

Game Complexity & Automation



What is our GOAL for this MODULE?

Make our game more complex and assign Artificial Intelligence to the paddle so that it can play the game on its own.

What did we ACHIEVE in the class TODAY?

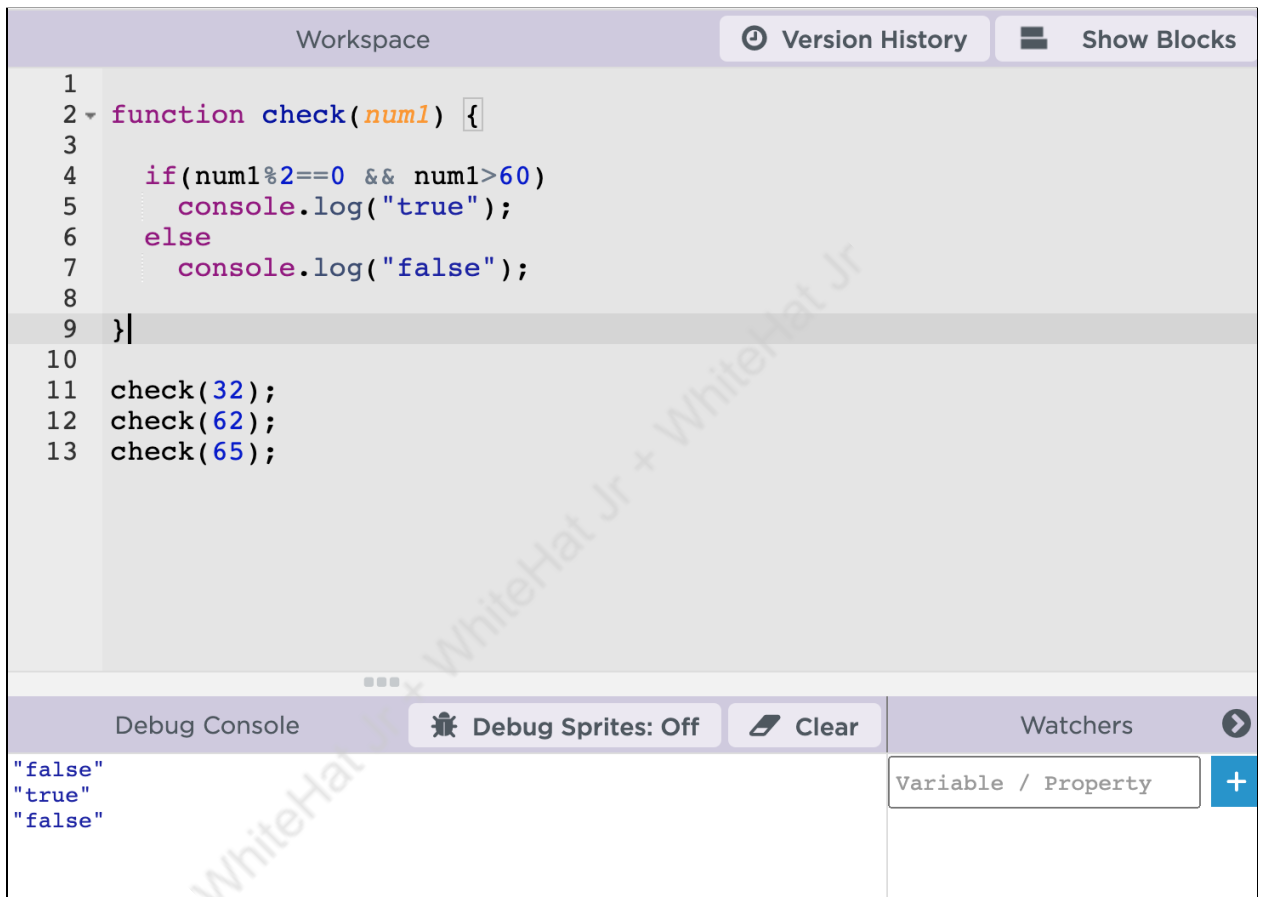
- Understood logical OR, AND, NOT operators.
- Understood these operators through coding.
- Increased the complexity in the breakout game as score increases.
- Assigned AI to the paddle by adding logic to the computer paddle so that it moves on its own.
- Assigned velocity to the bricks so they start moving in PLAY state.

Which CONCEPTS/ CODING BLOCKS did we cover today?

- Logical operators
- Combine conditions using logical operators
- Increase velocity of ball

How did we DO the activities?

1. Use conditional programming to write a program to check if the number is divisible by 2 and the number should be greater than 60.



The screenshot shows a coding workspace with a code editor and a debug console. The code editor contains the following JavaScript code:

```
1
2 function check(num1) {
3
4     if(num1%2==0 && num1>60)
5         console.log("true");
6     else
7         console.log("false");
8
9 }
10
11 check(32);
12 check(62);
13 check(65);
```

The debug console shows the output of the code execution:

```
"false"
"true"
"false"
```

The workspace also includes tabs for "Version History" and "Show Blocks", and a "Debug Sprites: Off" button.

2. Write another program if the number is divisible by 2 or it is less than 30.



```
1
2 function check(num1) {
3
4     if(num1%2==0 || num1<30)
5         console.log("true");
6     else
7         console.log("false");
8
9 }
10
11 check(15);
12 check(32);
13 check(65);
```

Debug Console

"true"
"true"
"false"

Variable / Property

3. Add the code to check if velocityY is less than 12; then only we should increase the velocity of the ball otherwise we will not increase the velocity further.

```
function brickHit(ball, brick) {  
  playSound("sound://category_hits/puzzle_game_button_04.mp3")  
  brick.remove();  
  score = score+5;  
  
  if(ball.velocityY<12)  
  { ball.velocityX *= 1.05;  
    ball.velocityY *= 1.05;  
  }  
}
```

4. Additionally, write one more condition to check that velocityY should not go below -12 as well.

```
function brickHit(ball, brick) {  
  playSound("sound://category_hits/puzzle_game_button_04.mp3")  
  brick.remove();  
  score = score+5;  
  |  
  if(ball.velocityY >-12 && ball.velocityY<12)  
  { ball.velocityX *= 1.05;  
    ball.velocityY *= 1.05;  
  }  
}
```

5. Wrote a code to assign YEach velocity to the bricks group using **setVelocityYEach()** function so that bricks move downwards.

```
function mousePressed()  
{  
  if(gamestate == "start")  
  {  
    gamestate = "play";  
    ball.velocityY = -7;  
    ball.velocityX = 7;  
    bricks.setVelocityYEach(0.2);  
  }  
}
```

6. Added automation to the game by giving the ball x position(**ball.x**) to paddle x position(**paddle.x**) so that paddle will follow the ball.

```
function gameplay()  
{  
  //paddle.x = World.mouseX;  
  paddle.x = ball.x; //automate  
  if(paddle.x < 60)  
  {  
    paddle.x = 60;  
  }  
  
  if(paddle.x > 340)  
  {  
    paddle.x = 340;  
  }  
  drawSprites();  
}
```

What's next?

In the next class, we will be building the world's toughest game.

Extend Your Knowledge:

1. Bookmark following link: it will be a reference for `group.setVelocityXEach()`:
<https://studio.code.org/docs/gamelab/setVelocityXEach/>

WhiteHat Jr + WhiteHat Jr + WhiteHat Jr