



OBJECTIVES:

- Students will review strategies and purposes for critiquing artwork, and critique several digital games as works of art.
- Students will be introduced to a variety of simple games by a diverse historical and contemporary creators from the commercial and independent spheres.
- Students will be introduced to Game Maker, a powerful user-friendly game design tool, and learn how to use it to implement several elementary game mechanics.
- Students will create their own simple game, applying the values and concepts derived from critique of exemplar games, and the skills developed in the Game Maker tutorials.

NATIONAL ARTS SOLs:

NA-VA.5-8.2 Students generalize about the effects of visual structures and functions and reflect upon these effects in their own work.

NA-VA.5-8.5 Students analyze contemporary and historic meanings in specific artworks through cultural and aesthetic inquiry.

NA-VA.9-12.2 Students create artworks that use organizational principles and functions to solve specific visual arts problems.

VA ARTS SOLs:

5.9, 7.4, AI.6, AII.5 The student will use contemporary media to create works of art.

5.20, 7.15, 8.18, AI.15, AII.17 The student will use specific criteria and criticism processes to evaluate theirs & others' work.

6.19 The student will explain the means by which works of art evoke personal sensory, emotional, and aesthetic responses.

This module is divided into 11 1-hour units. Units can be readily combined to accommodate longer class periods, taking fewer sessions, or affording more individual free work time for students.

This module also includes several worksheets for structuring idea generation and critique, and a quick-reference guide for Game Maker icons, as well as the 'Game Cards' explaining mechanics introduced in this unit. A full set of Game Cards can be found on the [CurrentLab website](#).

CONTENTS

DAY 1: CRITIQUING GAMES & GAME MAKER BASICS	2
DAY 2: THINGS TO 'AVOID' AND 'RELEASE' IN GAMES	7
DAY 3: EXPANDING THE PLAYSPACE WITH MORE & LARGER LEVELS	13
DAY 4: MAKING 'CONTACT' WITH ITEMS AND HAZARDS	16
DAY 5: INTRODUCING THE FINAL PROJECT, DISCUSSING GAMES' "CENTRAL CONCEPTS"	17
DAY 6: USING VARIABLES TO TRACK LIFE POINTS AND OTHER STATS	18
DAY 7: READING ERROR MESSAGES & WORKDAY	19
DAY 8: USING SOUND EFFECTS & WORKDAY	20
DAY 9: USING MUSIC & WORKDAY	21
DAY 10: BACKGROUND IMAGES & MAKING A TITLE/END SCREEN	22
DAY 11: CRITIQUE	24
APPENDIX A: WORKSHEETS	25
APPENDIX B: GAME CARDS	28
APPENDIX C: GAME MAKER SHORT ACTION REFERENCE	42

DAY 1: CRITIQUING GAMES & GAME MAKER BASICS

- Review **CRITIQUE**
 - What is critique? When you talk about what an artwork does well, and what it can do better.
 - Why do we critique? To help the artist, and ourselves, to make better art.
- In this class we're going to **critique** some video games. We'll be using these questions:
 - **What is fun/interesting in the game?**
 - **What is boring/frustrating in the game? How would we improve those parts?**

CRITIQUE & PLAY: Space Invaders by Tomohiro Nishikado (1978)

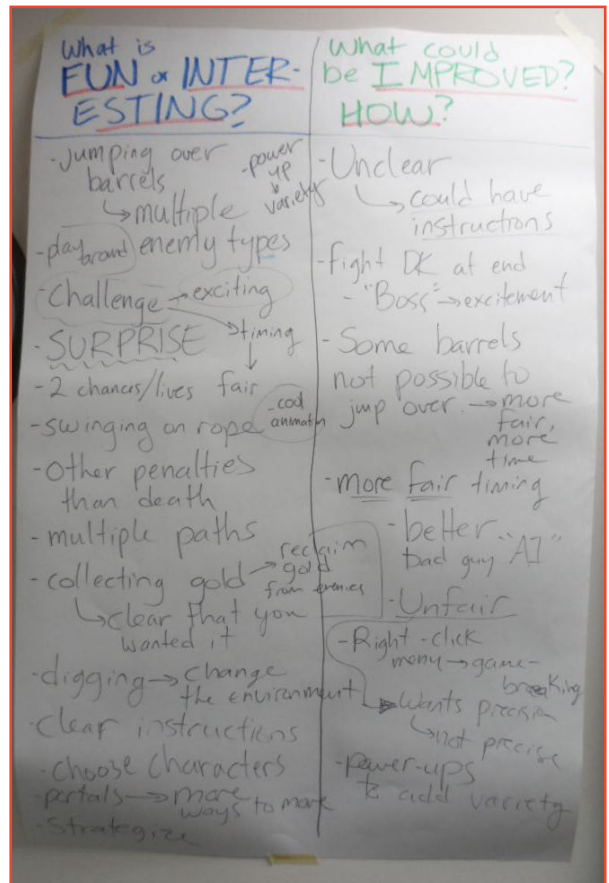
Space Invaders was an extremely influential and successful arcade game in Japan and around the world. It set the stage for later influential Japanese developers, such as Shigeru Miyamoto (Super Mario, Donkey Kong, Legend of Zelda) and Hideo Kojima (Metal Gear), who both cite it as an inspiration. While it is famous as one of the first 'shooting games,' it also includes other inventive elements such as the ability to take cover and a player-malleable environment.



- After playing the game for 5-10 minutes, elicit responses from class based on above critique questions and record to large paper labelled CRITIQUE for future reference.
 - You can divide the paper into two columns, "What is fun/interesting?" and "What is boring/frustrating? How can it be improved?" Retain this paper throughout the unit, adding to it (or extending onto more sheets) as you critique more games. Keep it posted in the room. This will serve as a useful reference document for students when designing and critiquing their own games.
- Also elicit/record to another large sheet, labelled "ACTIONS":

What actions could you DO in this game?

 - e.g. Remove enemies, erode the walls, take cover, move side to side, etc.
 - Again, keep *this* large paper posted, and add to it with each critique. It will serve as an ongoing list of 'verbs' the students can leverage in their own games when they start making them.
- Basic GM tutorial (see following tutorial pages for details):
 - Making a folder on the desktop for 'practice' project
 - Acquaint with layout of program – resources on left, work-space on right
 - Creating sprites (import sample images, explore image editors, give time to customize)
 - Creating objects (attach a sprite, add events and actions for 'player')
 - Creating rooms (play with room editor, place player in room)
 - Play sample 'player-only' level




TUTORIAL - DAY 1

When you open Game Maker, you will notice that the LEFT side of the screen has a list of folders. As you add new resources to your game, Game Maker will automatically sort them into these folders. These will ALWAYS be on your left just as the menu bar will always be on upper part of the screen.



TIP: If you hold the mouse over any of the menu icons or other buttons, a 'tool tip' will appear to tell you what it is or does.

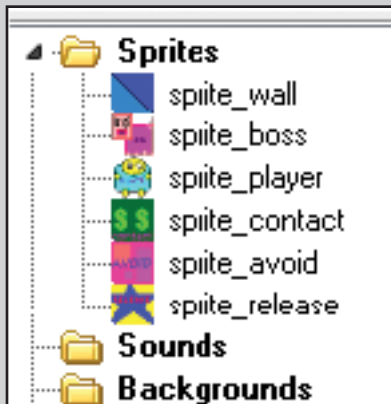
Clear the Game Maker pop-ups and website from your workspace and "hide" the tutorial on your right, if there is one. Try to figure out what each menu icon means. Find the Create Sprite and Create Object buttons. Click **Create Sprite**  and a new properties menu will appear in the gray workspace.

Create Your Sprites

1. Click the **Name** field, where it currently says **sprite0**. Game Maker will always create default names but you should rename it to **sprite_player**
2. Click the **Load Sprite** button. This will open a file window. Find "Example Folder" or the folder that you put the example sprites.



TIP: If you don't have example sprites prepared, students can make simple placeholder "block" sprites by using the image editor to just fill the sprite in with a square of color. Stress that they will later be able to replace these with their own imagery.



3. Select the player image and click **OK**.




Did your sprite show up in the **Sprite Folder**? Good. Repeat the steps until you have these:

sprite_boss	sprite_avoid
sprite_wall	sprite_contact
sprite_release	sprite_player

If you ever need to change a resource, you can reopen its properties by double clicking on its name in the resource folder.



WARNING: Did you notice we used an underscore instead of a space in the names? Game Maker translates our resources into coding language that creates the game. You might experience glitches or bugs if you use spaces or punctuation (!@#%\$%^&*) in the Name fields for sprites, sounds, objects, backgrounds, etc. because they may affect the code. Using `sprite_` and `object_` before the name also keeps resources from having identical names, which would interfere with coding.

4. We'll take a few minutes to customize our **sprite_player**, and become acquainted with the drawing tools. Double-click on **sprite_player**, and then click the **Edit Sprite**  button.
5. Double-click on the image of the player sprite, and you will be in a simple image editor. Invite students to briefly experiment with the **pencil** , **fill** , and other tools to customize their player sprite.


TUTORIAL - DAY 1

Create Your Game Objects

Sprites are just images; they don't do anything in the game on their own. You must create objects to attach the sprites to. Objects control the elements and interactions inside the game.

SPRITE = IMAGE/ANIMATION

OBJECT = BEHAVIOR/ACTION

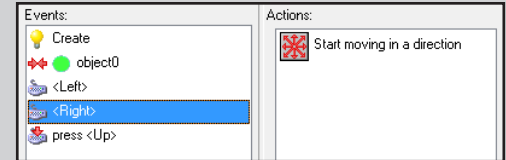
Click **Create Object**  in the menu bar and a new properties menu will appear in the gray workspace. Take a moment to explore the layout. Notice the two spaces labeled Events and Actions, which will determine the object's behavior.

Events are important moments that will trigger responses.

Examples: Objects Collide, Mouse Click, Key Stroke

Actions occur in response to an event.

Examples: Setting the Score, Movement, Release of an Object



Actions will not happen without an Event. Imagine you want to turn the lights on in a room. Flipping the switch is the event. The light coming on is the action. The light will not turn on without flipping the switch.




TIP: You can use the Game Maker Short Action Reference sheet at the end of this packet to help you find the action icons. This sheet is organized by the action tabs on the right of the object properties screen (move, main 1, main2...)



Create the Wall Object







This object will create level boundaries.

1. Click **Create Object**  and change the name to **object_wall**.
2. Click the icon at the end of the sprite field. Select the **sprite_wall** to assign the image to the object.
3. Make sure **'Visible'** and **'Solid'** are both checked. If you later want the barriers to be invisible, you will simply deselect **'Visible'**. If the object isn't solid, you would go through it.
4. Click **OK** to approve the changes.



Create the Player Object

We are going to make this object move when an arrow key is pressed. This section corresponds to the Move: Player (Free Form) game card (Appendix, pg. 28).

1. **Create an object** , name it **object_player**, and choose the **sprite_player**.
2. Click **Add Event** and select **Collision**  with **object_wall**.
3. Find **Move Fixed**  in the **Move tab** and drag the icon to the actions space. In the popup window, click the **center square**  and set the **speed at 0**. Click **OK**.
4. Click **Add Event** and select **Keyboard** . In the drop-down menu, select **Left**.
5. Find **Jump to Position**  in the **Move tab** and drag the icon to the actions space.

TUTORIAL - DAY 1



Create the Player Object (cont.)

6. In the pop-up menu, type **-4** in **x**. Check **Relative**.

7. Make sure **Applies to: Self** is checked before clicking **OK**.



TIP: When a value is not 'Relative,' it is 'Absolute,' meaning it will set the value to exactly that number. 'Relative,' on the other hand, modifies an existing value. For example, if we had not checked relative, pressing left would make our player jump right to '-4' x position on the map - off the left edge of the screen! Because we made it relative, our player will simply modify its current x position, moving 5 pixels to the left.




PLAIN ENGLISH: The sequences we just coded mean:

"When the player collides with the wall, the player will stop moving."

"While the left arrow key is pressed, the player will continuously move 5 pixels to the left."

Let's repeat step 4-7 for Right, Up, and Down using these settings:


Event: Keyboard <Right>

Action: Jump to Position 

x: 4

Check relative


Event: Keyboard <Up>

Action: Jump to Position 

y: -4

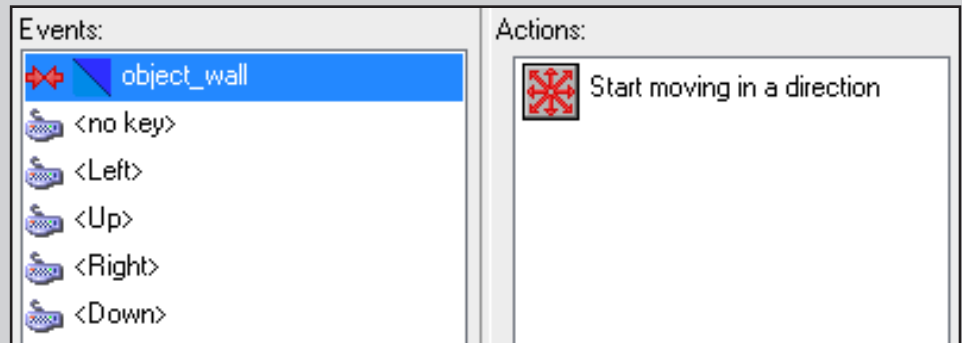
Check relative

Event: Keyboard <Down>

Action: Jump to Position 

y: 4

Check relative




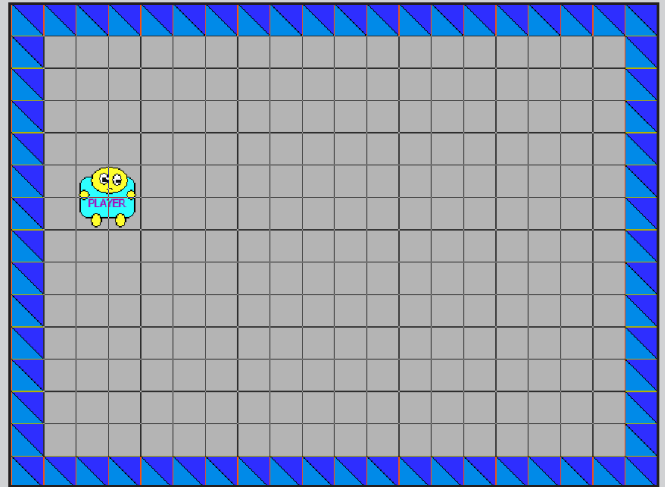
Your object_player's properties will look like this when you're done and ready to click OK and move on.

TUTORIAL - DAY 1

Creating the Room

Now our `object_player` is programmed for movements. Let's create a level to test if your programming sequences are correct. In Game Maker, we use rooms to create levels.

1. First make sure there are no open sprite or object windows. Game Maker will not accept changes until you click **OK**. So if you have changes made in still-open windows, they won't appear in your game!
2. Find and click the **Create Room**  icon in the menu bar. Maximize the window so that you can see the entire room.
3. Click **Settings**. Name this room **room_test** in the name field. You will test objects for functionality in this room and can delete it after the game is complete.
4. Make sure the room is **width=640, height=480** and **snap X=32, snap Y=32**. This will put a grid on the screen that helps object placement. The grid does not appear when you play the game.
5. Click the **Objects tab**. In the drop down menu, select your **object_wall**. Place a surrounding barrier inside the room with left click.



Example room ready to run.




TIP: To create a stream of multiple objects, hold the shift key while placing them.


6. Now select **object_player** and click a place in the left side of the room, but make sure it doesn't overlap an **object_wall**. If objects are partially outside the room, they may be paralyzed in game play.



TIP: Did you misplace an object? Add too many? Put one outside the game room? Then use the mouse to right-click and remove it. As with the above tip, you can remove multiple objects by holding the shift key and right-clicking.

7. When you are ready, click the green checkmark and find the **Run the Game** button  in the main menu bar.

Saving Your Game

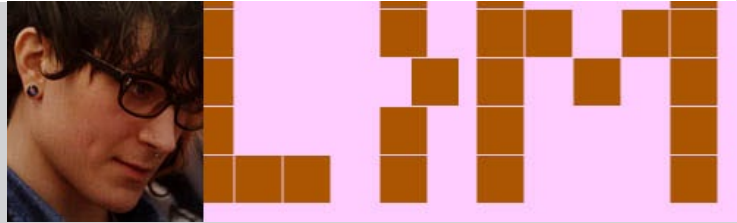
Ideally, students should be saving throughout the process, and not just at the end of the tutorial, to avoid losing work. To save, click the **Save the Game icon** , select your folder, and name your file.

At the of each lesson, you should also save a 'Benchmark' copy of your game. Just go to **File, Save As**, and save a copy of your game with the date (e.g. "AlexApril15"). This way, if anything happens to your game file, you can always open up the most recent benchmark, and never lose more than one day's work.

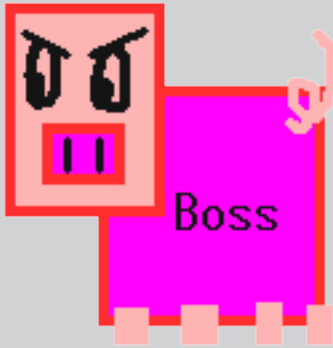
DAY 2: THINGS TO 'AVOID' AND 'RELEASE' IN GAMES

CRITIQUE & PLAY: LIM by Merritt Kopas (2012)

LIM is a metaphoric game relating mechanics to social experience of “passing” or “fitting in” and the aggression that can result when someone doesn’t “fit in.” While the metaphor is autobiographical and stems from the artist’s identity as a queer woman, the abstraction of the game has allowed players to draw connections to bullying, social anxiety, and other personal experiences.





- Remind students of the critique questions before playing:
 - **What is fun/interesting in the game?**
 - **What is boring/frustrating in the game? How would we improve those parts?**
- After playing the game for 5-10 minutes, elicit responses from class based on above critique questions and add to the large “CRITIQUE” paper for future reference.
- Also elicit/record to the large “ACTIONS” sheet: **What actions could you DO in this game?**
 - -e.g. Change color, disguise yourself, move up, down, left, and right, leave the play area, etc.
- Continue Basic GM tutorial using file from last lesson (see following tutorial pages for details):
 - AVOID—What kind of things to you AVOID in games (e.g. enemies, hazards, etc.)?
 - Create a “boss” object and a projectile for it to fire
 - The “boss” is an NPC, or “Non-Player Character” which we have to program behavior for
 - In a lot of games NPCs are bosses or enemies, but many games have friendly or neutral NPCs (like the townspeople in a fantasy adventure game, your team mates in an action game, etc.). Elicit examples from students.
 - RELEASE—What can a player RELEASE in games (e.g. projectiles, magic, fire from a fire hose, etc.)?
 - Create a projectile that the player can emit
 - Like the NPCs, most games use RELEASE objects as aggressive things to attack enemies (bullets, lasers), but a RELEASE can do other things, too (e.g. water to put out a fire, magic to transform one object into another, a ball to throw into a goal or basket, throwing food to feed an animal, etc.).
- If there is extra time after the tutorial, assign an independent task for remainder of class:
 - In pairs, how would you make the boss move side to side, instead of up and down? Can you make it bounce all around the room?





Creating the Boss Object

We are going to make this object move up and down independently in the game.

1. Click **Create Object**  and change the name to **object_boss**.
2. Click the icon at the end of the sprite field. Select the **sprite_boss** to assign the image to the object.
3. Click **Add Event** and the event selector will appear.
4. Click **Create**  **Create** and it will show up in the event space.



PLAIN ENGLISH: 'Create' means "When this object is created..."



5. Find **Move Fixed**  in the **Move tab** on the right. Drag the icon with your mouse to the actions space.
6. In the pop-up menu, select the **up arrow**  and enter a value of **8** for the speed.

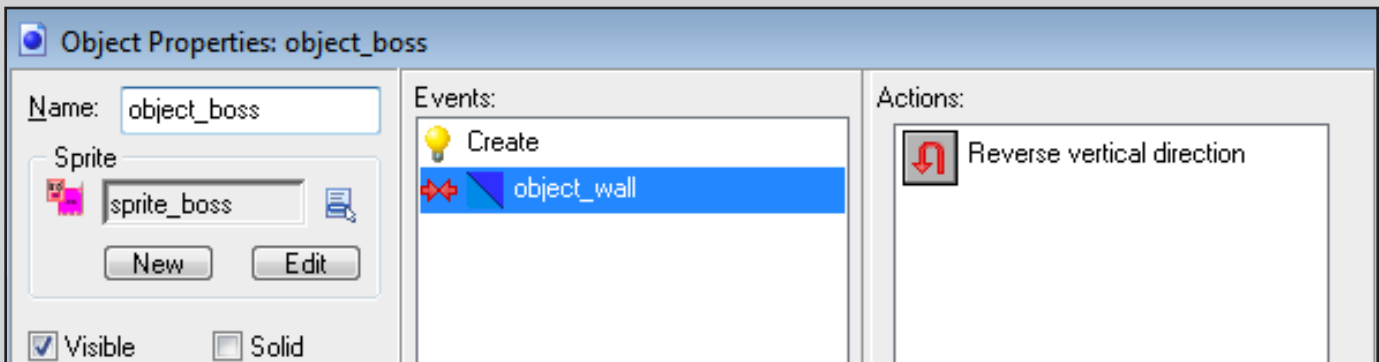


PLAIN ENGLISH: This action means "...the object will move up at the speed of 8."

So, altogether, the Event + Action we've just created means "When this object is created, the object will move up at the speed of 8."

Our next sequence will be: "When the boss collides with a block, its movement reverses vertically."

7. Click **Add Event** and select **Collision**  **Collision**. In the drop-down menu, select **object_wall**.
8. Find **Reverse Vertical**  in the **Move tab** and drag the icon to the actions space. Notice when you switch between events, the action list changes. If you want to alter the actions in the future, make sure you are under the appropriate event.
9. Click **OK** to approve the changes.








How your event sequence for the object_boss should appear. Note how the Actions visible relate the event highlighted on the left. If 'Create' were highlighted, we would see the Actions for that event.

TUTORIAL - DAY 2





AVOID: Having the Boss Release Objects






This object will be created at random by the object_boss and is for the object_player to avoid. If you want a visual reference for your students, this section corresponds to the “Avoid” game card.

1. Click **Create Object**  and change the name to **object_avoid**.
2. Click the icon at the end of the sprite field and select **sprite_avoid** to assign the image
3. Click **Add Event** and select **Create** . In the **Actions**, select **Move Fixed** .
4. Click **Applies to: Self**. Select the **three arrows pointing left**. This will make the object_avoid go randomly in one of three directions. Set the **speed to 12**. Do not check relative. Click **OK**.
5. **Add Event** and select **Collision**  with **object_wall**.
6. Find **Destroy the Instance**  in **Main 1 tab** and drag it to the Actions. In the pop-up menu, select **Applies to: Self** and click **OK**.



TIP: What do you think would happen if you selected ‘Applies to: Other’? Ask the students if they can guess. (Or if you have time, you can try it out and run the game to see what happens!) The object_avoid destroys the wall blocks—is there any way this ‘mistake’ could be used on purpose in a game?

7. **Add Event** and select **Other** . In the pop-up menu, select **Outside Room**.
8. Drag **Destroy the Instance**  to the Actions. In the pop-up menu, select **Applies to: Self** and click **OK**.

Events:	Actions:
 Create	 Destroy the instance
  object_wall	
 Outside Room	



PLAIN ENGLISH: These Sequences Mean:

“When the Avoid object is created, it will move to the left diagonally or straight across (chosen at random).”

“When the Avoid object collides with the wall, the Avoid object will destroy itself.”

“When the Avoid object goes outside of the room, the Avoid object will destroy itself.”

Because Game Maker keeps track of objects even when they’re off-screen, we destroy the object_avoid when it leaves the screen. This way, the game won’t be calculating the position for tons of unseen, irrelevant projectiles, which would slow the computer down!

Now, let’s make the boss release the object_avoids. Click **OK** to close the object_avoid properties and open the **object_boss**.



AVOID: Having the Boss Release Objects (cont.)

9. Click **Add Event** and select **Step**  . In the pop-up menu, select **Step** again.

10. Find **Test Chance**  in the **Control tab** and drag it to the Action space.

11. Enter **50** in the sides and click **OK**.



PLAIN ENGLISH: What 'Test Chance' does is roll a virtual die with the number of sides we say. Imagine our die has 49 sides saying 'NO' and one side saying 'YES.' 'Test Chance' only does the next action (here, releasing the object_avoid) if 'YES' comes up. If we had entered '2' sides, the chance would be 1/2 — a heads-or-tails flip of a coin — that at any given frame of the game the boss will be releasing objects.

Sometimes students have trouble understanding that larger numbers mean a smaller chance (or fewer fireballs). Giving them time to play with the 'sides' value could help solidify their understanding.







12. Find **Create Instance**  in the **Main 1 tab** and drag it to the Action space.

13. Click **Applies to: Self**. Click the icon at the end of the object field to select **object_avoid**. Leave **x: 0** and **y: 0**, but check **Relative**.



TIP: Remember, 'Relative' is the opposite of 'Absolute.' Since we clicked 'Relative,' it will place the object_avoid at point (0, 0) relative to the object_boss—the upper-left corner of its sprite. If we did not click 'Relative,' it would place it at (0, 0) on the game itself—the upper-left corner of the screen!

14. Click **OK**.

Events:	Actions:
 Create	 With chance 1 out of 50 perform next
 Step	 Create instance of object object_avoid
  object_wall	



PLAIN ENGLISH: This sequence means "At every step/frame in the game, there is a 1 out of 50 chance that the boss will create an Avoid object at the boss's current position in the game."

A step in Game Maker is a unit of time, like a single frame in a film or video. Since our default game speed is set to 30 frames per second, a step is .033 seconds long. A Step Event is an event that happens every step of the game. This is why we have our 'Test Chance' only creating on a 1/50 chance. The game is rolling that die 30 times every second, which means there are a lot of opportunities for it to create object_avoids. If the chance were higher, say, a 1/2 chance, there would be a high likelihood of the boss flooding the screen with projectiles!








15. Press the Play button to see if it worked. If something isn't working how you expect, look back at your Events and Actions—Did you miss something? Is something not 'Relative' when it should be?

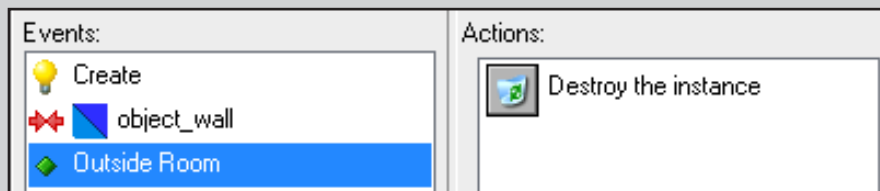
TUTORIAL - DAY 2



RELEASE: Having the Player Release Objects

Now we want to add a release for the player to destroy the 'Avoids.' This object will be created when the player hits the spacebar. This section corresponds with the 'Release' game card.

1. Create an **object**  and change the name to **object_release**.
2. Select the **sprite_release** to assign the image to the object.
3. Click **Add Event** and select **Create**  **Create**. Find **Move Fixed**  in the **Move tab** and drag the icon to the actions space.
4. In the pop-up menu, select the **arrow pointing right**  and set the **speed to 30**.
5. Click **Add Event** and select **Collision**  **Collision** with **object_wall**. Find **Destroy the Instance**  in the **Main 1 tab** and drag it to the actions space. Click **OK**.
6. Repeat this Action for an **Event: Other**  **Other** **<Outside Room>**, then click **OK** to close.





PLAIN ENGLISH: These sequences mean:

"When the Release is created, it will move to the right at a speed of 30."

"When the Release collides with the wall, the Release will destroy itself."

"When the Release moves outside the room, the Release will destroy itself."


Next we will program the player to create the **object_release**. Open the **object_player** by double-clicking it in the object resources folder.

7. Click **Add Event** and select **Key Press**  **Key Press** and **<space>**. Find **Create Instance**  in the **Main 1 tab** and drag it to the actions space.



WARNING: Did you notice we used KEY PRESS instead of KEYBOARD in Events? KEYBOARD **<space>** would create a constant flow of **object_releases**, which could be useful later. In this game, it would make it too easy.

8. Make sure to click **Applies To: Self**. Select **object_release** and check **Relative**

9. Click **OK** to approve the changes. Press **Play**  and test your game.



PLAIN ENGLISH: This sequence means: *"When the space bar is pressed, the player will create a Release object at the player's current position."*

AVOID: Giving the 'Avoid' Objects Consequences

From here on out, we'll just show you the Events and Actions to accomplish different tasks. This formatting is similar to the 'game cards' included in these modules. Here are the events and actions to make the game end when the player collides with an object_avoid:

Object_Player

Event: collision<object_avoid>



Actions: Show High Score

Restart game

Events:	Actions:
object_wall	Show the highscore table
object_avoid	Restart the game
<no key>	
<Left>	
<Up>	
<Right>	
<Down>	
press <Space>	



PLAIN ENGLISH: This sequence means "When the player collides with the avoid, the game will show the high score and restart."

Now, we'll add some code to the 'Avoid' object so that when the player's 'Release' collides with it, the 'Avoid' object is destroyed and the score goes up:

Object_Avoid

Event: collision<object_release>



Actions: Destroy Instance

-Applies to self

Set Score

-New Score: 25

-Check Relative

(remember 'Relative' adds 25 to the score—it doesn't set it to 25!)

Events:	Actions:
Create	Destroy the instance
object_wall	Set the score relative to 25
object_release	
Outside Room	



PLAIN ENGLISH: This sequence means "When the player's 'Release' projectile collides with the boss's 'Avoid' projectile, the 'Avoid' is destroyed, and the score goes up by 25."

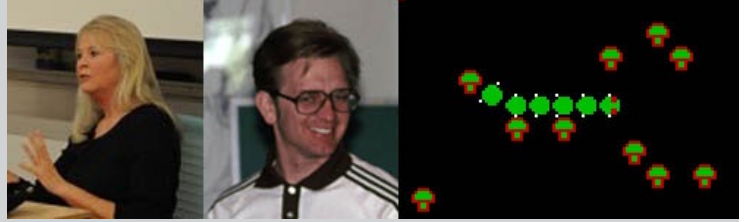


TIP: If there is free time after the tutorial, encourage students to play with the different values like speed, direction, and position. You could assign them a specific independent task, such as: "In pairs, how would you make the boss move side to side, instead of up and down? Can you make it bounce all around the room?"

DAY 3: EXPANDING THE PLAYSACE WITH MORE & LARGER LEVELS

CRITIQUE & PLAY: Centipede by Dona Bailey and Edd Logg (1981)

Centipede was a notable and very successful American arcade game that built upon Space Invaders by adding more NPC types and having actions that affect the environment.




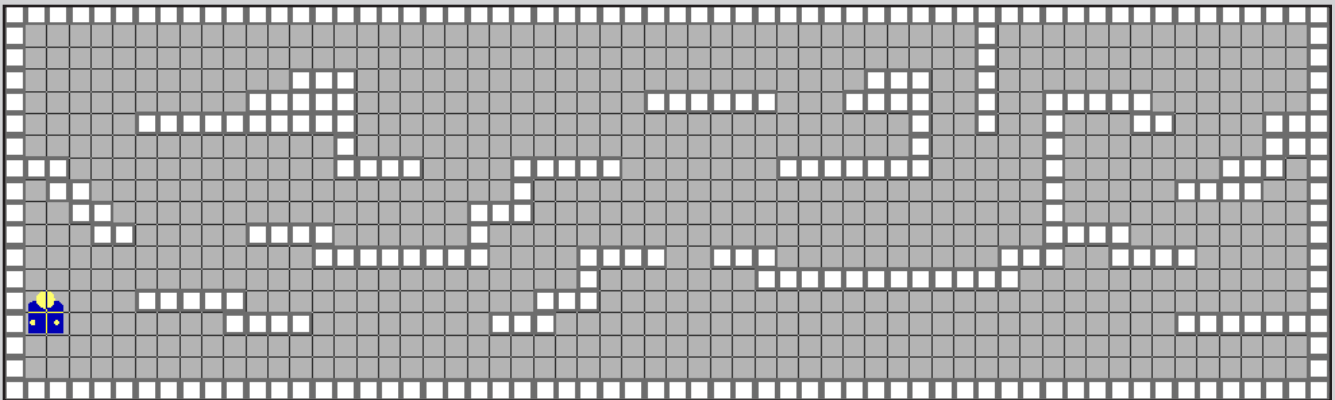
Shooting enemies creates mushrooms which block shots, however the mushrooms that make up the landscape can be destroyed.

- Remind students of the critique questions before playing:
 - **What is fun/interesting in the game?**
 - **What is boring/frustrating in the game? How would we improve those parts?**
- After playing the game for 5-10 minutes, elicit responses from class based on above critique questions and add to the large “CRITIQUE” paper for future reference.
- Also elicit/record to the large “ACTIONS” sheet: **What actions could you DO in this game?**
 - -e.g. Move up, down, left, and right. Shoot. Transform centipedes into mushrooms. Destroy mushrooms.
- GM Tutorial (see following pages):
 - Creating larger rooms where the camera follows the character
 - Can elicit the difference between the self-contained screens of Centipede and Space Invaders with the larger scrolling maze of LIM.
 - Students can make a few different rooms/levels of varying sizes.
 - Exits—creating an object that moves the player from one room/level to another
 - If there is extra time, encourage students to flesh out their large levels by creating additional ‘wall’ objects, or placing in more NPC/boss objects.

Creating Larger Levels

You are going to make some larger levels to get your ideas flowing—think about how different the space felt in LIM versus Space Invaders! Today’s task is to create 3 large levels in different shapes. Your challenge is to use your `object_walls` to make an interesting space for the player to travel across.

1. Click **Create a Room**  in the menu bar. In the **Settings tab**, label the room **long_rectangle** in the name field. Then type **1920** in the **width** field and **690** in the **height** field. Use the **object_wall** to create a border around the edges of the room. Remember that you can hold the <Shift> key and left-click to make a stream of objects, or <Shift> key and rightclick to remove the stream.
2. Decide if you are going to have the **object_player** start on the left and move right or start on the right and move left. Place the **object_player** in the room. Now create walls and passages that would require someone playing the game to find their way across.



3. Now, we will set the camera so the player can’t see the whole level at once. This will add challenge and force someone playing to explore your level. Click on the **Views tab**.
4. Check **Enable use of views** and **Visible** when room starts. Scroll down to where it says **Object following**. Select **object_player**. Change **hbor** to **290** and **vbor** to **150**.

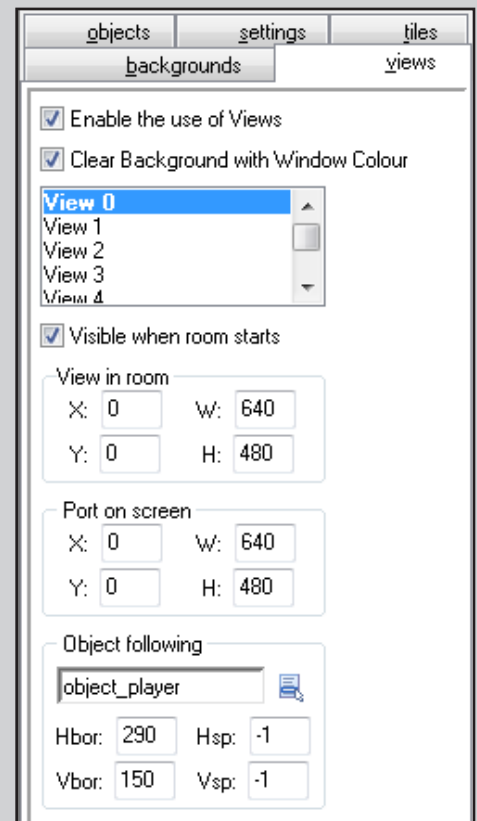


PLAIN ENGLISH: ‘hbor’ and ‘vbor’ are short for ‘horizontal border’ and ‘vertical border.’ They dictate how close the player must get to the edge of the screen before the camera moves to follow it. If the ‘hbor’ were 0, the player would have to move all the way to the edge of the screen before the camera moved, making it hard to see what was ahead!


5. Click the green check to accept the changes and drag the **long_rectangle** room up to the top in the “rooms resource folder” on the left of the screen. This will let you test this room first.

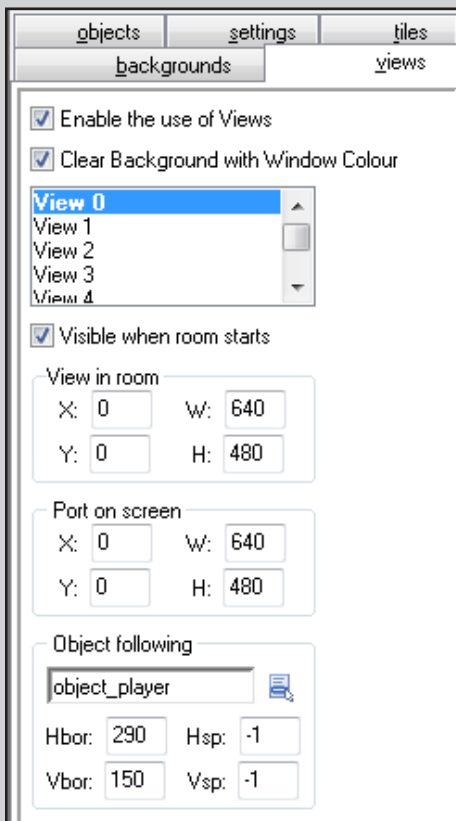
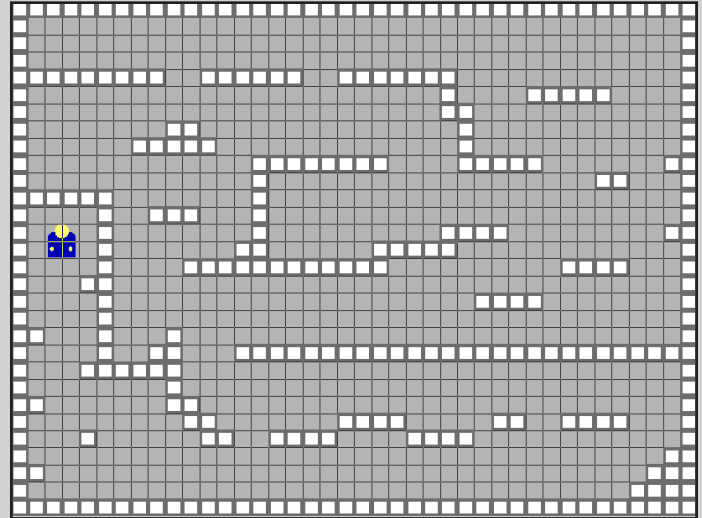



TIP: Levels play in the order they are listed on the left. You can drag them to reorder them or to dictate a ‘start screen.’





Creating Larger Levels (Cont.)

6. Create a new room  in the menu bar. In the Settings tab, label the room “**large_rectangle**” in the name field. Then type **1280** in the **width** field and **960** in the **height** field. Use the **object_wall** to create a border around the room.
7. Place the player in the room, and create walls and passages in the level.
8. The process for setting up the camera is the same as above. Click on the **Views tab**. Check **Enable use of views** and View when room starts. Scroll down to where it says **Object following**. Select **object_player**. Change **hbor** to **290** and **vbor** to **150**.
9. Click the green check to accept the changes and drag the **large_rectangle** room up to the top in the “rooms resource folder” so you can test the room.



10. See if your player gets stuck anywhere or cannot get across the room. Fix any issues and have another person test your level while you test theirs.
11. Finally, create a third room , labeled “**tall_rectangle.**” Then type **640** in the width field and **1440** in the height field.
12. Invite students to independently set up the camera as they did for the previous two rooms, and playtest it. Encourage peer assistance, or let students independently check their work against the “Scrolling Camera” game card.

Exiting Levels

1. Create a new sprite . This will be a ‘portal’ or ‘exit’ from one level to another.
2. Create an object  and assign it the new portal/exit sprite.
3. Use the “**Change Rooms**” game card (included at the end of this module) to walk the students through the Actions and Events.

If you haven’t been using the game cards (see Appendix, page 28) until this point, this will serve as an opportunity to introduce the students to using them independently as a resource.

If there is extra time, encourage students to flesh out their levels with additional wall objects using different sprites, to try adding boss/NPC objects to their levels, or to create more levels.

DAY 4: MAKING 'CONTACT' WITH ITEMS AND HAZARDS

CRITIQUE & PLAY: Katamari Damacy by Keita Takahashi (2004)

Katamari Damacy is a game by artist/designer Keita Takahashi with unconventional non-adversarial goals. Rather than fighting an enemy, the player plays as the 'Prince of Space,' trying to roll as many objects onto his ever-increasing 'Katamari,' or sticky ball, as he can. The player must start by rolling up smaller objects, until her/his ball is large enough to absorb bigger and bigger items.



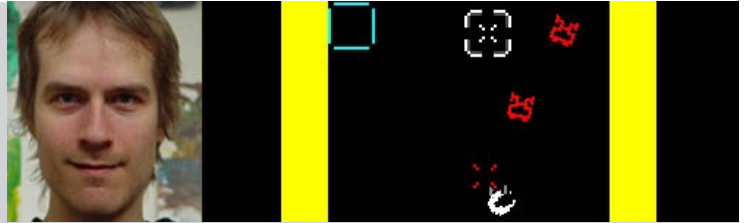
The version linked to on CurrentLab is a 2-D flash version of the game created to promote the 3-D Playstation 2 version.

- Remind students of the critique questions before playing:
 - **What is fun/interesting in the game?**
 - **What is boring/frustrating in the game? How would we improve those parts?**
- After playing the game for 5-10 minutes, elicit responses from class based on above critique questions and add to the large "CRITIQUE" paper for future reference.
- Also elicit/record to the large "ACTIONS" sheet: **What actions could you DO in this game?**
 - -e.g. Pick up smaller and larger objects (Including the player character itself, and parts of the level!). Grow from small to big. Control more than one player. Follow the mouse.
- GM Tutorial (see relevant game cards at end of module):
 - From here forward, we'll be using the game cards more than tutorials to encourage student autonomy.
 - Use game cards to introduce:
 - Collectables (see card "Contact: Collectable," pg. 34)
 - Hazards (see card "Avoid: Hazards," pg. 35)
 - If there is extra time at end of class, encourage students to explore other game cards in this module, and attempt to implement one in their game.

DAY 5: Introducing the Final Project, Discussing Games' "Central Concepts"

CRITIQUE & PLAY: Idealism by Jason Rohrer (2008)

A metaphoric "sketch" game by artist and game designer Jason Rohrer relating mechanics to loss of ideals and taking shortcuts in life. As the game levels become more complex, players can take shortcuts to save time and have a higher score, but taking shortcuts can turn allies into enemies and make the level harder. The game was actually made in Game Maker.



- Remind students of the critique questions before playing:
 - **What is fun/interesting in the game?**
 - **What is boring/frustrating in the game? How would we improve those parts?**
- After playing the game for 5-10 minutes, elicit responses from class based on above critique questions and add to the large "CRITIQUE" paper for future reference.
- Also elicit/record to the large "ACTIONS" sheet: **What actions could you DO in this game?**
 - -e.g. Move with mouse, shoot with mouse, choose a path, take (or don't take) shortcuts, lose allies, etc.
 - Note how the levels gradually introduced each element, to teach you how to play the game.
- **Introduce game project:**
 - We've been making a 'practice' project together, now you'll be making your own game
 - Your game needs a central concept. Board "**Concept - what is the game telling us?**"
 - A game can **tell a story**: Space Invaders: "The aliens are attacking, and must be defeated!"
 - A game can tell us about an **idea**: LIM: "Sometimes you feel like you have to blend in, but not being yourself can be harmful, too."
 - Look at the "ACTIONS" list you've been building over the last few classes – what actions are the most interesting to you? Is there something you want to DO in a game that isn't already on the list? What are good VERBS for your game? **How did the VERBS in the games we played relate to their central concept?**
 - Space Invaders: Shooting the aliens and taking shelter relates directly to the alien invasion story.
 - LIM: Being able to change colors relates to the idea of fitting in. Not being able to attack relates to the helplessness a lot of victims of bullying feel. The shaking and slowness when you're changed relates to how it feels to not be yourself around other people.
 - **Your game should include:**
 - A central **concept** (either an **idea**, like LIM or Idealism, or a simple **story**, like Space Invaders)
 - Actions/Verbs that relate to that concept
 - At least 3 rooms
 - Background images in every room
 - At least two different kinds of environment (not all rooms look the same!)
 - At least 3 different NPCs (non-player characters – allies, enemies, or neutral)
 - Something that your character can touch or pick up (collectible, health, power-up)
 - NONE of the placeholder sprites we used in our practice projects!
- **Sketch book:** students use pencils and markers to sketch out ideas for game levels, verbs, and characters
 - Students can use worksheets, included in this module, for character and level creation.
 - The character creation worksheet is fairly self-explanatory, but if you want, you could fill one out as a class for a famous game character such as Samus or Mario.
 - Once students' written/drawn plans are okayed by the teacher, they can begin creating the necessary sprites and objects in their game.

DAY 6: USING VARIABLES TO TRACK LIFE POINTS AND OTHER STATS

CRITIQUE & PLAY: River Raid – Carol Shaw (1982)

Building upon the ‘shooter’ template set out by Space Invaders and Centipede, Shaw innovated by adding a procedurally-generated scrolling world the player flies through. The level

was actually too large to be stored on an Atari cartridge, so Shaw had to code it to make the Atari system build the level as you play it. Another innovation is the “heads-up display” at the bottom of the screen showing a fuel meter which must be regularly replenished. (This is an unusual use of the ‘life bar’ mechanic students will learn in today’s tutorial.)



- Remind students of the critique questions before playing:
 - **What is fun/interesting in the game?**
 - **What is boring/frustrating in the game? How would we improve those parts?**
- After playing the game for 5-10 minutes, elicit responses from class based on above critique questions and add to the large “CRITIQUE” paper for future reference.
- Also elicit/record to the large “ACTIONS” sheet: **What actions could you DO in this game?**
 - -e.g. Move left and right, shoot, choose a path, replenish fuel, etc.
- GM tutorial (see relevant game cards at end of module)
- Creating a lifebar object that follows the camera (see cards: “Player Health Bar. pt. 1,” pg. 36, and “Follow Coordinate Placements,” pg. 38)
- Having collisions with ‘Avoid’ objects affect the lifebar (see card: “Player Health Bar, pt. 2,” pg 37)
 - Elicit from students:
 - **How could we make an object replenish the life bar, like the fuel tanks in River Raid, or like power-ups in other games?**
 - **How could you make the life bar go down constantly, like the fuel bar in River Raid? [Have a ‘Step Event’ decrease the bar, instead of a ‘Collision Event’] What else could you use such a lifebar for? [An ‘air meter’ for an underwater game, a ‘countdown clock,’ a ‘hunger meter’ in a survival game where you need to eat to keep from dying, etc.]**
- Giving the boss multiple hit points (see card: “Boss/Enemy Health”)
 - This card uses “variables.” A variable can keep track of anything you want it to, not just life. You could have a variable instead of a number for the strength of the character’s RELEASE objects, so the player could do more damage as s/he advanced through the game. You can use variables to keep track of the direction the character is moving so you can make them shoot in that direction.
- Remainder of class is free work time on students’ individual projects.

DAY 7: READING ERROR MESSAGES & WORKDAY

- Remind students of **critique**:
 - **Why do we critique works of art?** [To show how they can be stronger. To point out what is working and what's not working.]
 - **Which would make you a stronger artist: Listening to critiques and suggestions about your art, or getting frustrated and ignoring them?**
- Error messages are Game Maker's way of critiquing your game by telling you what isn't working.
 - If you listen to what the error says, you can fix your game and make it better
 - If you get frustrated and close it, you'll never fix your game!
- Open sample 'broken.gmk' game and run it. You'll get an error like this:

```
ERROR in
action number 1
of Draw Event
for object object_lifebar:
Error in expression:view xview +10
position 1: Unknown variable view
```

the object where the error is occurring

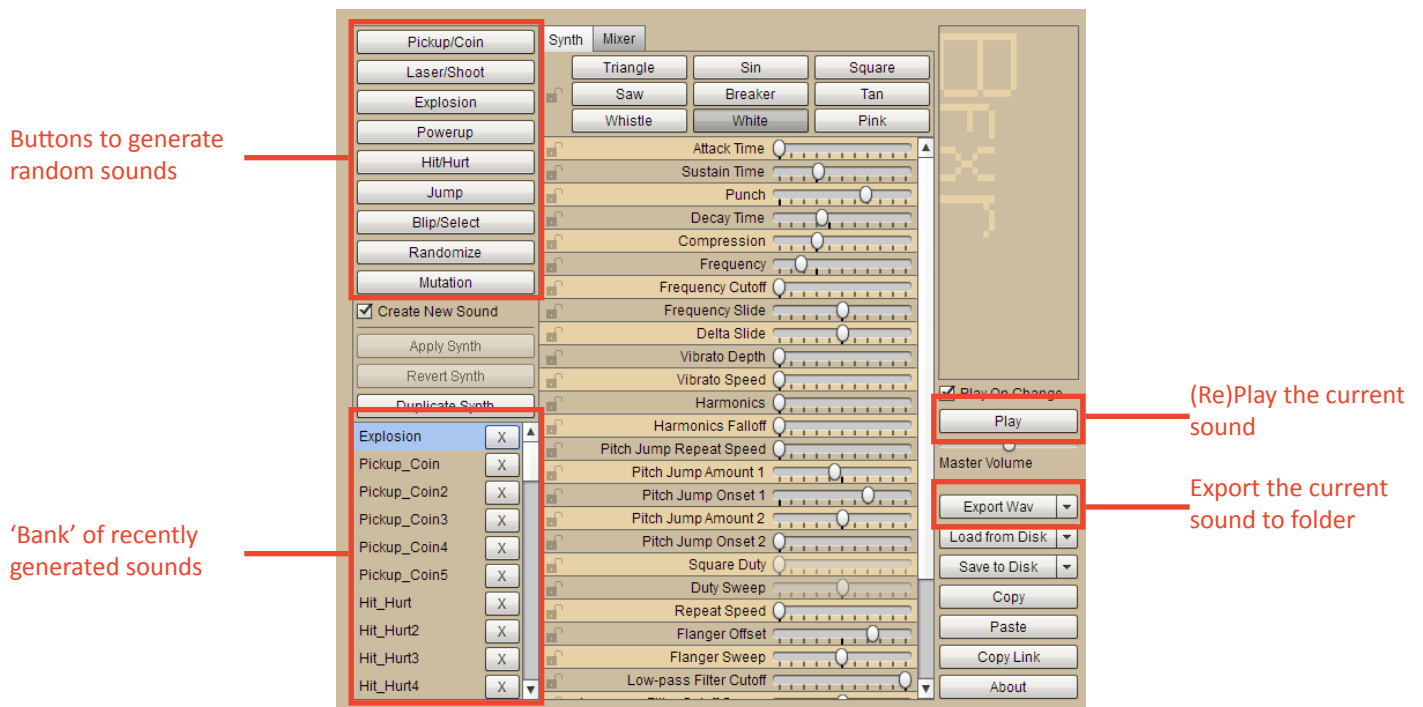
the code that is wrong

a description of the error

- Elicit from the students: **Where is this error happening in our game?** [object_lifebar] **What seems to be the problem in the object?** [Something with the "view_xview" code]
 - Open the game, and look in **object_lifebar** for the problem. Try to elicit the issue from the students.
 - Someone typed "view xview" with a space, instead of "view_xview" with an underscore!
 - Fix the error and run. The game works, but still has problems!
 - When we press "space" the RELEASE appears in the corner. Try to elicit solutions from students at each stage:
 - Check in the **object_player**, under the **keypress <SPACE>** event
 - Someone forgot to check "relative" in the "create" event!
 - The boss's projectiles destroy the level instead of being blocked by the walls. Try to elicit solutions from students at each stage:
 - Check in the **object_avoid**, under the **collision <object_wall>** event
 - Under the "destroy" event, someone checked "other" instead of "self"!
 - Remainder of class is free work time on students' individual projects.

DAY 8: USING SOUND EFFECTS & WORKDAY

- Discussion of sound in games:
- **What are the traditional five senses? [Sight, Hearing, Feel, Taste, Smell]**
- **Which of these senses do the games we play use most? Can you see the game world? Hear it? Can you feel it or taste it or smell it?**
- Since we as artists only have control over two senses (sight, sound) in the games we're making now, we have to find ways to use *those* senses to convey other sensations.
 - **In real life, when you pick something up, how do you know it's in your hand? [You feel it, perhaps see it, in your hand.]**
 - **In Super Mario, how do we know Mario has picked up a coin? [We hear a sound! Can play "Sound_Pickup_Coin.wav" included in this module]**
 - **In real life, how do you know that someone has hit you? [It hurts!]**
 - **In a game, how do you know? [Usually, you hear a sound. You might see your life bar go down, too. Can play "Sound_Hit_Hurt.wav" to illustrate]**
- Play other example sounds from module ("Sound_Jump.wav," "Sound_Grow.wav," "Sound_Laser.wav," "Sound_Rumble.wav"), and elicit was feeling/sensation the sound evokes for them
 - There are NO RIGHT ANSWERS, though the filenames indicate possible responses. Have students try to explain the connection they draw between the sound (e.g. "The sound goes up, so it feels like jumping.")
- Tutorial: Making sound with BFXR and adding sound effects to a game
 - Go to www.bfxr.net, or use a version downloadable from the CurrentLab website.
 - Show how students can generate different common types of electronic sounds using the buttons on the left, without having to play with any of the complex controls in the middle. Once they find a sound they like, they can "Export Wav" and save their sound effects to their project folder.
 - Ask the students to create a single "release" sound, stressing that they can add more to their game later
 - Adding sound effects to the game itself is covered under the "Adding Sounds" card.

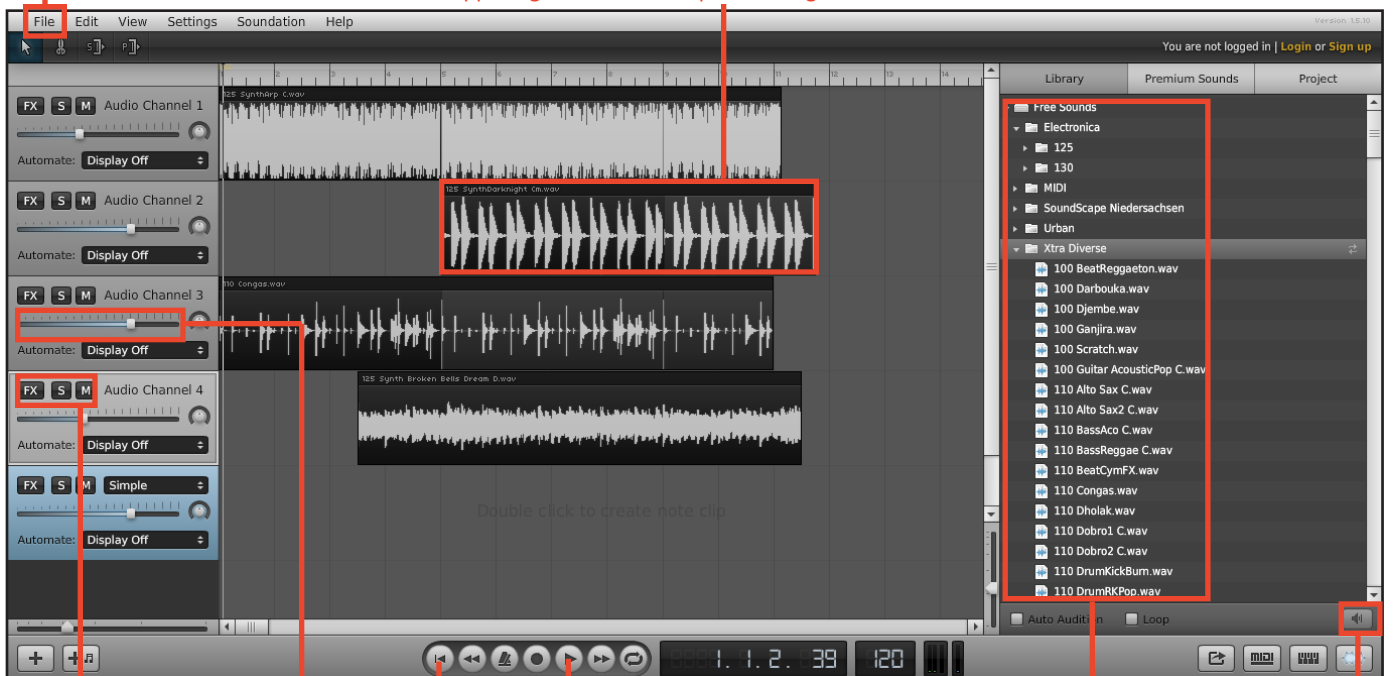


DAY 9: USING MUSIC & WORKDAY

- Play three different examples of game music which create different moods.
 - MegaMan 2 Boss Intro & BubbleMan theme
 - <http://www.youtube.com/watch?v=YruCYWffiiM>
 - **What does the music add to the game?**
 - **How does the introduction music make you feel before the level starts?**
 - **How does the music in the level itself fit with the MegaMan gameplay? (e.g. fast tempo)**
 - Music from Passage, by Jason Rohrer
 - <http://www.youtube.com/watch?v=n3o0HFXPfc0>
 - **What does the music add to the game?**
 - **How is the feeling of this music different from the MegaMan music?**
 - **What aspects of the music make the feeling different? (e.g. slower tempo, no pounding drums)**
- Tutorial: **Soundation Studio** (<http://soundation.com/studio>), a free online music sequencing program
 - Go to Soundation, and look at the screen.
 - There are four 'tracks' on the left. These are like different layers of the song - kind of like how your game has a background layer, a level, and then characters on top.
 - The work area is in the middle, like Game Maker.
 - On the right are folders from which you can drag different sounds into your song.
 - Demonstrate how to drag sounds into different layers, and press 'play' to test them. You can add more layers with the '+' button. Dragging the upper-right corner of a sound in the work area will let you 'loop' it for a longer time.
 - Some songs have different numbers (125, 130, 100, etc.) these indicate the tempo of the sound clip. If you mix clips with different tempos, you will get a window asking if you want to 'pitch shift' or 'time stretch' the sound. Choose 'time stretch.' 'Pitch shift' changes how high or low the sound is, while 'time stretch' just changes how fast or slow it is to fit the rest of the song.
 - Going under file -> export .wav will let you save the finished song to the computer so you can add it to your game.
 - Adding music to the game itself is covered under the "Adding Sounds" card.

File -> Export .wav saves song to computer

Sound in a layer in the work area. Click + drag the upper-right corner to 'loop' it for longer times



Add special effects, solo, or mute a track

Adjust volume for a track

Return player to beginning

Play song



Folders of available sounds

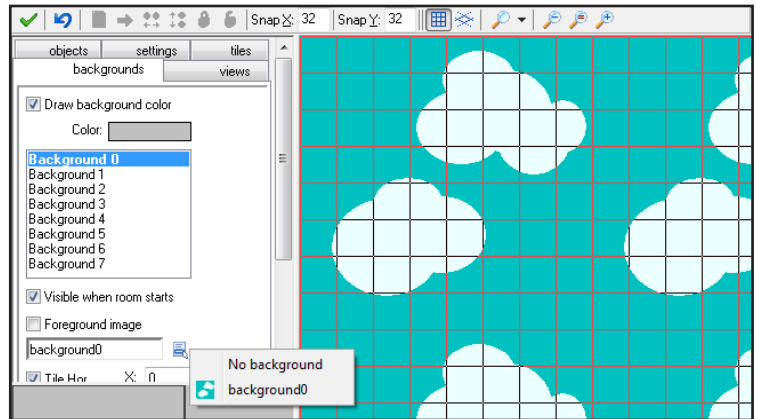
Preview a selected sound

DAY 10: BACKGROUND IMAGES & MAKING A TITLE/END SCREEN

- In this lesson, we'll demonstrate how to add background images to your levels to make them more visually interesting, and how to use the same tools to make a title and ending screen.
- If students need more work time, the title/ending screen part of the lesson can be postponed or elided.


Creating a Background Image

- Click on the 'Create Background' button 
- Click on the 'Edit Background' button 
- Click on **Transform - > Resize Canvas**
 - We'll make the background image bigger, say 256 pixels x 256 pixels
 - Use the drawing tools to make a simple image that will repeat in the background of the level, say, a brick wall or cloudy sky.
 - Click the check mark to save it.
- Open one of your rooms, and click on the **backgrounds tab**.
- Click on the icon next to <no background> and select the background you made. It should appear in the level window to the right.





Creating a Title Page

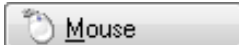
- As an optional precursor to this tutorial you could have students look at example title screens for five minutes at <http://www.titlescream.com/> and select a favorite, then share with the group which they chose and why, or pull a few examples such as [The Legend of Zelda](#), [Sonic the Hedgehog](#), or [Final Fantasy IV](#) to discuss and critique as a group. However, at this point in the project, time is typically running short, and you may prefer to give students more work time.


- As, before, 'Create' and 'Edit' a new background
- Click on **Transform - > Resize Canvas**
 - Uncheck **Keep Aspect Ratio**
 - Set the size to be 640 pixels x 480 pixels
- Use drawing tools to create a title screen
 - The **Text tool**  can be used to place text on the title page
 - The **Font** button, which appears when the text tool is selected, will allow you to choose the size and style of the text.

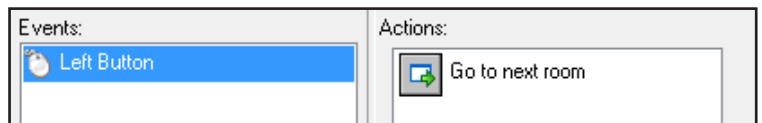


- Create a new sprite, 'sprite_start'**  - this will be our 'start' button
 - Edit the sprite** and create a 128 x 64 button, using the text tool to write 'START' or 'Click to Start' on it


- Create a new object, 'object_start'** 
 - Assign the object the sprite **sprite_start**
 - Add a **Mouse - Left Button Event**

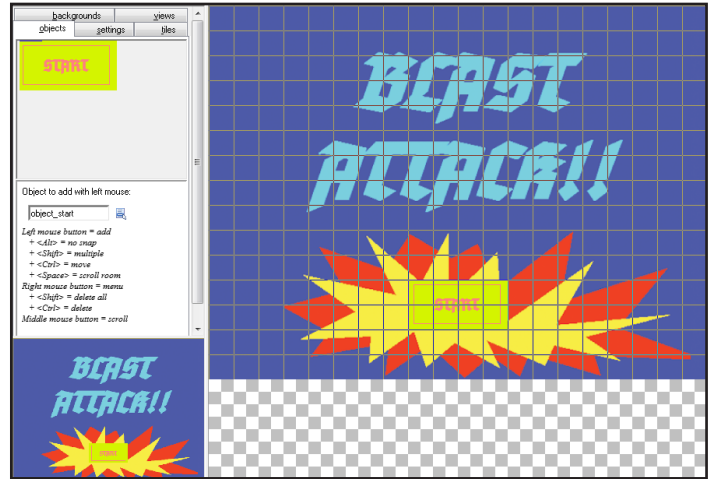


- For this event, add a **Go to Next Room** action 



DAY 10: BACKGROUND IMAGES & MAKING A TITLE/END SCREEN (cont.)

- Create a **new room**  and call it **room_start** under the **settings tab**.
- As above, go under the **backgrounds tab** and set the background to your title screen image.
- Then go under the **objects tab** and place your start button on the screen.
- Click the green check for **OK**.
- On the left, be sure to drag **room_start** to the top of the rooms folder, so that the title screen will be the first room the player sees.
- **Run the game**. You should see your title screen, and when you click on the button the game proper should begin.
- Students should have the rest of classtime to work on their individual projects.
 - Encourage them to create an ending screen with a 'reset' button using the strategies they used to create the starting screen.
- The next lesson in this module is the final critique. You may want to give students a period or two of pure work time to finalize their projects.
 - One good strategy to help ensure the quality of the final project is to tell the students that a game isn't complete until they can play through it from start to finish, and a friend can also do so without having to be told what to do by the creator. This allows students who 'finish' early to remain engaged by playing peers' games and getting feedback on their own games from peers.



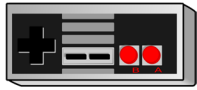

DAY 11: CRITIQUE

- This is a plan for the final critique of the project. If you have time in your schedule, it would also be helpful to do one of these earlier as a “process critique” while the students’ games are still in progress,
- Review with students the **purpose of critique**
 - To **help the artist** make their work stronger by talking about:
 - What their art is already doing well.
 - Parts that they can improve - **and how to improve it.**
 - Is it helpful to the artist if you just complain about their game? **NO.**
 - Would you like it if someone just complained about your game? **NO.**
- **Critique is not a place to bash other people’s work - the point is to be helpful.**
- Remember our two critique questions (rephrased here to encourage more positive criticism):
 - **What is fun/interesting in the game?**
 - **How could the artist improve parts of the game that aren’t fun/interesting?**
- **Critique:**
 - Give each student a critique sheet (included in this module).
 - The sheet has two columns reflecting the two critique questions, and three rows representing three games they will play.
 - Ask the students to load their games up on the computers and hit ‘run.’
 - Ask the students to rotate seats, and play the game on the computer for five minutes.
 - Then give the students five minutes to record their critique of their peer’s game.
 - Written critiques are a good, concrete artifact to assess/grade students’ understandings of the underlying concepts. Telling the students they will be graded on their written critiques also helps ensure that they will invest some effort in them.
 - Repeat this process for two more rotations.
 - Afterward, discuss as a class: What were things in general that worked really well in games we played in class? What things could have been improved? Are there any problems/issues that seemed to come up in a lot of games?

CHARACTER CREATION SHEET

This character is a:

PLAYER CHARACTER **NPC**

(circle one)

This is a sketch of the character:



These are the things the character **can DO**:



This is its **GOAL**:



These are the things the character **wants to AVOID**:



This is how it **MOVES**:

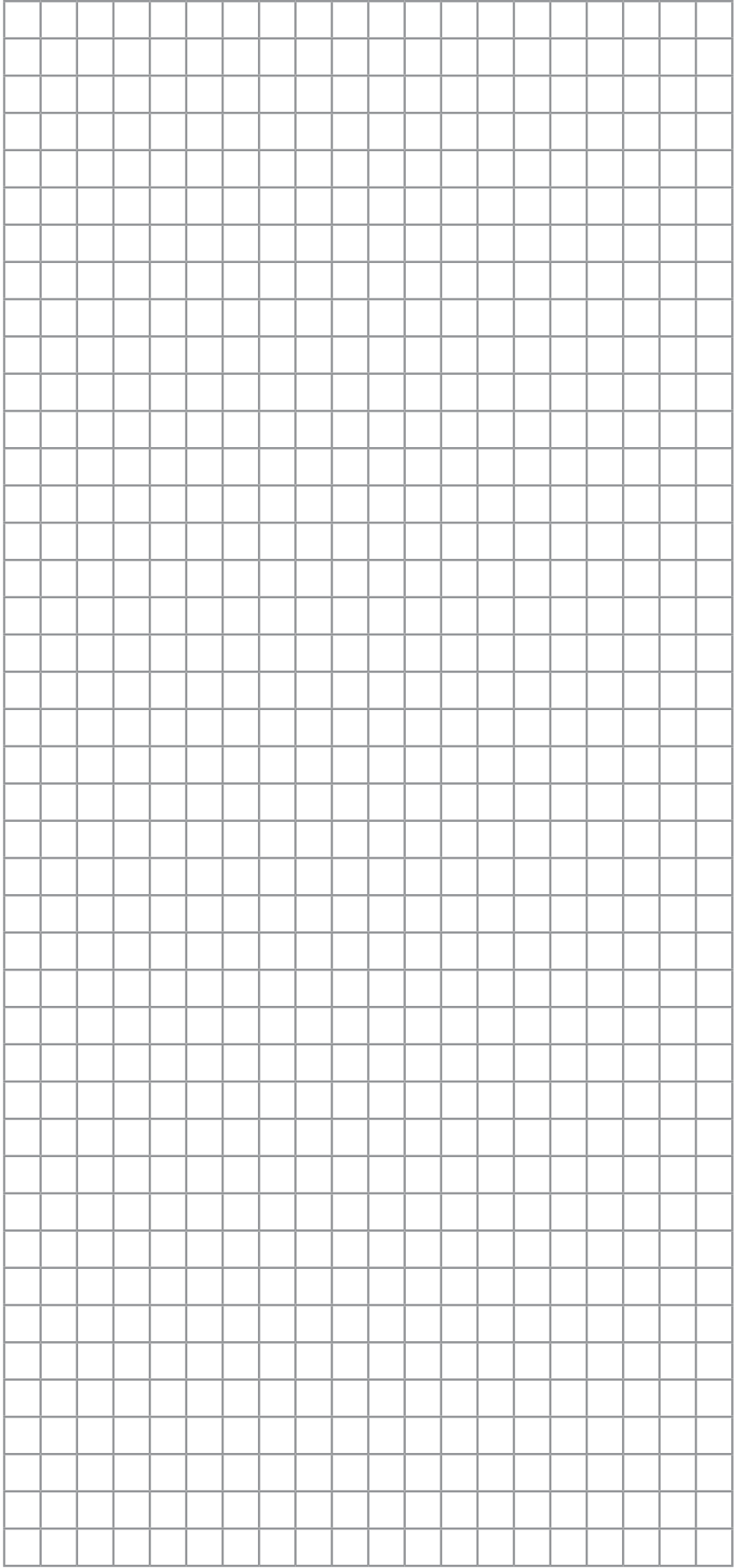
LEVEL CREATION SHEET

NAME: _____

Sketch out the level on the grid below. Plan the path(s) the player will take through it, and what obstacles, NPCs, and other things they might find. Imagine each square is 32 pixels by 32 pixels. This means if your sprite_player is 32x32, it will be the size of one square. If it is 32x64, it will be two squares tall.

DESCRIBE THE SETTING OF THIS LEVEL (WHERE IS IT?)

DESCRIBE THE GOAL OF THIS LEVEL (WHAT DOES THE PLAYER HAVE TO DO?)



NAME: _____

GAME CRITIQUE


	What is fun or interesting about the game?	How would you fix the things that aren't fun or interesting about the game?
<p>GAME #1</p> <p>Title: _____</p> <p>Artist: _____</p>		
<p>GAME #2</p> <p>Title: _____</p> <p>Artist: _____</p>		
<p>GAME #3</p> <p>Title: _____</p> <p>Artist: _____</p>		

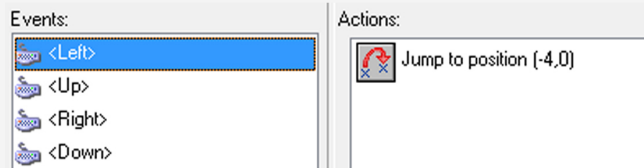
Move: Player (Free Form)


Make a player object move when arrow keys are pressed


This card is best for *Free Form Movement* or *Flight* style games because it will allow the player to press two keys at once to move diagonally.


Object_Player

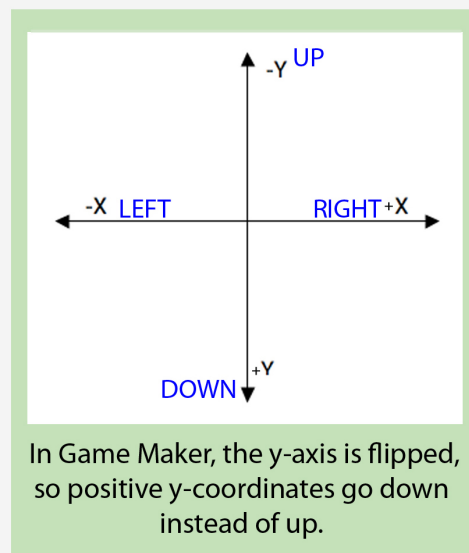
 **Event: Keyboard <Left>**
 Action: Jump to Position
 Applies to: Self
 x= -4
 y= 0
 Check Relative




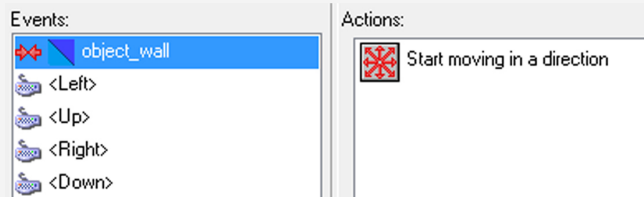
 **Event: Keyboard <Up>**
 Action: Jump to Position
 Applies to: Self
 x= 0
 y= -4
 Check Relative

 **Event: Keyboard <Right>**
 Action: Jump to Position
 Applies to: Self
 x= 4
 y= 0
 Check Relative

 **Event: Keyboard <Down>**
 Action: Jump to Position
 Applies to: Self
 x= 0
 y= 4
 Check Relative



 **Event: Collision <Wall>**
 Action: Move Fixed
 Applies to: Self
 Select Center Square
 Speed: 0
 Not Relative




Object_player should only move when keys are pressed and should stop when it hits object_wall, unless 'Solid' is not checked in object_wall properties.

Avoid: Projectile


Part 1 of 2


Create a projectile to be avoided by the player

Object_Avoid


Event: Create
 Action: Move Fixed
 Applies to: Self
 Select All Left Arrows:
 Speed: 12
 Not Relative

Events:
 Create
 Outside Room

Actions:
 Start moving in a direction


Event: Other <Outside Room>
 Action: Destroy the Instance
 Applies to: Self


Events:
 Create
 Outside Room

Actions:
 Destroy the instance



NOTE: You could set the object_avoid to collide with the object_wall and destroy the instance of self. This would keep them from going through walls.

Make the object_boss release multiple object_avoid at random


Event: Step <Step>
 Action: Test Chance
 Sides: 50
 No NOT

{same event}
 Action: Create Instance
 Select: object_avoid
 x: 0
 y: 0
 Check Relative

Events:
 Create
 Step
 object_block

Actions:
 With chance 1 out of 50 perform
 Create instance of object object

Here, Relative means that the object_avoid will be created at the current position of the object_boss. You can later select (x,y) coordinates on your boss to create object_avoid from a specific part of the sprite, such as the hand or mouth.



Test Chance is imaginary dice. The fewer sides the dice has to roll, the more object_avoids created.

More sides = Less Likely to Create Avoids
 Less Sides = More Likely to Create Avoids

Continued on Part 2 of 2







Avoid: Projectile

Part 2 of 2

Make the `object_avoid` collide with the `object_player`, with three possible outcomes







Option #1: Ends the game if the player does not avoid the projectile.

Object_Player

 Event: Collision <Avoid> Action: Show the High Score Table Select Font and Colors	Events:  <code>object_block</code>  <code>object_avoid</code>	Actions:  Show the highscore table  Restart the game
 Action: Restart the Game		







Option #2: Lowers the score if the player does not avoid the projectile.

Object_Player

 Event: Collision <Avoid> Action: Destroy the Instance Applies to: Other	Events:  <code>object_block</code>  <code>object_avoid</code>	Actions:  Destroy the instance  Set the score relative to -10
 Action: Set Score New Score: -10 Check Relative		

Option #3: Lowers the health if the player does not avoid the projectile.

Object_Player













 Event: Collision <Avoid> Action: Destroy the Instance Applies to: Other	Events:  <code>object_block</code>  <code>object_avoid</code>	Actions:  Destroy the instance  Set the health relative to -10
 Action: Set Health New Health: -10 Check Relative		

NOTE: Option #3 only works if you already have a health bar in your game. See the **Player Health Bar** card to complete the programming.




Release

Create a projectile for the player to release

Object_Release









<p> Event: Create Action: Move Fixed Applies to: Self Select Right Arrow Speed: 30 Not Relative</p>	<p>Events:  Create</p>	<p>Actions:  Start moving in a direction</p>
<p> Event: Collision <Wall> Action: Destroy the Instance Applies to: Self</p>	<p>Events:  Create  object_block</p>	<p>Actions:  Destroy the instance</p>
<p> Event: Other <Outside> Action: Destroy the Instance Applies to: Self</p>	<p>Events:  Create  object_block  Outside Room</p>	<p>Actions:  Destroy the instance</p>

Object_Player

<p> Event: Key Press <Space> Action: Create Instance Select object_release x: 0 y: 0 Check Relative</p>	<p>Events:  press <Space></p>	<p>Actions:  Create instance of object object</p>
--	---	---

Allow the object_release to destroy the object_avoid

Object_Avoid

<p> Event: Collision <Release> Action: Destroy Instance Applies to: Self</p>	<p>Events:  Create  object_block  object_release  Outside Room</p>	<p>Actions:  Destroy the instance  Set the score relative to 10</p>
<p> [same event] Action: Set Score New Score: 10 Check Relative</p>		

Each time a object_release destroys an object_avoid, the score will go up 10 points.

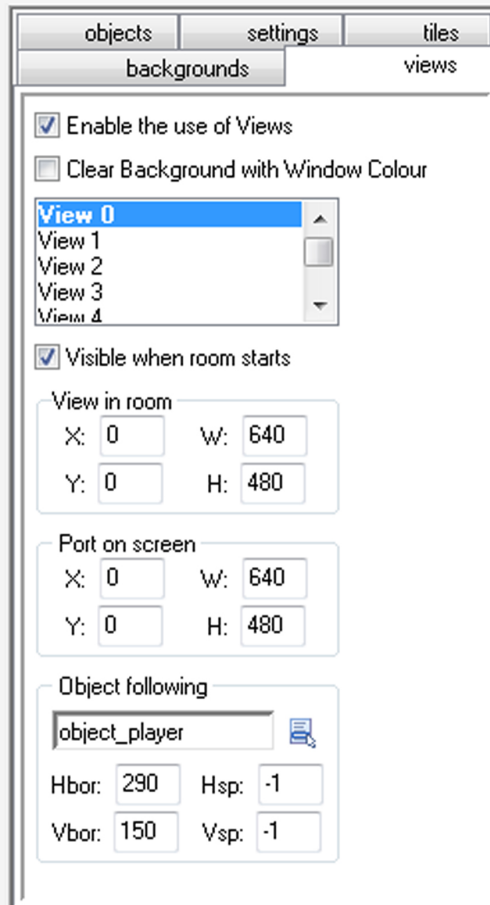
Scrolling Camera

Create a sidescroll camera with room levels larger than 640x480, so that you only see a 640x480 view of the game room during play and can hide what is to come in the game

Does the window margin at the top of your screen next to the *file name* and *GameMaker 8.1* have **(Simple Mode)** in it? If yes, you need to turn on **Advanced Mode**. If no, then skip this and proceed to Room Properties.

First, **SAVE YOUR GAME**. You will be turning on 'Advanced Mode' which will close your game. Save beforehand. Click 'File' in the Game Maker menu bar and scroll to 'Advanced Mode.' Let the program close, then you can reopen your game.

Room Properties



Click on the "Views" tab

Check **Enable the use of Views**

Make sure **View 0** is selected because your programming is set to this view.

Check **Visible when room starts**

View in Room

x: 0 W: 480
y: 0 H: 480

Port on Screen

x: 0 W: 480
y: 0 H: 480

Object Following


Select **object_player**
Hbor: 290
Vbor: 150
Hsp: -1
Vsp: -1

Change Rooms

Create a portal that will allow the player to change game rooms

You need to create a sprite_portal and object_portal. The sprite design and image size is up to you, but it shouldn't be smaller than 32x32. The object_portal has no programming

Object_Player

-  **Event: Collision <Portal>**
Action: Go to Next Room
Select a Transition




A transition is how a game visually moves to the next room.

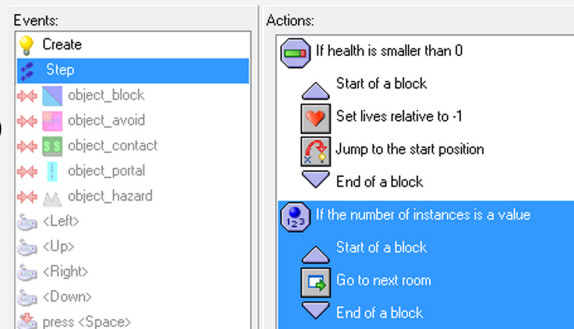
NOTE: You could also choose "Previous Room," "Restart Room," or "Different Room" which will allow you to choose a specific room. The "Next Room" portal can be used in any level except the closing level, where as a "Different Room" portal will always take you to the specified room.


You can use "Test Instance Count" to change rooms without a portal


If you wanted to change rooms when you collect all of one object or defeat the boss


Object_Player

-  **Event: Step**
Action: Test Instance Count
Object: *Your choice!*
(example: *object_collect*)
Number: 0
Operation: Equal to



- [same event]**
 Action: Start Block

- [same event]**
 Action: Go to Next Room

- [same event]**
 Action: End Block

If you have other programming in the Step Event for the player, make sure these are in order and won't mix up what was previously set.

NOTE: Be sure that you can get to all of the objects or that you don't have any outside of the room or under walls.

Contact: Collectable

Create an object for the player to collect in the game room through contact

You will need to create a sprite_collect before making an objects.

Examples of game collectables: coins, health boost, extra life, food, keys, money



Example Sprites from YOYO Games website

Object_Collect

Event: Collision <Player>

Action: Destroy the Instance

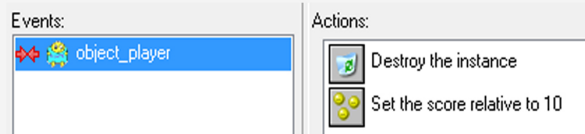
Applies to: Self

[same event]

Action: Set Score

New Score: 10

Check Relative



Each time the object_contact is collected, the score will go up 10 points. You can put as many contacts in a room as you would like.

Other ways to program a collectable:

Increase Health: Same as above with **Action: Set Health** instead of Action: Set Score. Be sure to check **Relative**.

Increase Lives: Same as above with **Action: Set Lives; value: 1: Check Relative**. Would give an extra life with each collectable.

Unlock Doors: (or "Collect all the tokens to get to the boss/next level/power up/key") See **Test Instances** or **Variable Wall** Card

Power Up: See **Power Up** Card


Avoid: Hazards

Create hazards for the player to avoid

A hazard can be anything you like. Some examples of common hazards: spikes, poison, toxic waste, fire, ice, or special walls. Create a sprite_hazard any size you'd like, then make a object_hazard, but don't add any Events or Actions. You have three options below.






Option #1: Ends the game if the player does not avoid the hazard.

Object_Player

 Event: Collision <Hazard> Action: Show High Score	<div style="border: 1px solid #ccc; padding: 5px;"> Events:  object_hazard </div>	<div style="border: 1px solid #ccc; padding: 5px;"> Actions:  Show the highscore table  Restart the game </div>
 [same event] Action: Restart Game		






Option #2: Lowers the health and restarts the level if the player does not avoid the hazard.

Object_Player

 Event: Collision <Hazard> Action: Set Health Value: -30 Check Relative	<div style="border: 1px solid #ccc; padding: 5px;"> Events:  object_hazard </div>	<div style="border: 1px solid #ccc; padding: 5px;"> Actions:  Set the health relative to -30  Jump to the start position </div>
 [same event] Action: Jump to Start		

Option #3: Lose a life and restarts the level if the player does not avoid the hazard.

Object_Player

 Event: Collision <Hazard> Action: Set Lives New Lives: -1 Check Relative	<div style="border: 1px solid #ccc; padding: 5px;"> Events:  object_hazard </div>	<div style="border: 1px solid #ccc; padding: 5px;"> Actions:  Set lives relative to -1  Jump to the start position </div>
 [same event] Action: Jump to Start		


Player Health Bar


Part 1 of 2

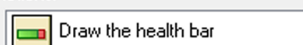
Create a health bar for the player

Create a object_health bar, but do not make a sprite. The programming will draw one for you and manage it during game play.

Object_Healthbar


Event: Draw
 Action: Draw
 x1: 10
 y1: 10
 x2: 110
 y2: 25
 Select a Background Color
 Select a Bar Color
 Not Relative

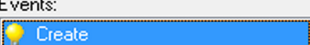
Events: 

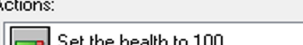
Actions: 


These (x,y) coordinates will put the health bar in the *Top Left* of the screen. For other placements, see **Coordinate Placements** card.

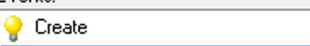

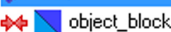
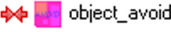
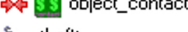


Object_Player

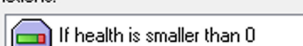
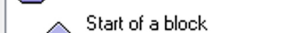

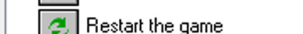
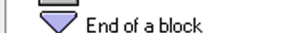
Event: Create
 Action: Set Health
 Value: 100
 Not Relative


Events: 


Actions: 

Event: Step
 Action: Test Health
 Value: 0
 Operation: Smaller Than
 No NOT


Events: 







Actions: 





{same event}
 Action: Start Block

{same event}
 Action: Show High Score
 Select Desired Visuals

{same event}
 Action: Restart the Game

{same event}
 Action: Close Block

The blocks keep the 'sub-actions' together. If you later add more actions in the 'STEP' Event, this programming won't be affected.









Continued on Part 2 of 2

Player Health Bar


Part 2 of 2

Change actions in COLLISION events with avoids, enemies, and/or hazards to affect player health

Object_Player

<p>Event: Collision  Action: Destroy the Instance Applies to: Other</p>	<p>Events:</p> <ul style="list-style-type: none">  Create  Step  object_block  object_avoid 	<p>Actions:</p> <ul style="list-style-type: none">  Destroy the instance  Set the health relative to -10
<p>[same event]  Action: Set Health Value: -10 Check Relative</p>		

The player will now lose 10 health points with every collision.

You must place the `object_healthbar` in the game room (it doesn't matter where) even though there isn't a sprite. You will see a  instead.

NOTES:

This programming is only for the *player* health cannot be used for enemy or boss health. See **Enemy/Boss Health** card.

The health bar can be place anywhere in the game room, see **Coordinate Positions** card for possibilities.

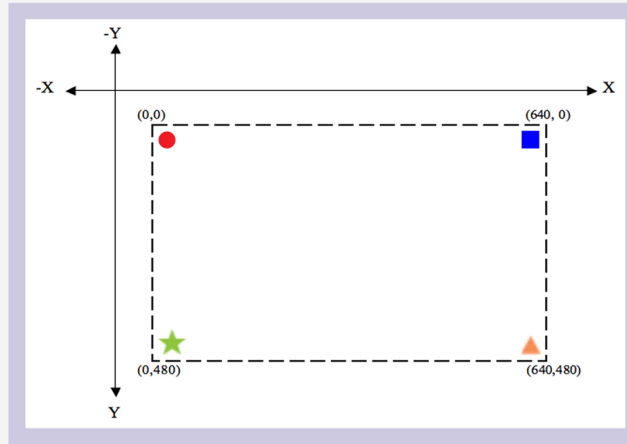
If you have LARGE ROOMS and a SCROLLING CAMERA, you must change your coordinates to include `view_xview` and `view_yview`. The health bar is not currently attached to the camera. See **Follow Coordinate Positions** card for details.

Follow Coordinate Placements

Coordinates for placements of 'DRAW' objects in a game with 'SCROLLING CAMERA'

In large game rooms, we set a scrolling camera that moves the view of the game screen with the player. Healthbar, Score, Timer and Lifebar must be attached to the *screen view* instead of the game room.

In Game Maker, the y-axis is flipped, so positive y-coordinates go down instead of up.



Object_Healthbar

Event: Draw



Action: Draw the Health Bar
Not Relative

● *Top Left*

x1: view_xview+10
y1: view_yview+10
x2: view_xview+110
y2: view_yview+25

■ *Top Right*

x1: view_xview+530
y1: view_yview+10
x2: view_xview+630
y2: view_yview+25

★ *Bottom Left*

x1: view_xview+10
y1: view_yview+455
x2: view_xview+110
y2: view_yview+470

▲ *Bottom Right*

x1: view_xview+530
y1: view_yview+455
x2: view_xview+630
y2: view_yview+470

Object_Score (or) Object_Timer (or) Object_Lifebar

Event: Draw



Action: Draw Score
Not Relative

Event: Draw



Action: Draw Variable
Not Relative

Event: Draw



Action: Draw Life Images
Not Relative

● *Top Left:*

x: view_xview+10
y: view_yview+40

■ *Top Right*

x: view_xview+535
y: view_yview+40

★ *Bottom Left*

x: view_xview+10
y: view_yview+425

▲ *Bottom Right*

x: view_xview+535
y: view_yview+425

NOTE: The score, timer, or lifebar coordinates are set to fit under the health bar if you want to place one in the same corner as health bar.

Boss/Enemy Health


Part 1 of 2

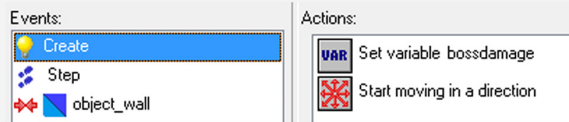
Allow the boss or enemy to take multiple "hits" before being destroyed


To do this, first decide how difficult you want the boss or enemy to be. Do you have limited or unlimited releases? If you have limited, will there be enough? For example, how will the player get through the room with 40 ammo if the boss needs 30 hits and each enemy needs 5? An enemy shouldn't be too difficult, so 2-5 is a good range. A boss should be between 10-25 depending on how hard it is to hit him.

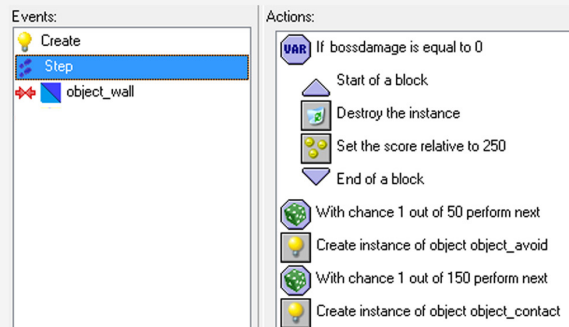
NOTE: The settings below apply for the boss and enemy, but if you have both use two variables, global.bosssdamage and global.enemydamage


Object_Boss


Event: Create
 Action: Set Variable
 Applies to: Self
 Variable: bosssdamage
 Value: 10
 Not Relative





Event: Step <Step>
 Action: Test Variable
 Applies to: Self
 Variable: bosssdamage
 Value: 0
 Operation: Equal To
 No NOT



[same event]
 Action: Start Block

[same event]
 Action: Destroy Instance
 Applies to: Self

[same event]
 Action: Set Score
 New Score: 250
 Check Relative

[same event]
 Action: End Block

NOTE: Assuming you set your boss to release avoids and contacts, you would put the above 4 actions at the top like it appears in the image.

Continued on Part 2 of 2

Boss/Enemy Health

Part 2 of 2

Event: Collision <Release>



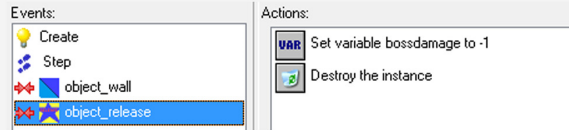
Action: Set Variable

Applies to: Self

Variable: bosssdamage

Value: -1

Check Relative



[same event]



Action: Destroy Instance

Applies to: Other

The **Create** is where the number of “hits” the boss can take is set. The **Step** is where the program checks if it needs to be destroyed. The **Collision** is what lowers the number of “hits” available.

If you wanted to, you could use **Set Health** or **Set Lives** instead of **Set Score** as a reward.


Adding Sounds

Add background tracks or sound effects to your game

To use sounds in Game Maker, you must first load them into the game resources. Click **Create Sound** in the menu bar. For now, name this **sound_test** and click **Load Sound**. Select the sound and click the green triangle (play) to hear the sound. It will automatically play your sound on repeat. Click **OK** and follow the directions below.

For a *background sound*, you will attach the sound file to an object in every level, such as a Health Bar or Lives Bar. Don't attach it to an object with multiple instances, such as Walls. If you do, every one of those walls will play the background music at the same time and it will distort the sound and the slow the game. You could make an invisible object for sound. For this example, make an object **object_backsound** with no sprite to put in the game room (you can delete it later).

Object_Backsound


Event: Create
 **Action: Play Sound**
 Sound: sound_test
 Loop: True

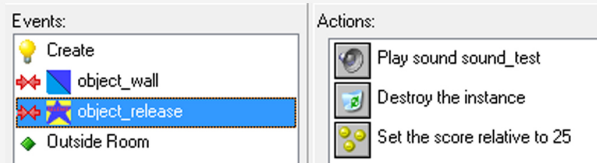


This will create a looping background sound. The test sound you chose may not be ideal. Take a look at the **Additional Resources** to find and make other sounds.

For a *sound effect*, you will attach the sound file to an existing Event/Action of your choosing. Maybe the sound emphasizes the release in the **Key Press <Space>: Create Instance** or it signals that an object was picked up in the **Collision<object_collect>: Destroy Instance**.

Object_Your Choice

Event: Your Choice
 **Action: Play Sound**
 Sound: sound_test
 Loop: False



Try to use a variation of sound effects, but pay attention to the way they overlap with each other and the background music.

WARNING: .WAV files are best for games as they are small. If you use an MP3 or large sound file, it could slow the game down.

NOTE: Be careful where you get sounds if you intend on publishing your game on websites or app stores, some are free to use and the creators have waived their rights. Some are free for personal use, but require special licenses for commercial use or distribution.

Game Maker Short Action Reference (Organized by Tabs)

- ### Move
- Move Fixed**
 - Move Free**
 - Move Towards**
 - Speed Horizontal
 - Speed Vertical
 - Set Gravity**
 - Reverse Horizontal**
 - Reverse Vertical**
 - Set Friction**
 - Jump to Position**
 - Jump to Start**
 - Jump to Random**
 - Align to Grid**
 - Wrap Screen**
 - Move to Contact
 - Bounce**
 - Set Path
 - End Path
 - Path Position
 - Path Speed
 - Step Towards
 - Step Avoiding

- ### Main1
- Create Instance**
 - Create Moving**
 - Create Random**
 - Change Instance**
 - Destroy Instance**
 - Destroy at Position
 - Change Sprite**
 - Transform Sprite*
 - Color Sprite**
 - Play Sound**

- Stop Sound
- Check Sound
- Previous Room**
- Next Room**
- Restart Room**
- Different Room**
- Check Previous
- Check Next

- ### Main2
- Set Alarm
 - Sleep
 - Set Time Line
 - Time Line Position
 - Display Message
 - Show Info
 - Show Video
 - Restart Game**
 - End Game**
 - Save Game**
 - Load Game**
 - Replace Sprite*
 - Replace Sound*
 - Replace Background*

- ### Control
- Check Empty
 - Check Collision
 - Check Object
 - Test Instance Count**
 - Test Chance**
 - Check Question
 - Test Expression
 - Check Mouse
 - Check Grid
 - Start Block**

- End Block**
- Else**
- Exit Event**
- Repeat**
- Call Parent Event**
- Execute Code**
- Execute Script**
- Comment
- Set Variable**
- Test Variable**
- Draw Variable**

- ### Score
- Set Score
 - Test Score
 - Draw Score
 - Show Highscore**
 - Clear Highscore
 - Set Lives**
 - Test Lives
 - Draw Lives
 - Draw Life Images**
 - Set Health**
 - Test Health
 - Draw Health**
 - Score Caption**

- ### Extra
- Create Part System*
 - Destroy Part System*
 - Clear Part System*
 - Create Particle*
 - Particle Color*
 - Particle Life*
 - Particle Speed*
 - Particle Gravity*

- Particle Secondary*
- Create Emitter*
- Destroy Emitter*
- Burst from Emitter*
- Stream from Emitter*
- Play CD*
- Stop CD*
- Pause CD*
- Resume CD*
- Check CD*
- Check CD Playing*
- Set Cursor*
- Open Webpage*

- ### Draw
- Draw Sprite**
 - Draw Background
 - Draw Text**
 - Draw Scaled Text*
 - Draw Rectangle**
 - Horizontal Gradient*
 - Vertical Gradient*
 - Draw Ellipse
 - Gradient Ellipse*
 - Draw Line
 - Draw Arrow
 - Set Color**
 - Set Font**
 - Set Full Screen
 - Take Snapshot*
 - Create Effect*

bold = used in book
italic = registered version