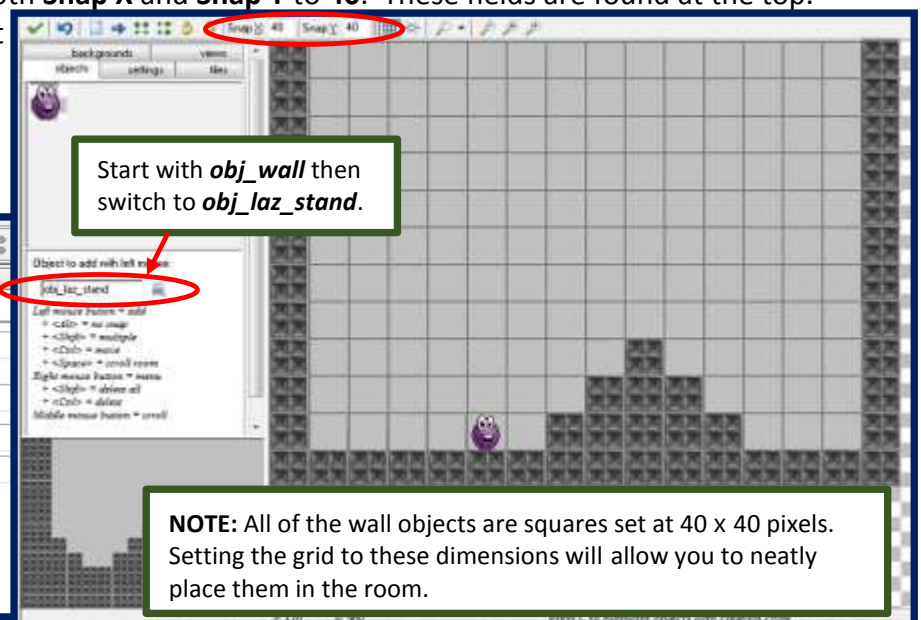
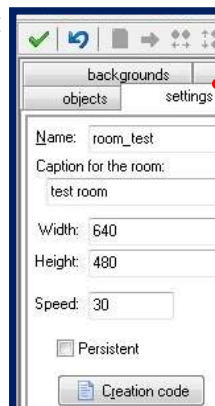
- 1- Create a new sprite called **spr\_wall** using **Wall.gif**.
- 2- Create a new object called **obj\_wall**,
- 3- assigning it **spr\_wall**. Enable the **Solid** option.



In later stages, some wall blocks will be replaced by boxes in the room. For now, the wall blocks will be an easy object to use for testing your hypothesis and programming of properties for the Lazarus objects.

### Creating a Test Room

- 1- Create a new room called **room\_test**. Click on the **settings** tab and write a caption (like "test room").
- 2- In the Room Properties form, set both **Snap X** and **Snap Y** to **40**. These fields are found at the top.
- 3- Switch to the **objects** tab and select **obj\_wall** under "Object to add with left mouse". Create a room with boxes that form flat areas and staircases.
- 4- Change the "Object to add with left mouse" to **obj\_laz\_stand**. Add an instance of the standing Lazarus.



**NOTE:** All of the wall objects are squares set at 40 x 40 pixels. Setting the grid to these dimensions will allow you to neatly place them in the room.



## DO THIS ON YOUR TUTORIAL GUIDE

**STAGE 2 HYPOTHESIS STATEMENTS:** Now it's time to predict the behaviors of Lazarus as you expect to see them in the room you created with wall objects.

- Is your hypothesis the same as it was at the end of STAGE 1? If not, explain why it changed.

Now it's time to test your game, so go ahead and click on the green triangle ( ▶ ) on the menu bar to run the game normally.

## DOES THE ACTION THAT YOU SEE AND THE CONTROL OF THE OBJECTS MEET YOUR HYPOTHESIS STATEMENTS?

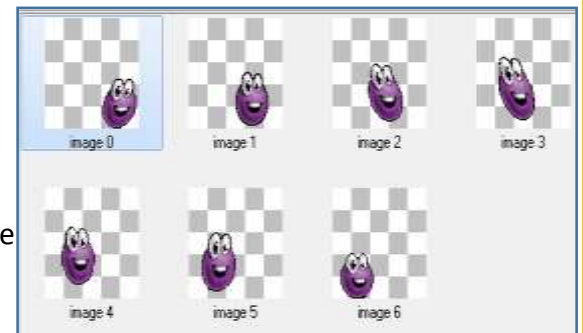
As you move Lazarus left and right using the arrow keys, you should see Lazarus move into open spaces on his left or right, or jump onto the wall blocks that on his left or right if they are present. If you configured the wall blocks to look like steps, Lazarus should be able to climb them. The GIF files used to make the Lazarus sprites and objects were designed to show a natural looking movement. When Lazarus moves, you will see him lean in the direction he is going. When he jumps, you see him squish down like a rubber ball before and after he jumps. He has a happy expression on his face because he is content to be able to move freely through this room. The creators of the Lazarus GIF assets understood that this effect will make the movement appear to be natural. The movement should make sense to the player, even though this would never happen in real life. The notion of a happy grape bouncing around seems **plausible**, doesn't it?

## DO THIS ON YOUR TUTORIAL GUIDE

**STAGES 1 & 2 TEST & EVALUATION:** Write an evaluation of your hypothesis and programming properties from STAGES 1 & 2.

- If you state "valid", provide a detailed explanation of "why" the object behaves properly based on your hypothesis, and the events and actions that you programmed.
- If you state "invalid", make sure that you expose your errors in reasoning, or your errors in programming.

**Squash and Stretch and Exaggeration** All of the moving Lazarus objects apply the **squash and stretch** principal of animation. Characters and objects are shown to "squash" (become distorted or flattened) and "stretch" (become elongated) from motion and other natural forces like gravity. The object changes shape from frame to frame to give the sense that forces are acting on it, not unlike a ball squeezing or compressing when it hits the ground, only to stretch as moves away from the floor. In animation or game design, the look of these behaviors is often **exaggerated**, drawing more attention to the nature of the motion, while keeping a sense of natural movement for the character. In other words, it looks **plausible**. **Exaggeration** is used in animation, often just to make sure that the player gets the idea of what is happening. Sometimes it is simply to enhance the character and give it personality. Lazarus's big "google" eyes and joyful smile are out of proportion with the rest of his body. Hence, **exaggeration** is the enlargement of character, or object features beyond what they would be in reality. Study the sub-image frames of the object for Lazarus moving left. Look for the way in which both **squash and stretch** and **exaggeration** are used.



**CONTINUE ON NEXT PAGE FOR IMPORTANT TUTORIAL GUIDE INSTRUCTIONS**

## DO THIS ON YOUR TUTORIAL GUIDE

**STAGES 1 & 2 CONCEPT SUMMARY:** Review the concepts found in the gold boxes throughout the tutorial. Write a full paragraph describing how animation concepts are used in the behaviors of the Lazarus objects to achieve **plausibility**. Use new vocabulary correctly in your writing.

- Include concepts such as **staging**, **anticipation**, **squash and stretch**, and **exaggeration**.

**END OF STAGES 1 & 2. REVIEW YOUR WORK ON THE TUTORIAL GUIDE. STUDY THE TUTORIAL GUIDE RIGOR SCALE. MAKE REVISIONS AND IMPROVEMENTS BASED ON THE SCALE. UPLOAD COMPLETED TUTORIAL GUIDES TO EDMODO.**