Directions for Lazarus Game

Directions from the Game Maker's Apprentice book by Jacob Habgood and Mark Overmars

Game Description

Lazarus has been abducted by the Blob Mob, who are intent on bringing this harmless creature to a sticky end. They've imprisoned him at the Blobfather's factory, where they are trying to squish him under a pile of heavy boxes. However, they've not accounted for Lazarus's quick thinking as the boxes can be used to build a stairway up to the power button to stop the machinery. Do you have the reactions needed to help Lazarus build his way up or will the mob claim another victim?

Each level traps Lazarus in a pit of boxes stacked on either side of the screen to keep him within the level. The arrow keys will move him left or right, and he will automatically jump onto boxes that are in his way. However, he can only jump the height of a single box, and stack two or more boxes high and his path is blocked. New boxes will periodically appear directly above Lazarus's current position and fall vertically down from the top of the screen until they come to rest. This mean that the player will be able to use Lazarus's position to control when boxes fall and build a stairway up to the power button.

There will be four different types of boxes, increasing in weight and strength: cardboard, wood, metal and stone. Falling boxes will come to rest on boxes that are stronger than them, but will crush boxes that are lighter. The type of each box is chosen at random, but the next box to drop will be shown in the bottom left corner of the window just before it is released. There will be a number of increasingly difficult levels, with higher stairways to build, and boxes that fall faster. When Lazarus gets squished, the level will restart to give the player another try.

We will be working with an animated character, Lazarus, who jumps, moves and gets squished. We need to account for these different variations by having multiple sprites of Lazarus and corresponding objects. We also need to make sure that all of the animations fit within the confines of the room. The boxes are 40x40 and when boxes are stacked, it becomes 80x80. We will need to make sure that the origins are at the same position, regardless of which version of Lazarus we are working with. Follow the step directions very closely and it will make more sense.

Creating the Lazarus sprite resources for the game:

1. Create a new sprite called sprite_laz_stand using Lazarus_stand.png. This sprite is 40x40. Origins for sprite defaults to the top-left corner (with X and Y set to 0). Leave these settings alone and click **OK** to close the form.
2. Create another sprite called sprite_laz_right using Lazarus_right_strip7.png. This sprite is 80x80 so we need to modify the settings. Change the Y value to 40 and click **OK**.
3. Create a Sprite_laz_jump_right using Lazarus_jump_right_strip7.png. X =0 and Y=40.
4. Create a sprite_laz_left using Lazarus_left_strip7.png. Set X and Y to 40 and close.
5. Create a sprite_laz_jump_left using Lazarus_jump_left_strip7.png. X=40 and Y=40
6. Create 2 more sprites: sprite_laz_afraid and sprite_laz_squished using Lazarus_afraid_strip10.png and Lazarus_squished_strip11.png. These do not require the X and Y settings to change.

First we will deal with the normal or standing Lazarus. We need to create objects for each of these sprites after getting the standing Lazarus set up.

Creating Lazarus object resources for the game:

1. Create a new object called object_laz_stand using the standing sprite.
2. Press **Ok** to close the form.
3. Create a new object called object_laz_right using the Lazarus_right sprite.
4. Add an **Other**, **Animation** end event.
5. Include a **Jump to Position** action (**move** tab). Set X to 40 and Y to 0 and **Relative** is enable.
6. Include a **Change Instance** action (**main1**) and choose object_laz_stand.
7. Click **OK** to close the form.
8. Create another object called object_laz_left and use the Lazarus_left sprite. Repeat step 4 and 5 with the X being set to -40 in the **Jump to Position** action. Then follow steps 6-7.
9. Create an object for object_laz_jump_right and use the sprite Lazarus_jump_right. Repeat the same process as above but the X=40 and Y=-40 in **Jump to Position** action.
10. Add a final object called object_laz_jump_left and use sprite Lazarus_jump_left. Repeat the steps with X=-40 and Y=-40 in **Jump to Position** action.

Creating the squished Lazarus object resource:

1. Create an object called object_laz_squished using Lazarus_squished sprite.
2. Add an **Other**, **Animation** End event and include **Display Message** action (**main2**).
3. Type “You’re history!!#Better luck next time” into the message properties. The # starts a new line.
4. Finally include a **Restart Room** (**main1**) after message action and click **OK** to close the properties.

Now we go back to working with the standing Lazarus object.

Adding a right key event for standing Lazarus:

1. Reopen the properties for object_laz_stand.
2. Add a **Key Press**, **<right>** event and include **Check Collision** action (**control**).
3. This will check that there would be a collision if we moved Laz to a particular position. We have to make sure that Lazarus is on solid ground before letting him move. Set X to 0 and y=8 and enable **Relative**.
4. Create a **Start Block**.
5. Include a **Check Empty** conditional action. This action checks that there wouldn’t be a collision if we move to a particular position. Set X=40 and Y=0 and enable **Relative**.
6. Include a **Change Instance** action (**main1**) and choose object_laz_right. Select **YES** to **Perform Events**. Perform events control whether the Destroy and create event will be called.
7. Include an E**l**se action from **Control** tab.
8. Include another **Check Empty** action after this. We are checking that no boxes are diagonally, up and to the right of Lazarus. Set X=40 and Y=-40 and Relative is **enabled**.
9. Include a **Change Instance** action and choose object_laz_jump_right. Select Y to **Perform events**.
10. Include an **End Block** action. This will conclude that all actions should be performed if Lazarus is on solid ground.

Adding a left key press to the standing Lazarus object:

1. Add a **Key Press <left>** event and include a **Check Collision**. Set X=0 and Y=8 and enable **Relative**.
2. Include a **Start Block**.
3. Include a **Check Empty** action (**control**) with X=-40, Y=0 and **Relative** enabled.
4. Include a **Change Instance** action (**main1**) and choose object_laz-left. Choose **Yes** to **Perform Events**.
5. Now include **Else** from the control tab.
6. Include a **Check Empty** with X=-40, Y=-40 and **Relative** enabled.
7. Include a **Change Instance** action and select object_laz_jump_left. Choose **Yes** to **Perform Events**.
8. Include an **End Block**.

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

1. Add the **Step**, **Step** to standing Lazarus object.
2. Include a **Check Empty** action setting X=0 and Y=8. Enable **Relative**.
3. Include a **Jump to Position** action. Set X=0 and Y=8 with **Relative** enabled.

Creating a wall object resource for the game:

1. Create a new sprite called sprite_wall using wall.png.
2. Create a new object called object_wall. Enable Solid.
3. Create a new room call room_test and create a caption on the settings tab.
4. Set the Snap X and Snap Y to 40. This makes all of the boxes 40x40.
5. Switch to the objects tab and select the wall object. Create a level with a number of boxes to create flat areas and staircases. You can use the Shift key to add multiple instances. Add one instance of the standing Lazarus object.
6. Test the game. If something doesn't work, go back through the steps, making sure Relative is enabled where required. **SAVE YOUR WORK**!!

Now we will move to working with the falling boxes. They will be chosen at random and the heavier boxes will crush the lighter ones. To give the player a way to think ahead, the next box to drop will be shown in the corner of the screen as the current box is falling.
Each box will have 3 behaviors: 1- appearing in the corner as the next box, falling and landing on another box and creating an obstacle for Lazarus to get around. We will need to create 3 different objects for each box—one for each behavior. We start with the stationary boxes and creating more sprites.

Creating new box sprite and object resources for the game:

1. Create sprites called sprite_box_stone, sprite_box_metal, sprite_box_wood, and sprite_box_cardboard using Stonebox.png, Metalbox.png, Woodbox.png and Cardbox.png.

Create a new object called object_box_stone using the stone sprite. Set as **Solid**.

2. Repeat the steps for the other boxes, naming them object_box_metal, object_box_wood and object_box_cardboard.

Creating falling box objects for the game:

1. Create a new object called object_falling _stone using the stone box sprite. Select S**o**lid.
2. Add a **Create** event and include a **Jump to Position** action. Type the variable object_laz_stand.x into X and Y=-40. This makes the box start to fall above Lazarus.
3. Include a **Move Fixed** action with a downward direction and **Speed** of 5.
4. Add a **Collison** event with object_laz_stand and include a **Change Instance** action. Change Applies to option to **Other**. Select object_laz_squished and select **Yes** to **Perform Events**.
5. Add another **Collision** event, this time with object_wall. This will stop the box and include a **Move Fixed** action with the **middle square** pressed and a **Speed** of 0. Include a **Change Instance** action and select object_box_stone.
6. Add a third **Collision** event with object_box_stone and include the same two action as the Collision event with the wall (these can be copied).
7. Add a fourth **Collision** event with the object_box_metal. Metal is lighter than stone so it must be crushed. Include a **Destroy Instance** action and select **Other** object.
8. Add fifth and sixth **Collision** events with object_box_wood and object_box_cardboard, including **Destroy Instance** actions like in step 7 to destroy the **Other** box in the collision.
9. Create the remaining three falling objects for the other types (object_falling_metal, object_falling_wood, and object_falling_cardboard). Repeat steps 1-8 for each one using step 7 when a box crushes another box and step 5 when a box stops.

Material	Material(s) That It Crushes
Stone	Metal, Wood and Cardboard
Metal	Wood and Cardboard
Wood	Cardboard
Cardboard	None

Creating next box object resources for the game:

1. Create a new object called object_next_stone, assign the stone box sprite and make it **Solid**. Click **OK**.
2. Create objects for object_next_metal, object_next_wood and object_next_cardboard.

Now we have to create a controller. A controller object is an invisible object without a sprite which performs important actions on other objects. Our controller will use a Step event to check if there is a falling box. If not, then it will turn the current box in to a falling box and create a new next box It will run until the level is completed or the player gets squished.

Creating a controller object resource for the game:

1. Create a new object called object_controller and leave it without a sprite.
2. Add a **Step**, **Step** event and include the **Test Instance Count** conditional action (**control**). This counts the number of instances of a particular object and tests it against a value. Choose object_falling_stone; leave the **Number** as 0 and **Operation** as Equal to.
3. include three more **Test Instance Count** conditional actions to check if there are no instances of object_falling_metal, object_falling_wood, and object_falling_cardboard.
4. Include a final **Test Instance Count** action for the object_laz_stand but set the **Number** to 1 and **Operation** to Equal to. This checks to see if there is a standing Lazarus in this level.
5. Include a **Start Block** action. This will group the actions to create the new box.
6. Include a **Change Instance** action and select **Object** for applies to, so it changes all instance of one object on the level into another. Set **Object** to object_next_stone, Change into object_falling_stone, and select **Yes** to **perform events**. This turns any stone next boxes into stone falling boxes.
7. Add 3 more **Change Instance** actions to change object_next_metal into object_falling_metal objects, object_next_wood into object_falling_wood, and object_next_cardboard into object_falling_cardboard.
8. Include a **Create Random** action (**main1**) and select four different next box objects. Set X=0 and Y=440 leave **Relative** disabled. If Relative is disabled, X and Y are measured from the top-left corner of the screen. With these coordinates, you are putting the new next box where it should be in the bottom left corner of the screen.
9. Include an **End Block** action to close the block of actions that are dependent on all conditions being true.

Editing the test room to add new instances:

1. Reopen the test room
2. **Remove all extra wall instances** so it is just a pit with walls on both sides and across the bottom.
3. Add **one instance of the controller object** into the room. It will show up as a blue ball with a red question mark. This won't appear in the game but lets us know it is there when we are editing the room.

Now run the game and test it. Make sure that the box appears in the bottom left is the box that falls down the screen next and check that heavier boxes are crushing lighter ones.

Finishing touches..creating an exit to the next level. As we left off, there was no way out of the room. We need to create a way for Lazarus to avoid being trapped.

Editing the standing Lazarus object to detect for being trapped:

1. Reopen the standing Lazarus object and select the Step event.
2. Include the **Check Collision** conditional action (control) below the last action in the list. Set X=40 and Y=0 and enable **Relative**.
3. Include another **Check Collision** with X=40 and Y=-40 and R**e**lative enabled.
4. Include 2 more **Check Collision** actions: 1 with X=-40 and Y=0 and the other with X=-40 and Y=-40. **Relative** should be enabled on both.
5. Include a **Change Sprite** action using afraid Lazarus sprite. This will only happen if the 4 conditions above are true and Lazarus is boxed in.

Editing the standing Lazarus object to detect for being freed:

1. Select the **Step** event for the standing Lazarus.
2. Include a **Change Sprite** at the very beginning of the list of actions. Set it to change into standing Lazarus sprite.

Adding a goal. All good games must have a goal so players don't get bored. The goal here is to reach one of the stop buttons so the machinery stops and the boxes stop dropping. If the player reaches the end and there are no more rooms, the completion message must be seen and the ability to restart the game made available.

Creating a new button object resource for the game:

1. Create a new sprite called sprite_button using Button.png.
2. Create a new object called object_button using the button sprite. Set **Depth** to 10 so it appears behind other objects.
3. Add a **Collision** event with the standing Lazarus and include a **Sleep** action (**main 2**). Set **Milliseconds** to 1000 and **Redraw** is true.
4. Include a conditional **Check Next** action (**main1**).
5. Include a **Next Room** action (**main1**).
6. Include an **Else** action followed by the **Start Block** action
7. Include a **display Message** action (**main2**) and set Message to "Congratulations#You have completed the **game!"**
8. **Include a Different Room** action and set **New Room** to first room
9. Finally, include an **End Block** actions and close properties form.
10. Edit your test room and add a **stop button** on either side of the top of the pit.

Starting a level

Creating a new starter object resource for the game:

1. Create a new sprite called sprite_title using Title.png.
2. Create a new object called object_started using the title sprite.
3. Add a **Create** event and include a **Sleep** action. Set **Milliseconds** to 2000 to wait 2 seconds.
4. Include the **Change Instance** action and select the controller object. Close the properties.
5. Edit your test room, **remove the controller object** using the right mouse button. **Add** the **starter object** at an appropriate place.

Sounds, Backgrounds and Help

1. Add **sounds** for Music.mp3, Wall.wav, Crush.wav, Squished.wav, Move.wav and Button.wav. and play them at the right times using the **Play Sound** action (**main1**).
2. Create a **Game Start** event in **Other** events. Set **Loop** to true in the **Play Sound** action.
3. Add crush and wall sound effects to the existing **Collision events** between the falling box objects and stationary box objects.
4. Add a new **Create** event for the squished lazarus object and play the squished sound effect. This will save you work to put it in each of the four collision events for falling boxes and Lazarus.
5. Add **Create events** to play the move sound effects for the four moving Lazarus objects.
6. Play the button sound effect in the Collision between the button and Lazarus.

Test the game and make sure the sound effects are working and playing when needed. If you don't hear a sound when moving around, check that you set Perform Events to Yes in the Change Instance actions to change into the animating objects.

Creating a background resource and Game information:
Create a **background** using background.png from the folder.

1. Reopen the properties form for the room and select the backgrounds tab. Choose the new background from the menu.
2. Double click the **Game Information** and add a help text for the game. Remember to include the name of the game, who created it and a description of the aims and controls.

Levels
We will talk more about this later but you should have some pits and buttons on each side to keep things fairly easy. As the levels progress, make the pits higher and that will make the levels harder. By increasing the speed, the challenge will increase. Duplicating the rooms wil save a lot of work. Remember you can right click and choose Duplicate. One other thing to make it helpful is to add some cheats to allow you to skip levels.
Editing the controller object to add cheats:

1. Open the properties form for the controller object.
2. Add a **Key Press <N>** event and include the **Next Room** action.
3. Add a **Key Press <P>** event and include the **Previous Room** action.

Good luck and remember—these cheats will be removed before the game is finally finished. You might want to add an opening and closing screen or a scoring system so players can compete for high scores.