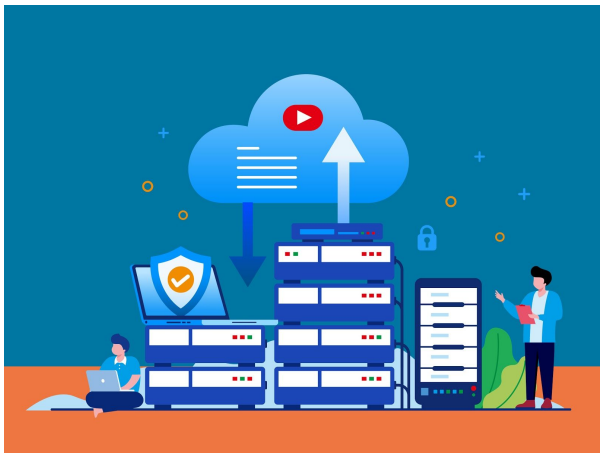
Realtime Database



What we did:

Learned the importance of using realtime database to create multiplayer games and to connect, read and write data into a remote realtime database.

How we did it:

Designed a multiplayer game: Sketched file code:

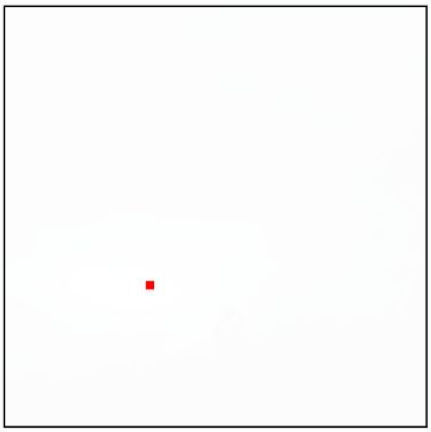
```
sketch2.js > draw
var ball;

function setup(){
    createCanvas(500,500);
    ball = createSprite(250,250,10,10);
    ball.shapeColor = "red";
}

function draw(){
    background("white");
    if(keyDown(LEFT_ARROW)){
        changePosition(-1,0);
    }
    else if(keyDown(RIGHT_ARROW)){
        changePosition(1,0);
    }
    else if(keyDown(UP_ARROW)){
        changePosition(0,-1);
    }
    else if(keyDown(DOWN_ARROW)){
        changePosition(0,+1);
    }
    drawSprites();
}

function changePosition(x,y){
    ball.x = ball.x + x;
    ball.y = ball.y + y;
}
```

Css file code:

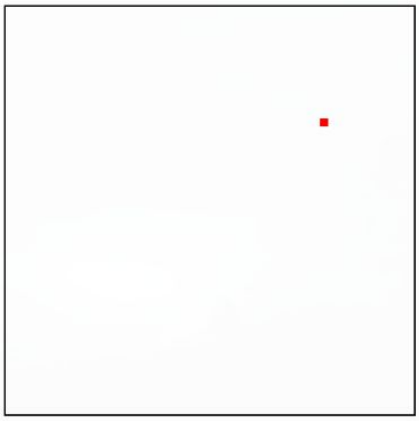
```
style.css ▸ html
1   html, body {
2       margin: 0;
3       padding: 0;
4   }
5   canvas{
6       margin-left: 25%;
7       margin-right: 25%;
8       margin-top: 100px;
9       border: 2px solid black;
10  }
11
12
```

When you opened the application in two different browsers, the ball in the two browsers moved independently. Their movements were asynchronous.

Browser one



Browser two



This happens because the ball's position in each browser is independent of the other's position. However, we could store the ball's position in a remote common database and our application reads the ball's position from the database and updates it when it changes.

Thus, we used Google Firebase's Real Time Database for this purpose.

Step 1: Go to https://console.firebase.google.com Login with your gmail id.
Step 2: Click on Create a New Project
Step 3: Enter the name of your project. Accept terms and continue
Step 4: Deny Google Analytics use
Step 5: Visit Database section of the dashboard and click on Create Database
Step 6: Create the database in test mode for now
Step 7: Add child to create nodes which can hold ball's x and y positions.
Then, connect the firebase database to our application. Import a few firebase libraries which will allow us to read and write to our database.
Step 1: Click on the "Project Overview". Choose Add to web to get started.
Step 2: Get the firebase config key

Step 3: Add this to the index.html file along with the src library for firebase database

```
<script src="https://cdnjs.cloudflare.com/ajax/libs/p5.js/0.9.0/addons/p5.dom.min.js"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/p5.js/0.9.0/addons/p5.sound.min.js"></script>
<script src="p5.play.js"></script>

<script src="https://www.gstatic.com/firebasejs/6.3.4/firebase-app.js"></script>
<script src="https://www.gstatic.com/firebasejs/6.3.0/firebase-database.js"></script>

<script>
  // Your web app's Firebase configuration
  var firebaseConfig = {
    apiKey: "AIzaSyBYV9kWljd-8zlRsYSLGIv2zBX4MhkNAo8",
    authDomain: "multiplayer-car-racing-game.firebaseapp.com",
    databaseURL: "https://multiplayer-car-racing-game.firebaseio.com",
    projectId: "multiplayer-car-racing-game",
    storageBucket: "",
    messagingSenderId: "936147099930",
    appId: "1:936147099930:web:dba47c5bb648f4ef"
  };
  // Initialize Firebase
  firebase.initializeApp(firebaseConfig);
</script>
<link rel="stylesheet" type="text/css" href="style.css"/>
</head>
<body>
<script src="sketch.js"></script>
</body>
</html>
```

To get the reference to the position of the ball in the database:

.ref() is used to refer to the location of the database value we care about.
.on() creates a listener which keeps listening to the changes in the database.
Every time a change in the database values of position (reference) occured, the readPosition function was called.
If there was any error in reading the values in the database, the showError function was called.

```
var hypnoticBall, database;
var position;


function setup(){
  database = firebase.database();
  console.log(database);
  createCanvas(500,500);

  hypnoticBall = createSprite(250,250,10,10);
  hypnoticBall.shapeColor = "red";


  var hypnoticBallPosition = database.ref('ball/position');
  hypnoticBallPosition.on("value", readPosition, showError);
}

function draw(){
  background("white");
```

In the readPosition function we read the position of the value in the database.
We assigned the x and y values of the ball position in the database to the ball sprite.

```
37
38  function readPosition(data){
39    position = data.val();
40    console.log(position.x);
41    hypnoticBall.x = position.x;
42    hypnoticBall.y = position.y;
43  }
```

Thus, when you run the program, the ball in both the browsers moved in synchronized positions when the arrow keys were pressed. However, you noticed the bug where there was an error if the keys were pressed immediately on loading:

```
function draw(){
  background("white");

  if(keyDown(LEFT_ARROW)){
    writePosition(-1,0);
  }
  else if(keyDown(RIGHT_ARROW)){
    writePosition(1,0);
  }
  else if(keyDown(UP_ARROW)){
    writePosition(0,-1);
  }
  else if(keyDown(DOWN_ARROW)){
    writePosition(0,+1);
  }
  drawSprites();

}

function writePosition(x,y){
  database.ref('ball/position').set({
    'x': position.x + x ,
    'y': position.y + y
  })
}
```

You identified the bug and fixed it by updating the ball positions only when the position variable is defined.
And you debugged the program.

```
  var hypnoticBallPosition = database.ref('ball/position');
  hypnoticBallPosition.on("value", readPosition, showError);
}

function draw(){
  background("white");
  if(position !== undefined){
    if(keyDown(LEFT_ARROW)){
      writePosition(-1,0);
    }
    else if(keyDown(RIGHT_ARROW)){
      writePosition(1,0);
    }
    else if(keyDown(UP_ARROW)){
      writePosition(0,-1);
    }
    else if(keyDown(DOWN_ARROW)){
      writePosition(0,+1);
    }
    drawSprites();
  }

}

function writePosition(x,y){
  database.ref('ball/position').set({
    'x': position.x + x ,
    'y': position.y + y
  })
}
```

What's next?
In the next class, you will be learning about structuring code prior to coding.