WhiteHat Jr
Live online Coding for Kids

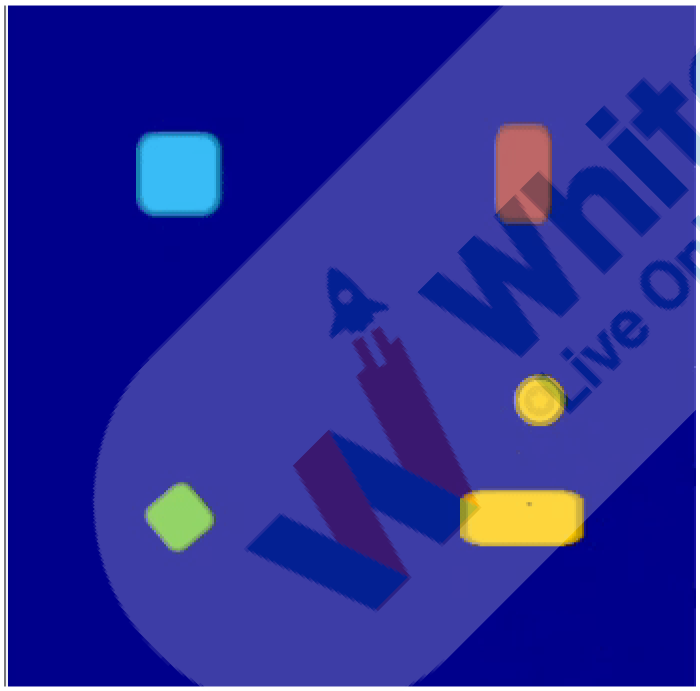
Topic	Game States	
Class Description	**Students learn to store the state (mode) of a game in a variable. Students assign different behavior to the objects in the game depending on the state of the game.**	
Class	**PRO-C6**	
Class time	**45 mins**	
Goal	• Store the state (mode) of a game in a variable. • Display different information on the screen according to the state of the game. • Use conditional programming and logical operators to assign different behaviors to the objects in the game depending on the state (mode) of the game.	
Resources Required	• Teacher Resources ○ Code.org login ○ Laptop with internet connectivity ○ Earphones with mic ○ Notebook and pen • Student Resources ○ Code.org login ○ Laptop with internet connectivity ○ Earphones with mic ○ Notebook and pen	
Class structure	**Warm-Up Slides** **Teacher-Led Activity 1** **Student-Led Activity 1** **Wra-Up Slides**	**10 mins** **15 mins** **15 mins** **5 mins**

WARM-UP SESSION - 10 mins

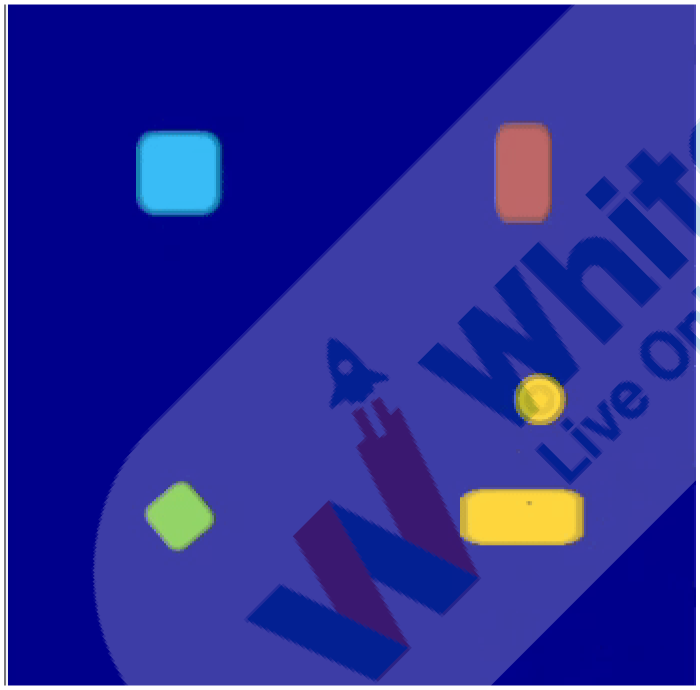
The teacher starts slideshow from slides 1 to 13
Refer to speaker notes and follow the instructions on each slide.

Activity details	Solution/Guidelines

How have you been? Are you excited to learn something new? **Run the presentation from slide 1 to slide 4.** **Following are the warm-up session deliverables:** ● Connect students to the previous class. ● Warm-Up Quiz Session.	**ESR**: Varied Response. Click on the slide show tab and present the slides.
QnA Session	
Question	**Answer**

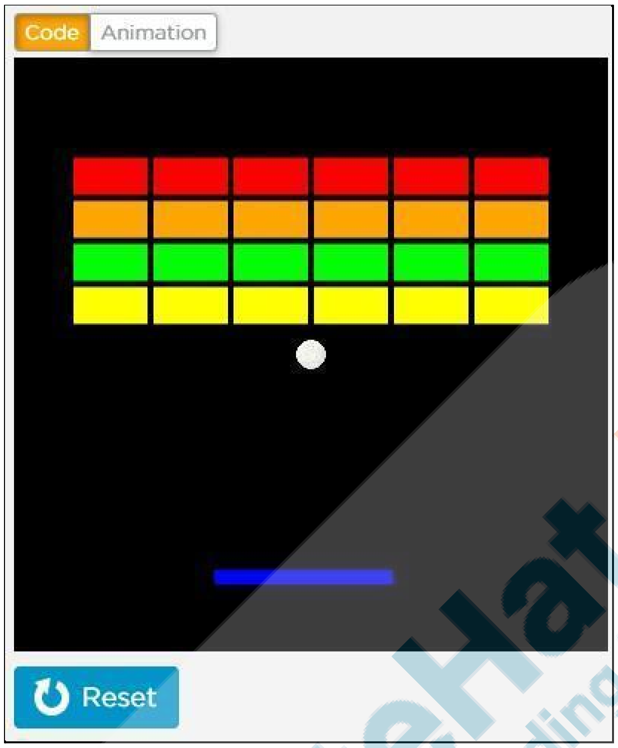
Select the block of code that checks whether the coin bounces off the blue_power sprite.

A.
`if(coin.bounceOff(blue_power)) {`
`}`
 	
B.
	A

```
if(coin.bounceOff(blue))  {

}
```
 C.
```
if(coin.bounceOff(power))  {

}
```
 D.
```
if(coin.bounce(blue_power))  {

}
```

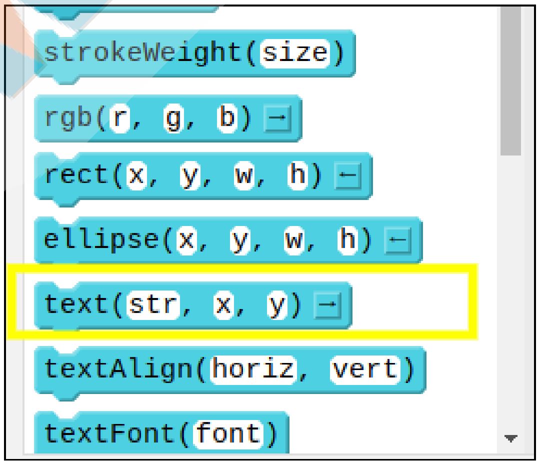
Select the line of code that increases the width of yellow_power sprite by 5 units.  A. yellow_power.width -= 5;
B. yellow_power.width = 5;
C. yellow_power.width += 5;
D. yellow_power.width ++ 5;
	C

Continue the warm-up session	
Activity details	**Solution/Guidelines**

Run the presentation from slide 5 to slide 13 to set the problem statement. **Following are the warm-up session deliverables:** ● Use of gameStates.	Narrate the story by using hand gestures and voice modulation methods to bring in more interest in students.
The teacher ends slideshow	
TEACHER-LED ACTIVITY 1 - 15 mins	
Teacher Initiates Screen Share	
<u>**CHALLENGE**</u> ● **Store the state (mode) of the game in a variable.** ● **Assign behavior to the game objects depending on the state (mode) of the game using conditional programming.**	
Teacher-led Activity (15 mins)	
Teacher Action	**Student Action**
In the last class, we added sound, score and learnt how to display text on the canvas screen. Today we will add a few more texts to the game which can help players understand how to start or restart the game. We will also learn the very important concept of **Game State** (mode).	
Let's display playing instructions or game controls as text on the canvas screen for players to understand how to start the game. *Teacher opens [Teacher Activity Link 1](#) (Breakout Game 1.5)*	



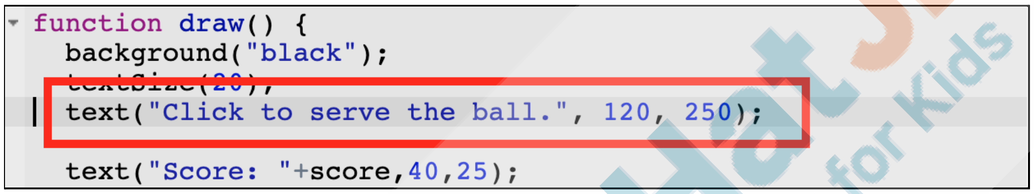
Did you observe that we have to click the 'mouse button' to start the game?	**ESR:** "Yes!"
Do you think we should put this information on the screen somewhere so that the player knows what to do?	**ESR:** Yes.
How do we do that? What instruction do we use?	**ESR:** text() instruction?



```
strokeWeight(size)
rgb(r, g, b)
rect(x, y, w, h)
ellipse(x, y, w, h)
text(str, x, y)
textAlign(horiz, vert)
textFont(font)
```

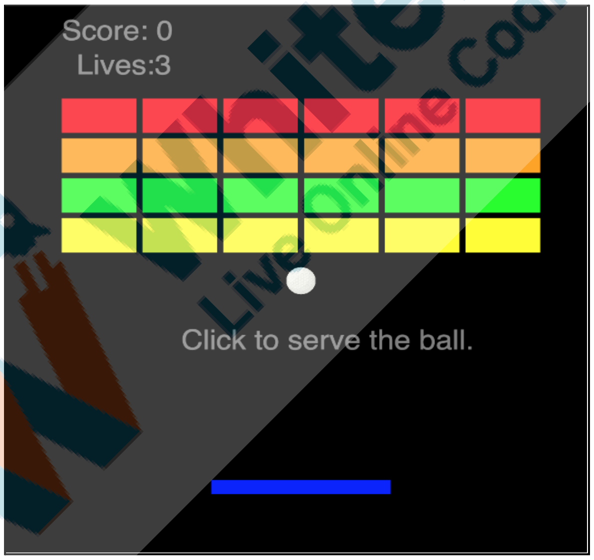
Let's do that. Let's add a ***text()*** instruction near the center which says "**Click to serve the ball.**" Remember the text() instruction, we need to give the string to be displayed, x-position and y-position. *The teacher clicks on the 'Show grid' option to decide the x and y position for the text on the canvas. (120, 250) can be one of the values.* *Encourage the child to decide the same.*	*The student observes the screen and learns the code.*

Code:



```
function draw() {
  background("black");
  textSize(20);
  text("Click to serve the ball.", 120, 250);

  text("Score: "+score,40,25);
```

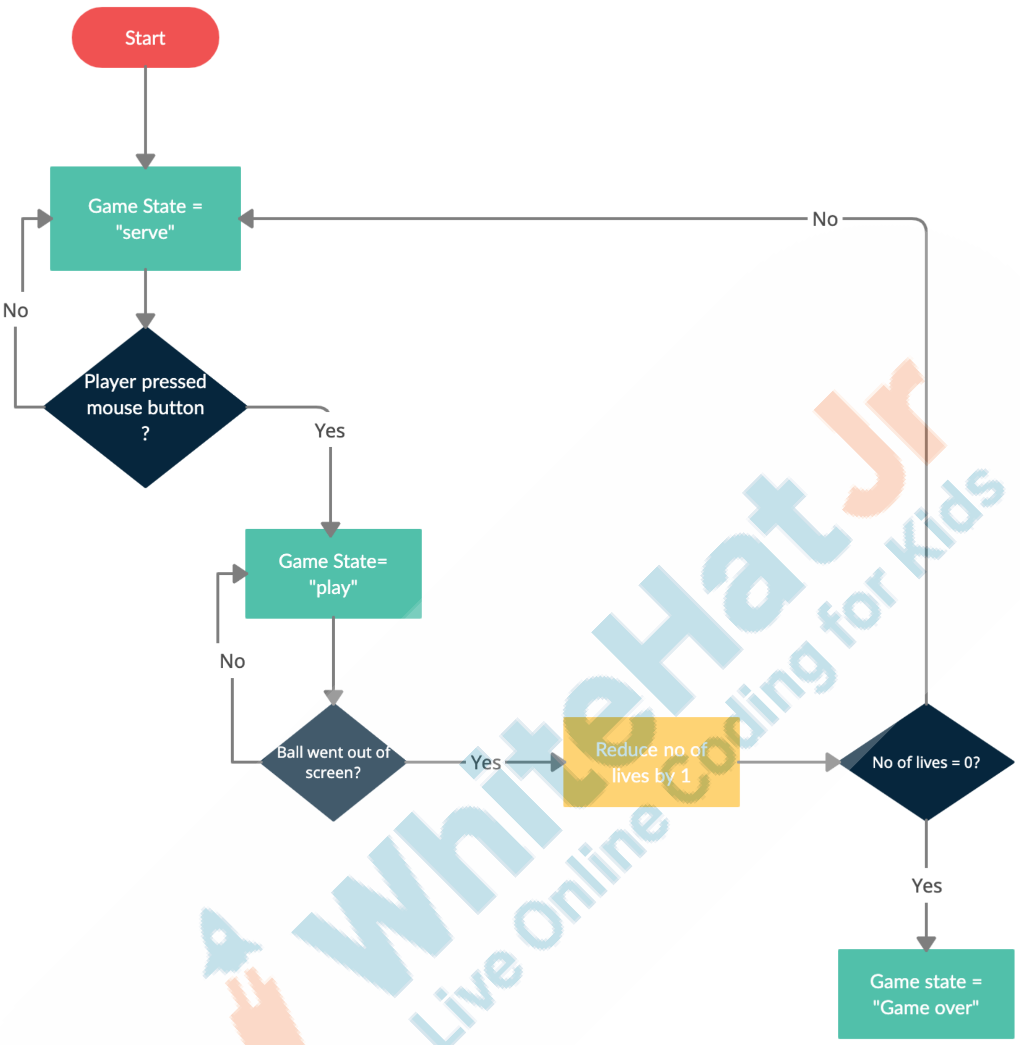
Output:



So how do we decide when to give the instruction to the computer to serve the ball and remove the same as game starts? We can do it by assigning game states to the game.	**ESR**: *varied*

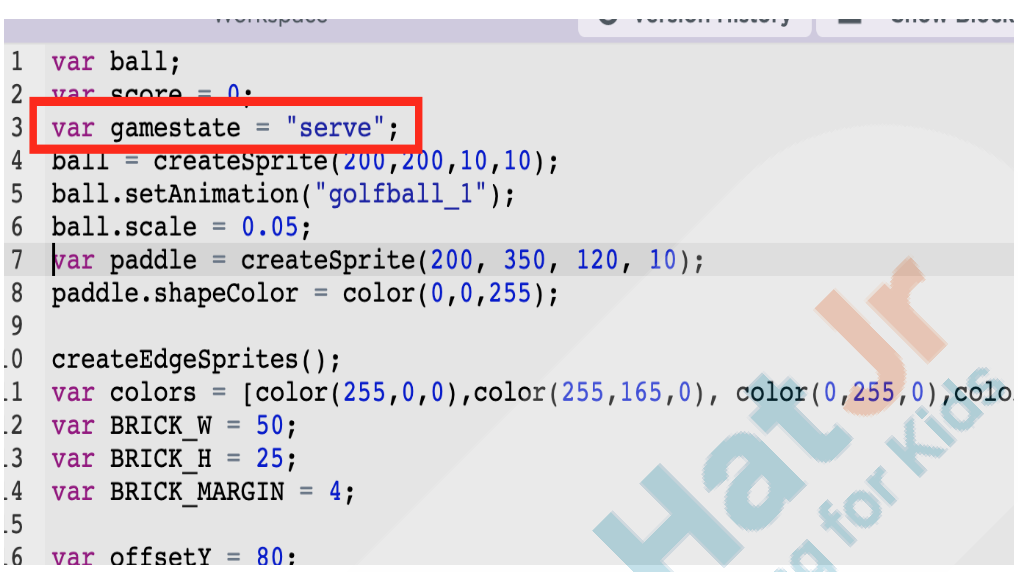
As in real life, matter changes from solid to liquid to gaseous likewise, in the game there is a change of state (mode) like “Start, Playing, and End.” For e.g. Water kept in the refrigerator changes to solid state (ice); if it is boiled, it changes to gaseous state (vapors). We will define game states for our game to figure out which state the game is in and display the relevant messages as per the current state.	
If you look at the finished game, you will realize that our game has 3 different states (modes). *The teacher opens the finished Breakout game (Full Breakout game) from the activity link and shows the game while explaining the states (modes).* ***Note-*** *The teacher can also give an example of any other real-world game to explain the states (modes) of the game. (Tennis, Ping Pong)* *Every game has 3 modes in common:* *1) The Initial Mode where the game objects are at rest.* *2) The Play Mode where the game is played.* *3) The GameOver Mode where the game comes to a halt.* ***These States or Modes always follow the same order as given above.*** ● The first state (mode) is when the ball is at the center and the user needs to “click" to serve the ball. We can call this state in the game **"SERVE" STATE (Serve Mode)**. ● The second state (mode) is when the play starts, and the ball is in motion. We can call this state	*The student listens, observes, and learns.*

(mode) in the game **"PLAY" STATE (Play Mode)**. ● The third state (mode) is when the ball goes off the screen. The game then ends and the player needs to press "Reset" to restart the game. We can call this **"GAME OVER" STATE (Game Over Mode)**. The objects in the game behave differently at different stages. Even Information displayed is different at different stages. We can store the information about the state (mode) of a game and use conditional programming to instruct the computer to behave differently for different states (modes). *The teacher quickly explains the flowchart image to the child to understand the transition between game states or take an example of a tennis game to explain 3 states of the game.*	



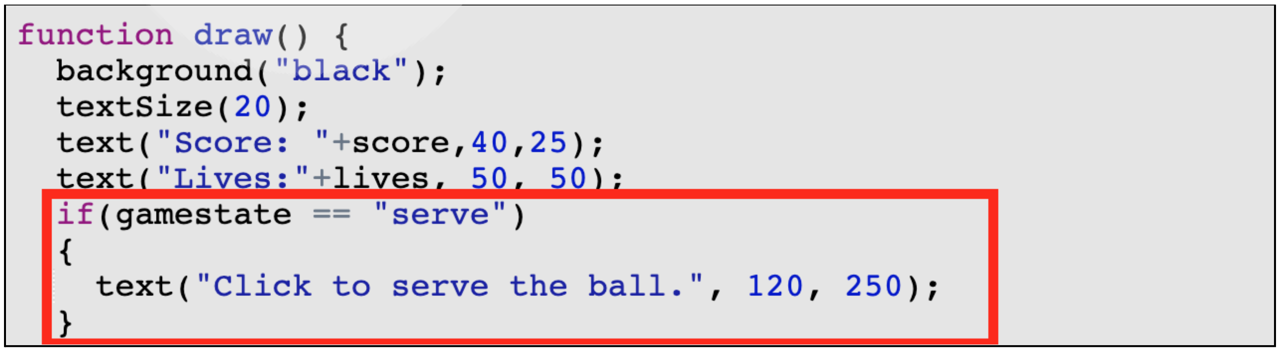
How do you think we can store information about the state (mode) of the game? What does the variable do? Yes! Let's create a variable called **gamestate (mode)** and give it a starting state (mode) of "**serve**". Whenever we are storing any text inside a variable we put it inside " ".	**ESR:** Using variables? **ESR:** It takes up some space in the computer's memory and stores the value we have allocated there.

The teacher writes code to create a variable called ***gamestate****.*	

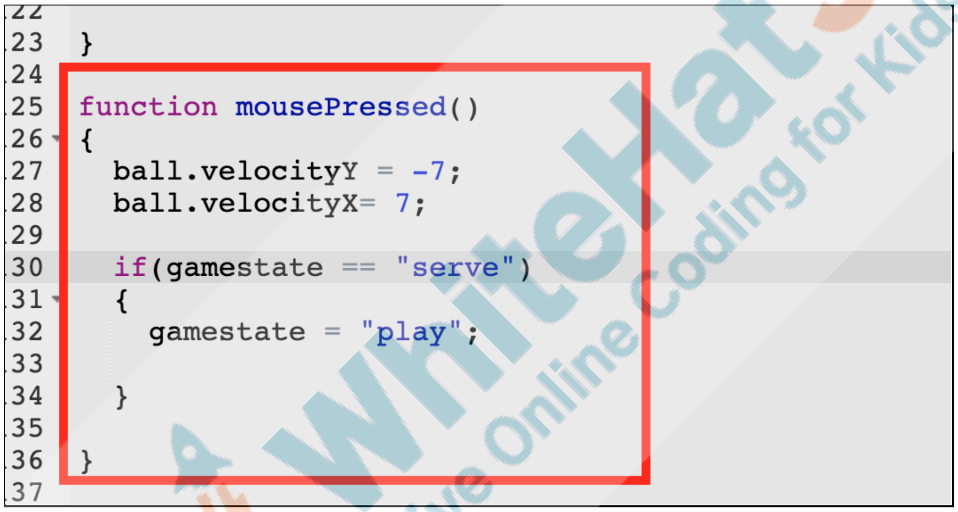


```
var ball;
var score = 0;
var gamestate = "serve";
ball = createSprite(200,200,10,10);
ball.setAnimation("golfball_1");
ball.scale = 0.05;
var paddle = createSprite(200, 350, 120, 10);
paddle.shapeColor = color(0,0,255);

createEdgeSprites();
var colors = [color(255,0,0),color(255,165,0), color(0,255,0),colo
var BRICK_W = 50;
var BRICK_H = 25;
var BRICK_MARGIN = 4;

var offsetY = 80;
```

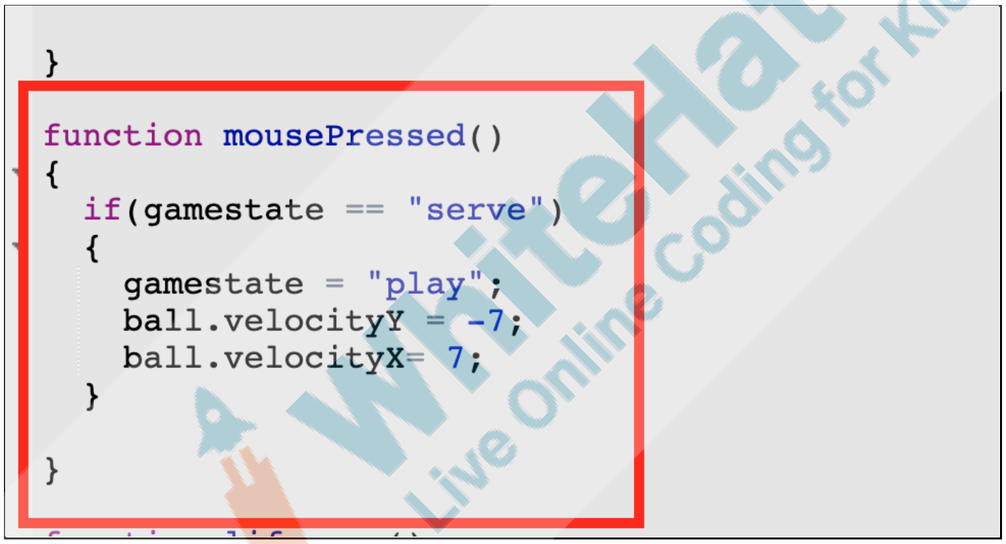
Now we have a variable called ***gamestate*** which has information about the state (mode) of the game. How do we tell the computer to display "**Click to serve the ball.**" only when the game is in the 'serve state'?	**ESR:** We use conditional programming. If gamestate (mode) is "serve", then display text.
Exactly! And we do it like this: *The teacher writes code to display text only when the game state is in the "serve" state.*	*The student observes the screen and learns the code.*



```
function draw() {
  background("black");
  textSize(20);
  text("Score:  "+score,40,25);
  text("Lives:"+lives, 50, 50);
  if(gamestate == "serve")
  {
    text("Click to serve the ball.", 120, 250);
  }
```

We also want the game state (mode) to change after the user clicks a Mouse Button. What do we want the new game state (mode) to be? Let's change the gamestate variable to "play" after the space key is pressed. Remember the values inside the variables can change! That's why they are called "variables" *The teacher writes and runs code.*	**ESR:** PLAY state? *The student observes and learns.*
	
Does the display text disappear when we press the mouse click and the game is in play mode? This is because the gamestate (mode) changes on mouse click and we have asked the computer to display text only when the game state (mode) is in 'serve the state (mode)'.	**ESR:** Yes
There is still another problem though. Pressing the mouse at any point of time changes the speed and direction of the ball. We want to give speed and direction to the ball only when the user presses spacebar in the 'serve state (mode).' How do we do that?	**ESR:** We can put velocity instructions

```
}

function mousePressed()
{
  ball.velocityY = -7;
  ball.velocityX= 7;

  if(gamestate == "serve")
  {
    gamestate = "play";

  }
}
```

We can instruct the computer as follows: If the game is in SERVE state (mode), then only serve the ball. Let's add these instructions inside the 'if condition' only. *The teacher writes and runs the code.* *The teacher presses the "mouse button" while the game is in play mode to show that the ball is not changing direction mid-way now.*	inside the 'if condition' to check if the gamestate is equal to serve.
	
Awesome, we have successfully done one transition from state "Serve" to "Play". It's your turn now. Can you code for the third mode i.e. Game Over?	
Teacher Stops Screen Share	
STUDENT-LED ACTIVITY 1 - 15 mins	
Please share your screen with me.	
• **Ask Student to press ESC key to come back to the panel** • **Guide Student to Start Screen Share** • **Teacher gets into Fullscreen**	

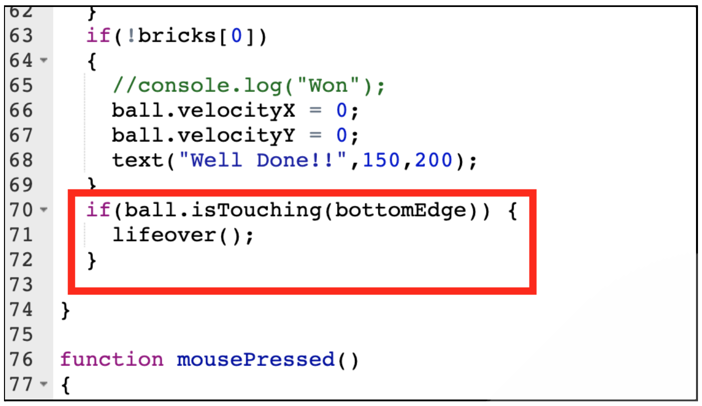
```
}
function mousePressed()
{
  if(gamestate == "serve")
  {
    gamestate = "play";
    ball.velocityY = -7;
    ball.velocityX= 7;
  }

}
```

CHALLENGE

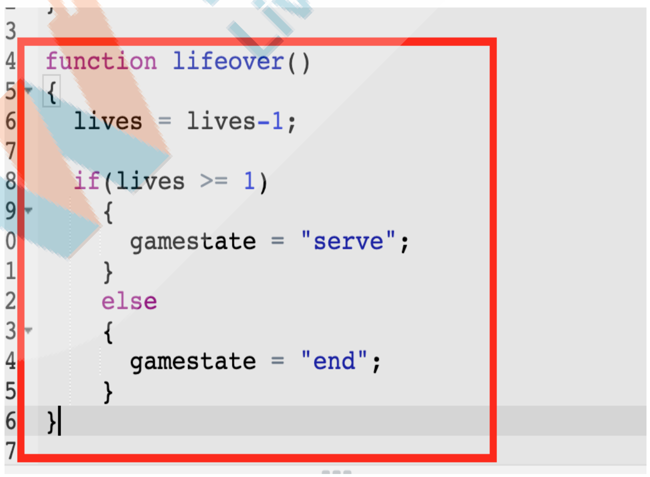
- **The student displays different text information on the screen depending on the state of the game.**
- **The student builds the 'player lives' feature in the game.**
- **The student assigns different behavior to different objects in the game depending on the state of the game.**

The teacher starts slideshow from slides 14 to 16
Refer to speaker notes and follow the instructions on each slide.

Teacher Action	**Student Action**
Let's code for the last game state i.e. Game Over mode. How many total lives/chances a player must have before making the game over?	**ESR**: Varied.
When do you think Game over state should start?	**ESR**: When all the lives of the player are over.
Great. Now can you tell me when a player will lose a life?	**ESR**: When the ball goes off the screen.
You are rocking. Finally, can you tell me a condition to check if the ball has gone out of the screen?	**ESR**: When the ball touches the bottom edge.
Awesome! The teacher guides the student to open Student Activity 1 and write the code for detecting touch between the ball and bottom edge of the canvas.	*Student opens [Student Activity Link 1]*


```
  }
  if(!bricks[0])
  {
    //console.log("Won");
    ball.velocityX = 0;
    ball.velocityY = 0;
    text("Well Done!!",150,200);
  }
  if(ball.isTouching(bottomEdge)) {
    lifeover();
  }

}

function mousePressed()
{
```

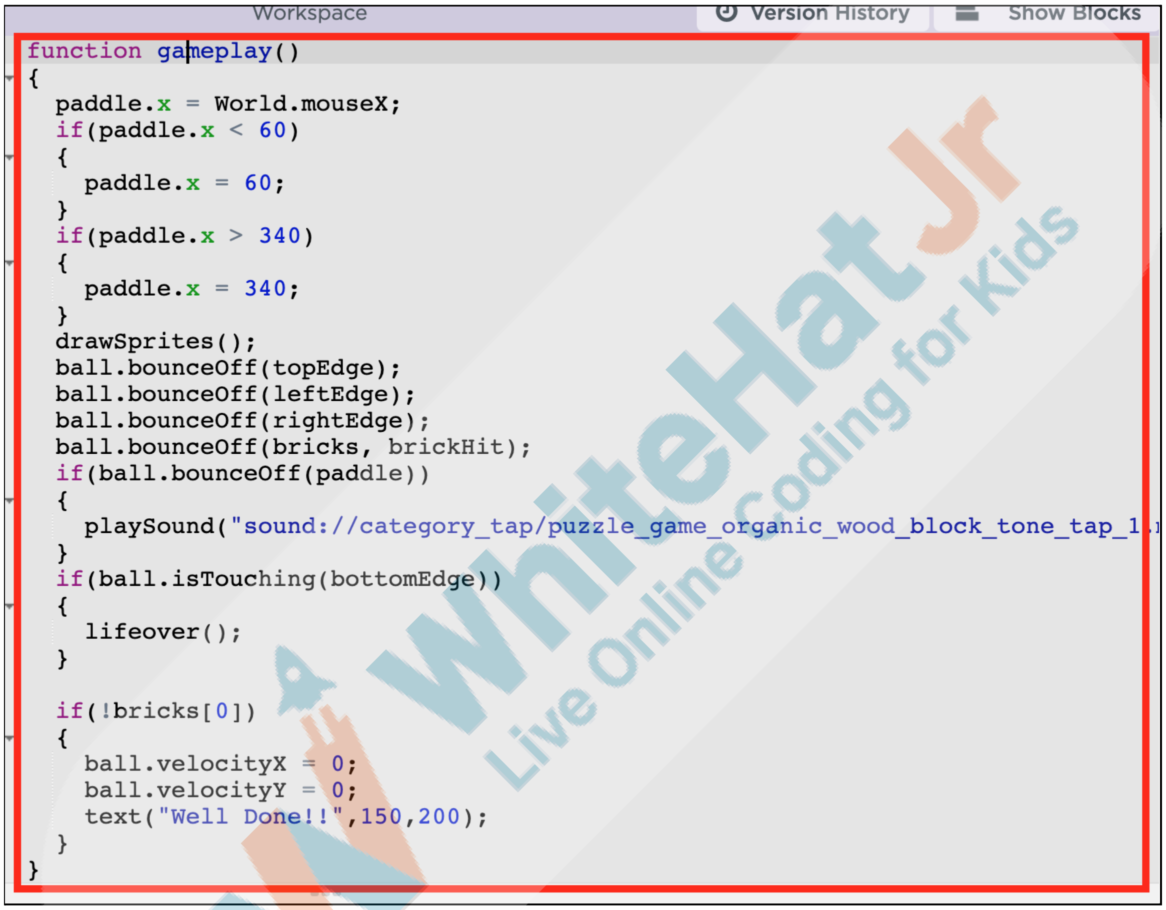
*Guide the student to create a **lifeover()** function and call the function when the ball touches the bottom edge.* *Further, inside the **lifeover()** function:* *i) reduce the number of lives by 1* *ii) if lives >= 1, change the **gamestate** variable back to **"serve"** state (mode)* *iii) else (i.e. lives = 0), change the state to "**Game Over**".* *Observe the student for any typos.*	*The student changes the **gamestate** back to "**serve**" state (mode) when the ball crosses the screen.* *The student runs the code to see if the result is as desired.*


```
}

function lifeover()
{
  lives = lives-1;

  if(lives >= 1)
  {
    gamestate = "serve";
  }
  else
  {
    gamestate = "end";
  }
}
```

Great Job! The game looks quite complete now with Serve, Play, and Game Over states.	

Let’s make our code more readable by creating a ***gameplay()*** function and moving our ‘play’ code (like paddle movement and ball bouncing off) inside that function.

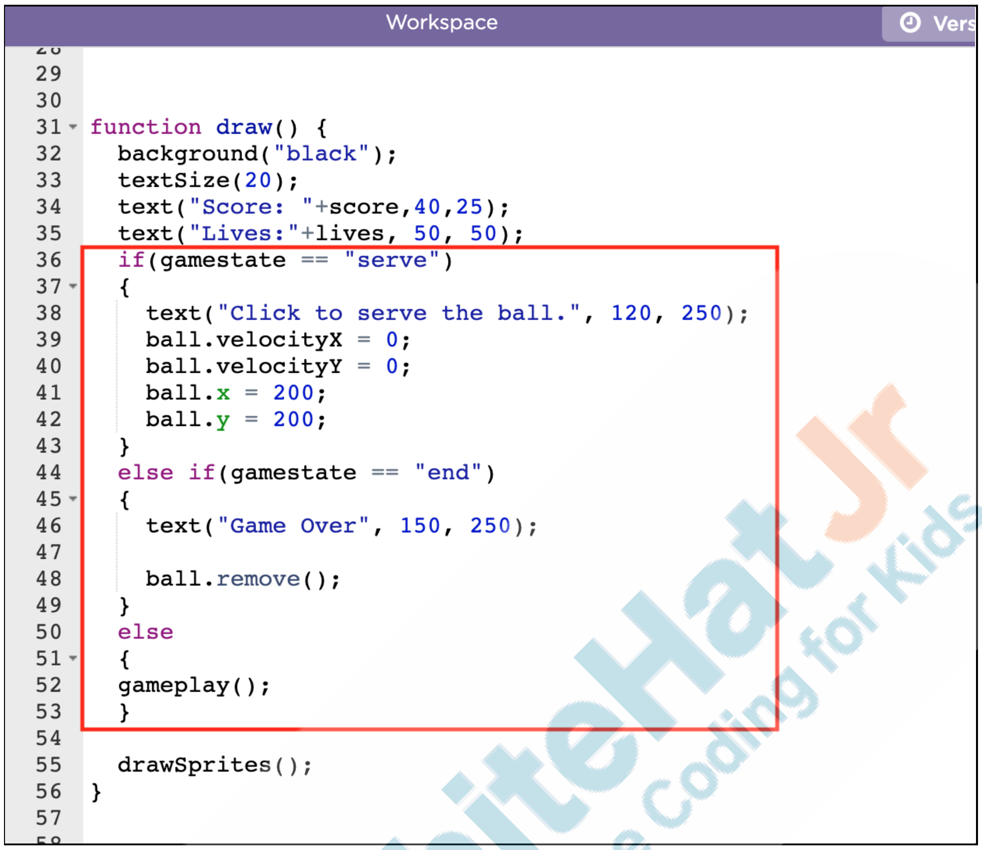
**Note: Please ensure that drawSprites(); instruction will remain inside draw() only.



```
function gameplay()
{
  paddle.x = World.mouseX;
  if(paddle.x < 60)
  {
    paddle.x = 60;
  }
  if(paddle.x > 340)
  {
    paddle.x = 340;
  }
  drawSprites();
  ball.bounceOff(topEdge);
  ball.bounceOff(leftEdge);
  ball.bounceOff(rightEdge);
  ball.bounceOff(bricks, brickHit);
  if(ball.bounceOff(paddle))
  {
    playSound("sound://category_tap/puzzle_game_organic_wood_block_tone_tap_1
  }
  if(ball.isTouching(bottomEdge))
  {
    lifeover();
  }

  if(!bricks[0])
  {
    ball.velocityX = 0;
    ball.velocityY = 0;
    text("Well Done!!",150,200);
  }
}
```

Though we have successfully detected and changed the states, we still need to display a few more text instructions for the player to know if it’s a serve state or game over state.

The teacher helps the child to complete the text instructions to be displayed for different game states on the screen.



Workspace

```
function draw() {
  background("black");
  textSize(20);
  text("Score: "+score,40,25);
  text("Lives:"+lives, 50, 50);
  if(gamestate == "serve")
  {
    text("Click to serve the ball.", 120, 250);
    ball.velocityX = 0;
    ball.velocityY = 0;
    ball.x = 200;
    ball.y = 200;
  }
  else if(gamestate == "end")
  {
    text("Game Over", 150, 250);

    ball.remove();
  }
  else
  {
  gameplay();
  }

  drawSprites();
}
```

Amazing. The game looks complete now. Next class, I will tell you how to increase the complexity of the game and apply Artificial Intelligence to the game.	

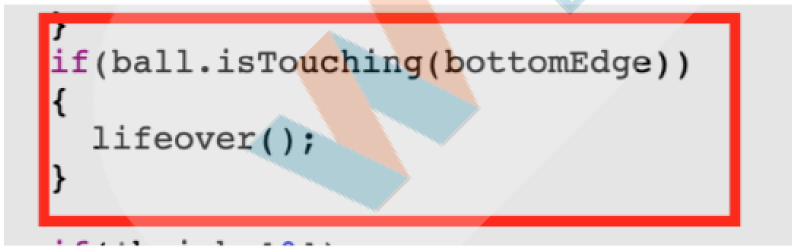
WRAP UP SESSION - 5 Mins

The teacher starts slideshow from slide 17 to slide 26

Activity details	Solution/Guidelines
Run the presentation from slide 17 to slide 26. **Following are the WARM-UP session deliverables:** ● **Explain the facts and trivias.** ● **Next class challenge.** ● **Project for the day.**	*Guide the student to develop the project and share with us.*

WhiteHat Jr
Live online Coding for Kids

● **Additional Activity.**	
QnA Session - Click on in-class quiz	
Question	**Answer**
What are the 3 common modes in a game? A. The Serve mode The Initial mode The End mode B. The Initial Mode The Play Mode The GameOver C. The Play mode The GameOver mode The Begin mode D. The Initial Mode The End The GameOver	**B**
__________ takes up some space in the computer's memory and stores the value we have allocated there. A. gameState B. variable C. sprite D. condition	**A**

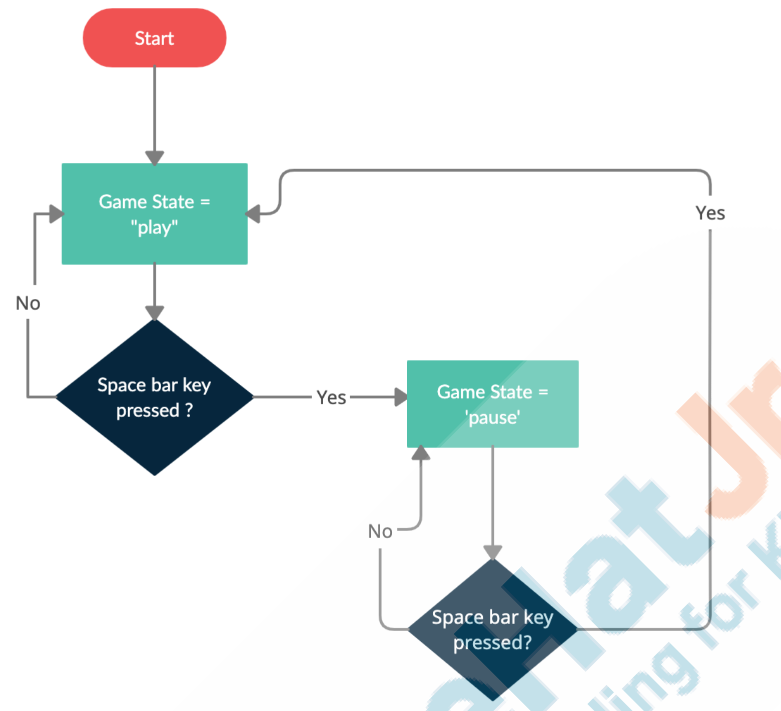
What does the following code do in a Breakout game?

`}`
`if(ball.isTouching(bottomEdge))`
`{`
`  lifeover();`
`}`
 	
A. Detecting touch between ball and bottom edge of canvas and calling a predefined function called ***lifeover()***.
B. Calling a ***lifeover()*** function when the ball touches any of the topEdge.
C. Calling a ***lifeover()*** function when the ball
	D

touches any of the edges. D. It detects touch between the ball and bottom edge of the canvas and calls a custom function for ***lifeover()***.	

Project Overview **BALL GAME** **Goal of the Project:** By Class 6, you have learned the use of conditional programming, functions, and adding game states in the project. In this project, you will have to practice and apply what you have learned so far and create a single-player ball game for a company that makes some innovative games for kids. **Story:** A company, Crafty Child, focuses on making innovative games for kids. This time they are trying to make a new version of a very popular multiplayer game, which you must have played as a team. They are trying to make a single-player ball game, using which a child can practice the game even if he/she can't go outside or doesn't have a team to play with. Can you help Crafty Child create this new, interesting game?	*The students engage with the teacher over the project.*

Teacher ends slideshow

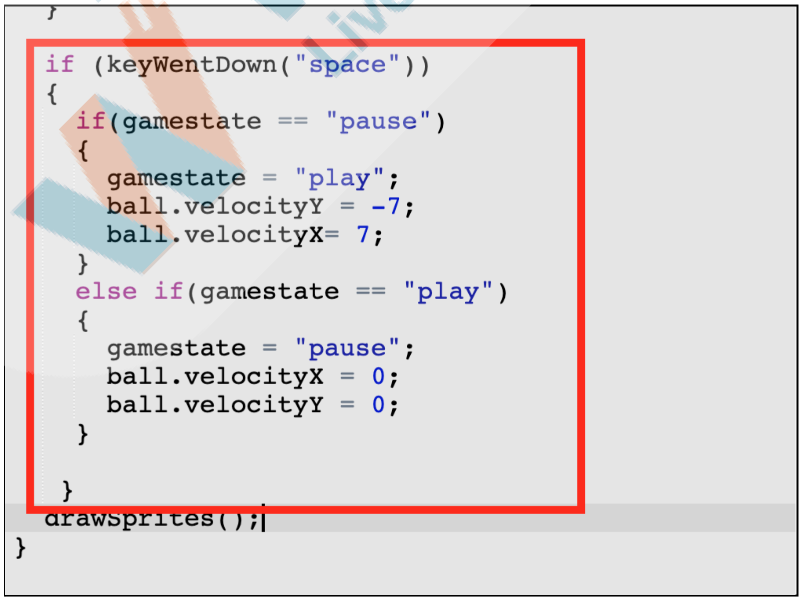
ADDITIONAL ACTIVITY (only for fast students) (To build pause resume feature in the game: 7-10 mins)	
Has it ever happened to you that you are playing a game and suddenly your parents call you for urgent work? Wouldn't it be great if we had a pause and resume functionality in our game? How do we generally pause our game or movie? Let us quickly understand how we can detect if a key is pressed. For every key press, there are 3 events fired in JavaScript: i) ***keyWentDown(key):* fires once** as soon as the key is pressed down. ii) ***keyDown(key):* keeps firing** till the key is Down. iii) ***keyWentUp(key)*: fires once** as soon as the key is released. *The teacher should show these three instructions to the child in the World tab under Toolbox.* Can you tell me which event we need to check to detect if the space key is pressed down? Great! Now that you know about key press events. Let me show you the flowchart to create pause and resume game states in our game.	**ESR**: Yes. **ESR**: Yes. **ESR**: By pressing the 'space' key or pressing 'pause' key. **ESR**: ***keyWentDown('SPACE');***



Flowchart

<span style="color:red">The teacher guides the student with Step 1.</span>

Step 1: Detect a keypress on the Space bar and switch the game state between play and pause.

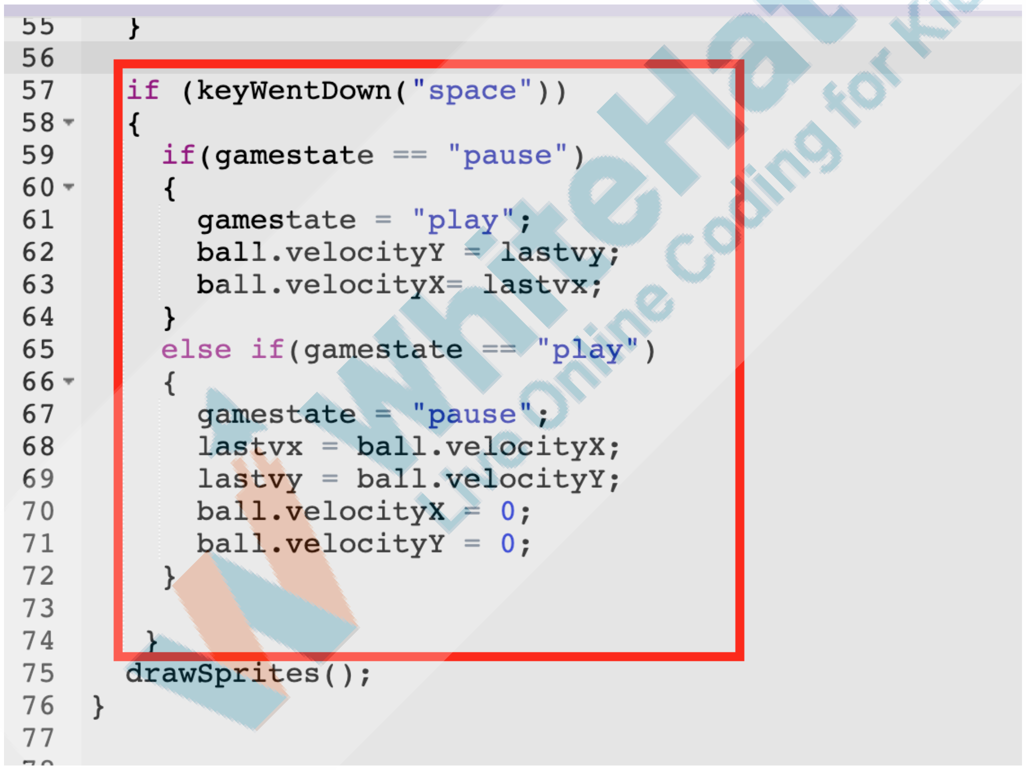


```
  }
  if (keyWentDown("space"))
  {
    if(gamestate == "pause")
    {
      gamestate = "play";
      ball.velocityY = -7;
      ball.velocityX= 7;
    }
    else if(gamestate == "play")
    {
      gamestate = "pause";
      ball.velocityX = 0;
      ball.velocityY = 0;
    }

   }
  drawSprites();
}
```

Did you notice, when we resume the game after the pause, the 'ball sprite' goes in the same direction which looks a bit odd?

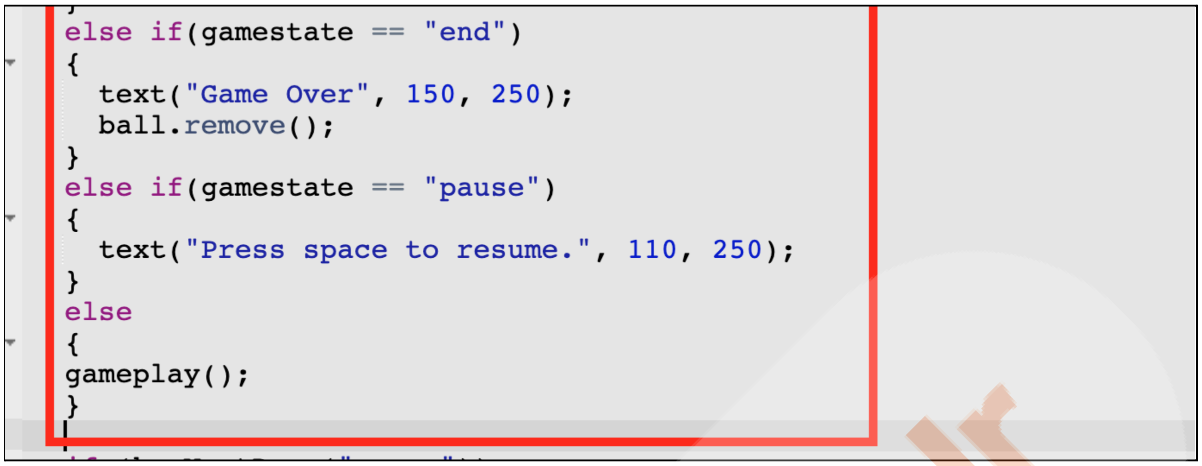
Don't you think the 'ball sprite' should resume in the same direction and with the same speed as it was before the pause?

The teacher will explain to the child how to store the last velocities in the x and y direction in another global variable. (The variables that can be used anywhere in the code)
And also assign the same velocities to the ball sprite when the game is resumed from pause.



```
  }

  if (keyWentDown("space"))
  {
    if(gamestate == "pause")
    {
      gamestate = "play";
      ball.velocityY = lastvy;
      ball.velocityX= lastvx;
    }
    else if(gamestate == "play")
    {
      gamestate = "pause";
      lastvx = ball.velocityX;
      lastvy = ball.velocityY;
      ball.velocityX = 0;
      ball.velocityY = 0;
    }

   }
  drawSprites();
}
```

Let's display a few more text instructions inside draw() for the player to know if it's a pause state.



```
else if(gamestate == "end")
{
  text("Game Over", 150, 250);
  ball.remove();
}
else if(gamestate == "pause")
{
  text("Press space to resume.", 110, 250);
}
else
{
gameplay();
}
```

Activity	Activity Name	Links
Teacher Activity Link 1	Breakout game 1.5	https://studio.code.org/projects/gamelab/hU1zppxb3QL4zX8T_Zp6P1sSRuFidLtuYJNoWUpAWfU
Teacher Activity Link 1 (Ref Code)	Game State	https://studio.code.org/projects/gamelab/hU1zppxb3QL4zX8T_Zp6PxFsCQLKupJddOmo_yg7boE
Student Activity Link 1	Game State	https://studio.code.org/projects/gamelab/ulosojbQE6JFm5ldi8fzlIJpM0n2uGy5YFTUTIl8GPs
Student Activity Link 1 (Ref code)	Breakout game 1.6	https://studio.code.org/projects/gamelab/ulosojbQE6JFm5ldi8fzlDBq9XuszQwKCXlya-z1RdM
Teacher Reference visual aid link	Visual aid link	https://curriculum.whitehatjr.com/Visual+Project+Asset/PRO_Fun+with+tech/BJFC-PRO-V3-C6-withcues.html
Teacher Reference In-class quiz	In-class quiz	https://s3-whjr-curriculum-uploads.whjr.online/5ab66c40-fb51-41b5-878f-08f21619d20c.pdf

Additional Activity Link (Ref Code)	Pause the Game	https://studio.code.org/projects/gamelab/WMmjyr0kcPtFQlKCLiVc3NZ7w1YZ0PU67YEaJy9GE0M