SPEECH RECOGNITION

What is our GOAL for this MODULE?

You have learned to apply speech recognition to create voice based web applications.This enables them to see how different technologies are connected through code.

What did we ACHIEVE in the class TODAY?

- Developed Voice Command Canvas Web Application.
- Complete Javascript code.
- Drew shapes on the canvas over a voice command

Which CONCEPTS/ CODING did we cover today?

- Initialized the window.webkitSpeechRecognition and added the functionality to it.
- Created start() function and added the functionality to it.
- Created recognition.onresult function and added the functionality to it.
- Created setup() function and added the functionality to it.
- Created draw() function and added the functionality to it.

How did we DO the activities?

Predefined HTML Code:

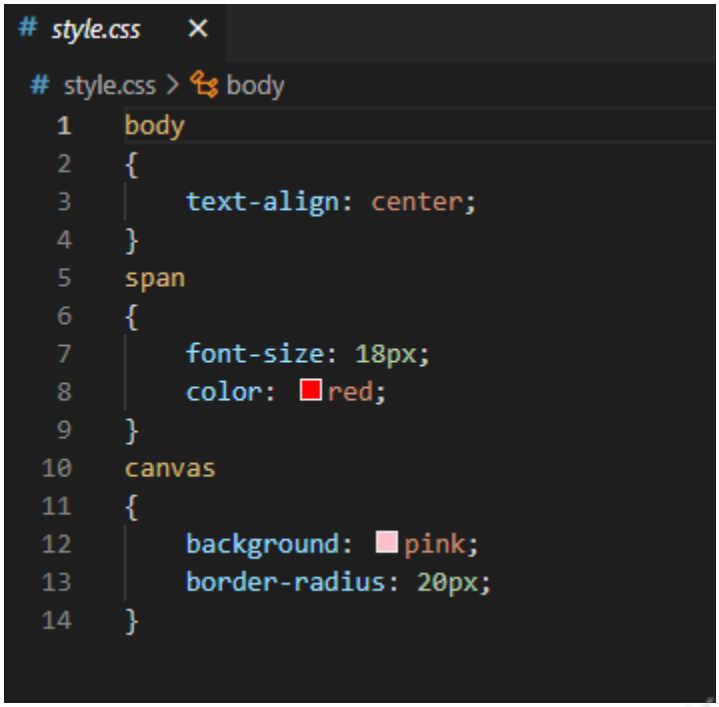
index.html

```
<html>
  <head>
    <title>Voice Command Canvas</title>
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.0/css/bootstrap.min.css">
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>
    <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.0/js/bootstrap.min.js"></script>

    <script src="https://cdnjs.cloudflare.com/ajax/libs/p5.js/1.4.0/p5.js"></script>

    <link rel="stylesheet" type="text/css" href="style.css">
  </head>

  <body>
    <h3>Voice Command Canvas</h3>
    <span id="status"></span>
    <button onclick="start();" class="btn btn-success">Draw</button>
    <br><br>
<script src="main.js"></script>
  </body>
</html>
```

In the above HTML file, we did the following:

- Included the **bootstrap** link
- Included the **p5.js** link
- Our **style.css** file link
- Our **main.js** file link
- In **h3 tag** we have written the name of the application as **Voice Command Canvas**
- Defined **id** as the **status** to the **span tag** which holds the **status**.
- Added the **onclick event** to the **button** element.
- Added two line **breaks**.

Predefined CSS Code:



In the above CSS code we set the below properties for the HTML elements:

- Aligned the **text content** of the **body tag** to **center**.
- Set the **font size** of the **span tag** as **18 pixels** and **text color** as **red.**
- Set the **background color** of the **canvas** as **pink** and **border radius** as **20 pixels**.

WhiteHat Jr
Live Online Coding for Kids

Code & Explanation

Full code of JS:

main.js

```
JS main.js > onresult
1   x = 0;
2   y = 0;
3   draw_circle = "";
4   draw_rect = "";
5
6   var SpeechRecognition = window.webkitSpeechRecognition;
7
8   var recognition = new SpeechRecognition();
9
10  function start()
11  {
12      document.getElementById("status").innerHTML = "System is listening please speak";
13      recognition.start();
14  }
15
16  recognition.onresult = function(event) {
17
18   console.log(event);
19
20  var content = event.results[0][0].transcript;
21
22     document.getElementById("status").innerHTML = "The Speech has been recognized as: " + content;
23        if(content =="circle")
24        {
25          x = Math.floor(Math.random() * 900);
26          y = Math.floor(Math.random() * 600);
27          document.getElementById("status").innerHTML = "Started drawing circle ";
28          draw_circle = "set";
29        }
30        if(content =="rectangle")
31        {
32          x = Math.floor(Math.random() * 900);
33          y = Math.floor(Math.random() * 600);
34          document.getElementById("status").innerHTML = "Started drawing rectangle ";
35          draw_rect = "set";
36        }
37  }
```

```
function setup() {
  canvas = createCanvas(900, 600);
}

function draw() {
  if(draw_circle == "set")
  {
    radius = Math.floor(Math.random() * 100);
    circle(x,y,radius);
    document.getElementById("status").innerHTML = "Circle is drawn. ";
    draw_circle = "";
  }

  if(draw_rect == "set")
  {
    rect(x,y,70,50);
    document.getElementById("status").innerHTML = "Rectangle is drawn. ";
    draw_rect = "";
  }

}
```


We studied the above code in detail:

1. As we had to draw the shapes on random positions of the canvas. So we first declared the variables x and y and assigned zero to them, like this:

```
x = 0;
y = 0;
```

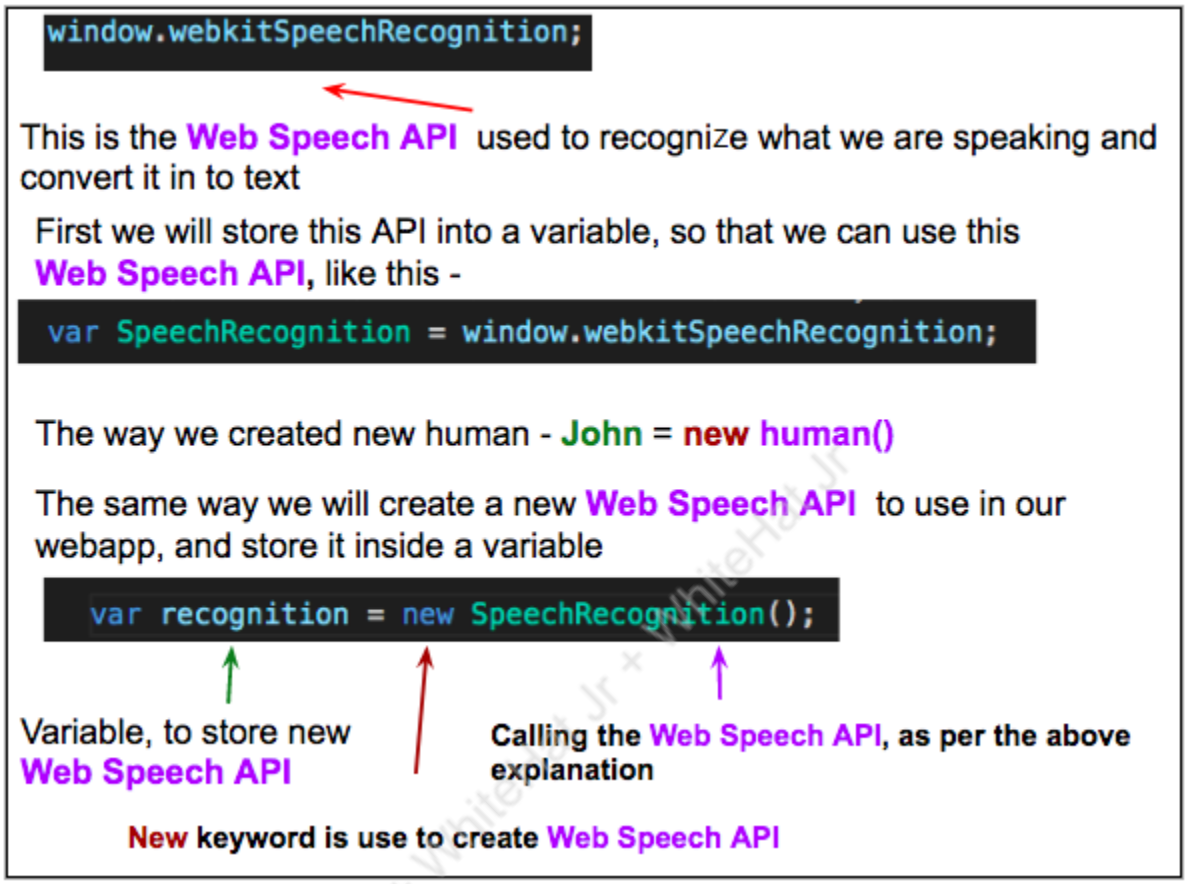
2. Then we declared two more variables which we used to set the status, depending on the shapes drawn. So in our application, we draw two shapes, a circle and a rectangle. So, we declared two variables and set them to empty string like this:

```
draw_circle = "";
draw_rect = "";
```

3. Then we used the speech to text API:

```
x = 0;
y = 0;
draw_circle = "";
draw_rect = "";

var SpeechRecognition = window.webkitSpeechRecognition;

var recognition = new SpeechRecognition();
```

Explanation of Speech to text API:



4. Then to initiate the speech recognition, we defined a function as start:

```
function start()
{
```

Whenever a **Draw** button is pressed, the system should start recognizing the speech. And inside this function, we updated the status to notify the user that the application has begun recognizing the speech of the user. So for this we updated the span tag of the html by accessing the span tag using the id as status.

5. So for this we wrote the following code:

```
document.getElementById("status").innerHTML = "System is listening please speak";
```

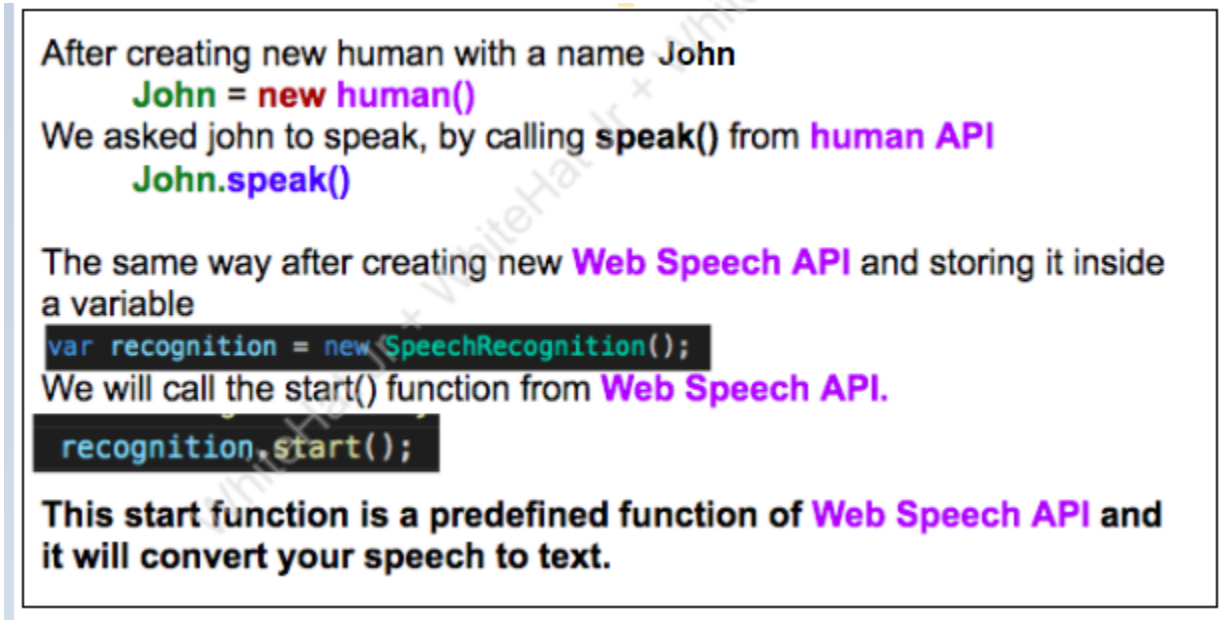
Here, **document** refers to the current html document.

Then as we have given the span tag id as the "**status**", we used the **getElementById()** function and passed the id as status inside it, like this:

```
document.getElementById("status").
```

Then using the dot operator we accessed the innerHTML, to update the value of the span tag and stored the value as **"System is listening please speak".**

6. Next, we called the recognition.start(). It is a function to initiate the recognition of the speech. Here is the explanation related to it:



Here is the complete code till now:

```
x = 0;
y = 0;
draw_circle = "";
draw_rect = "";

var SpeechRecognition = window.webkitSpeechRecognition;

var recognition = new SpeechRecognition();

function start()
{
    document.getElementById("status").innerHTML = "System is listening please speak";
    recognition.start();
}
```

7. Then we coded to display the result on the HTML page. This result was the conversion of our speech to text **(which is done by recognition.start(); in the previous step):**

```
recognition.onresult = function(event) {

console.log(event);

var Content = event.results[0][0].transcript;
console.log(Content);
```

We called the **start()** function like this:

```
recognition.start();
```

In the same way we called the **onresult** function:

```
recognition.onresult
```

The onresult function holds all the values of the speech converted to text.

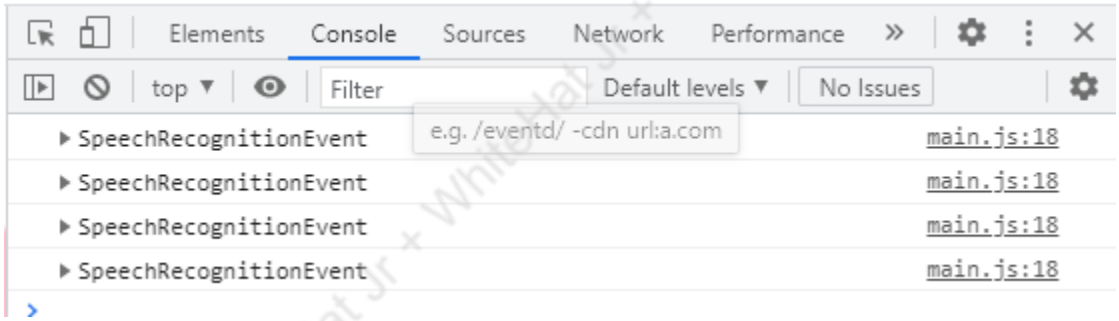
So to get these converted text from **onresult** we wrote a function like this:

```
recognition.onresult = function(event) {
```

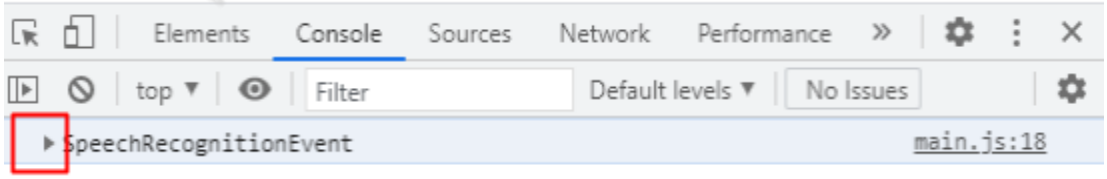
After we console this event we got the following output::

```
recognition.onresult = function(event) {

console.log(event);
```

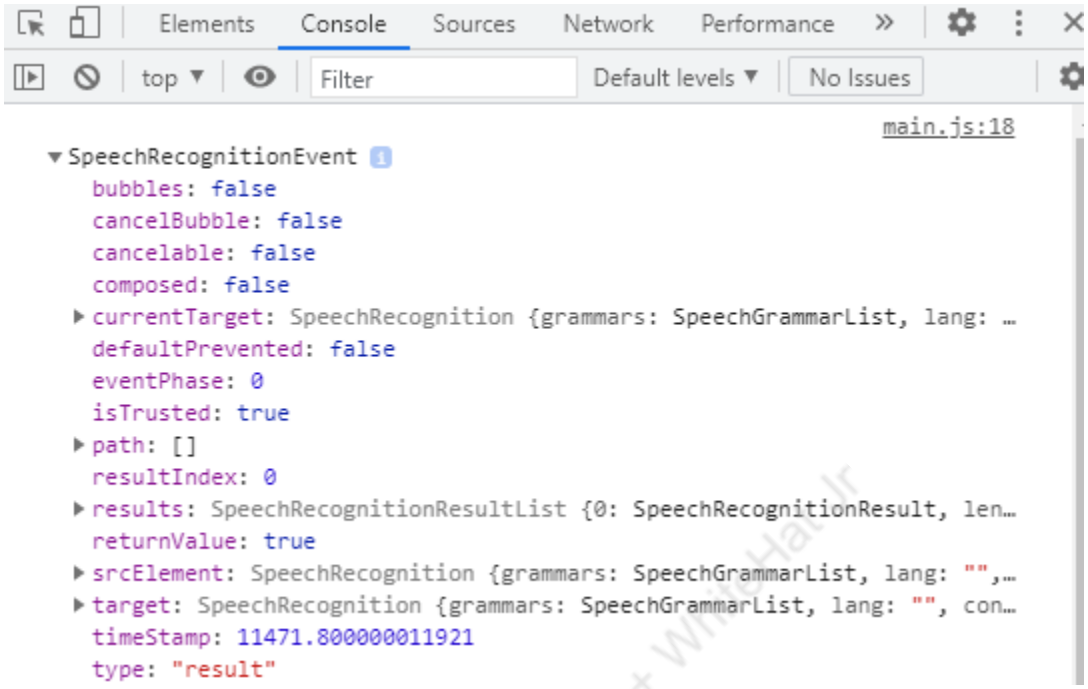
Here is how the Output looks on the console screen:



We clicked on the sign shown in the below screenshot:



We kept track of which section we have clicked. This is done because we need to know in which section our speech to text output is there. So when we find that section, we fetched the speech to text output from that section:



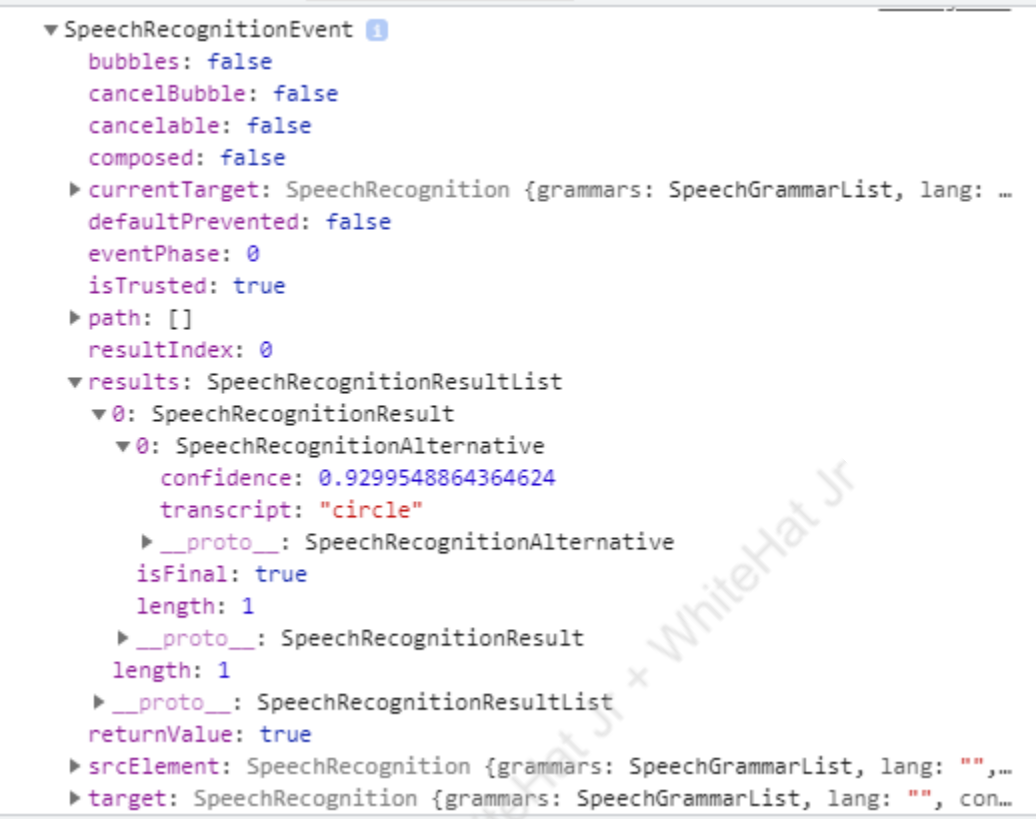
We see too much data, but we don't need this data, so click on the sign as shown in the screenshot.

8. We keep a track of which section we clicked. In the below screenshot, we have clicked on the **results**. Thus, we can say that inside the **event** we have clicked on the **results**. In the code it is written as - **event.results**:

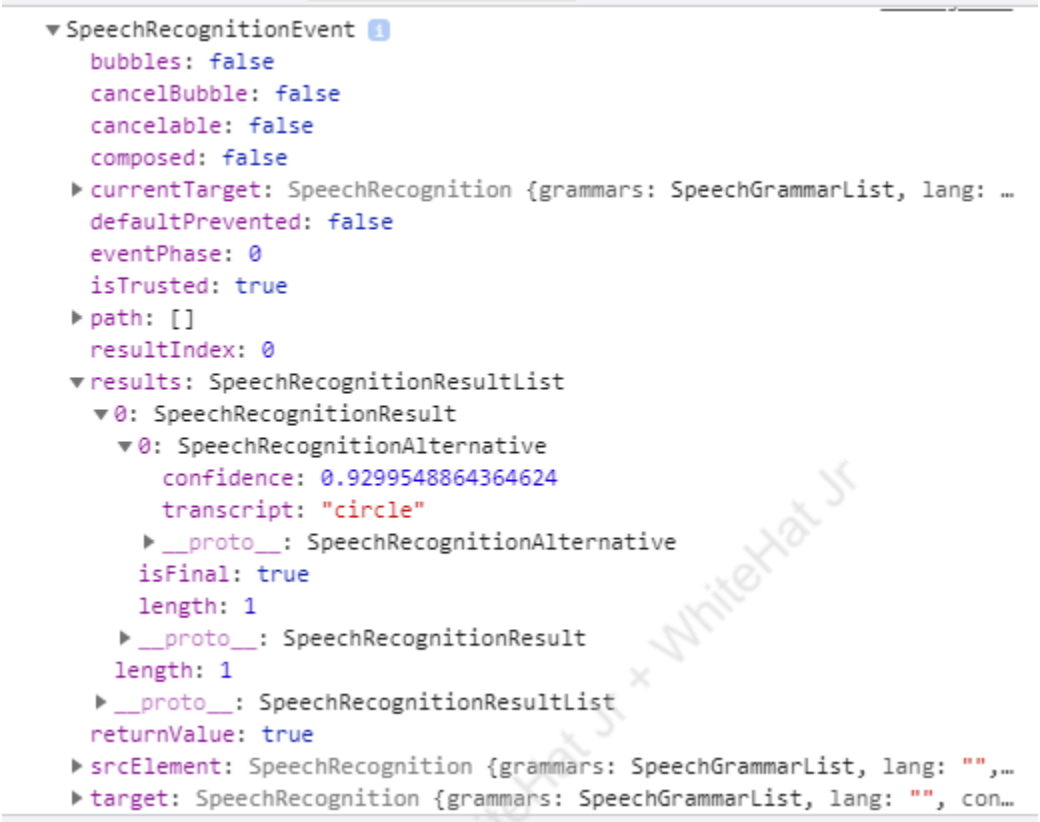
9. After clicking on the results, we see too much data. But we don't need all this data. So, again we clicked on the sign as shown in the below screenshot:

10. **We keep a track on which section we have clicked. We clicked on the results which are present inside the event. In the code it will be written as - event.results. After that, we clicked on the "0" which is present inside the results. Thus, in the code it will be written as - event.results[0].**

11. After clicking on "0" which is inside the results, we see too much data. But we don't need all this data. Again click on the sign:



12. In the below screenshot, we have clicked on the “0” which is inside the “0”. So, we can say that inside the event we have clicked on the results, inside the results we have clicked on “0”, and inside “0” we have again clicked on the “0”. In the code, it is written as - **event.results[0][0]**:



At last we reached where we wanted to. The marked thing above (circle) has the text of our speech, which we wanted to fetch.

13. **We have clicked - event -> results -> 0 -> 0. In the code, it is written as - event.results[0][0].** Now, to fetch the text from the **transcript** which is inside **0,** the code is **event.results[0][0].transcript.**

    Now we know this **event.results[0][0].transcript** has our speech to text output. So, we store this inside a variable like this:

```
var content = event.results[0][0].transcript;
```

14. After getting the speech converted to text, we updated that text with the status by using the **document.getElementById()** function. And pass the Id as **status** inside this function and using the **innerHTML**, we updated the status

as “**The Speech has been recognized as:** ” and concatenated the converted speech to text present with the **content** variable like this:

```
x = 0;
y = 0;
draw_circle = "";
draw_rect = "";

var SpeechRecognition = window.webkitSpeechRecognition;

var recognition = new SpeechRecognition();

function start()
{
    document.getElementById("status").innerHTML = "System is listening please speak";
    recognition.start();
}

recognition.onresult = function(event) {

 console.log(event);

var content = event.results[0][0].transcript;

    document.getElementById("status").innerHTML = "The Speech has been recognized as: " + content;
```

15. Then, we checked what shape the user wanted to draw, and accordingly drew the shape on the canvas. So the speech of the user is being stored in the variable **content**. And if the speech is a **circle**, the **content** variable has the value as a **circle**. So for this we used the **if** condition, to check whether the value of the **content** variable is **circle**, like this:

```
if(content =="circle")
{
```

16. And if the value inside the **content** variable is “**circle**”, then inside this **if** condition, we wrote code for generating 2 random numbers for placing the **circle** randomly on the canvas.

    So for this, we first got the **random x coordinate** along the **width** of the **canvas** which is **900**. So we will use the **Math.floor()** function and inside it we

passed the **Math.random()** function and multiplied it by **900**.

Similarly, we get the **random y coordinate** along the **height** of the **canvas** which is **600**. So again we used the **Math.floor()** and inside it we passed the **Math.random()** function and multiplied it by **600**.

The code looks like this:

```
if(content =="circle")
{
  x = Math.floor(Math.random() * 900);
  y = Math.floor(Math.random() * 600);
```

17. Then, we updated the **status** by using the **document.getElementById()** and setting the text as "**Started drawing circle**". This was because if the circle is not drawn then the '**Started drawing circle**' message gets displayed. This helped us to find in which "**if**" condition our code is getting stuck. It's just for the debug purpose:

```
if(content =="circle")
{
  x = Math.floor(Math.random() * 900);
  y = Math.floor(Math.random() * 600);
  document.getElementById("status").innerHTML = "Started drawing circle ";
```

Then we updated the value of the variable **draw_circle** which we have created earlier in the class as "**set**":

```
if(content ==="circle")
{
  x = Math.floor(Math.random() * 900);
  y = Math.floor(Math.random() * 600);
  document.getElementById("status").innerHTML = "Started drawing circle ";
  draw_circle = "set";
}
```

18. Then, similarly we checked if the user wants to draw a rectangle. So for this we checked if the variable **content** has the word **rectangle** using the **if** condition:

```
if(content ==="rectangle")
```

19. And if the value inside the content variable is "**rectangle**", then inside this **if** condition we generate a random number and draw a rectangle randomly on the screen.

    So for this, we first get the **random x coordinate** along the **width** of the **canvas** which is **900**. So we used the **Math.floor()** function and inside it we passed the **Math.random()** function and multiplied it by **900**.

    Similarly, we get the **random y coordinate** along the **height** of the **canvas** which is **600**. So again we used the **Math.floor()** and inside it we passed the **Math.random()** function and multiplied it by **600**.

    The code looks like this:

```
if(content ==="rectangle")
{
  x = Math.floor(Math.random() * 900);
  y = Math.floor(Math.random() * 600);
```

20. Then, we updated the **status** by using the **document.getElementById()** and set the text as "**Started drawing rectangle**". This is because if the circle is not drawn then the '**Started drawing rectangle**' message is displayed. This helps us to find in which "**if**" condition our code is getting stuck. It's just for the debug purpose:

```
if(content =="rectangle")
{
  x = Math.floor(Math.random() * 900);
  y = Math.floor(Math.random() * 600);
  document.getElementById("status").innerHTML = "Started drawing rectangle ";
```

Then we updated the value of the variable **draw_circle** which we have created earlier in the class as "**set**":

```
if(content =="rectangle")
{
  x = Math.floor(Math.random() * 900);
  y = Math.floor(Math.random() * 600);
  document.getElementById("status").innerHTML = "Started drawing rectangle ";
  draw_rect = "set";
}
```

So the full code of **recognition.onresult function** will look like this:

```
recognition.onresult = function(event) {

  console.log(event);

var content = event.results[0][0].transcript;

  document.getElementById("status").innerHTML = "The Speech has been recognized as: " + content;
    if(content =="circle")
    {
      x = Math.floor(Math.random() * 900);
      y = Math.floor(Math.random() * 600);
      document.getElementById("status").innerHTML = "Started drawing circle ";
      draw_circle = "set";
    }
    if(content =="rectangle")
    {
      x = Math.floor(Math.random() * 900);
      y = Math.floor(Math.random() * 600);
      document.getElementById("status").innerHTML = "Started drawing rectangle ";
      draw_rect = "set";
    }
}
```

21. Then we wrote the code to create a canvas. For this, we created a function called **setup().** And inside this function, using the **createCanvas()** function, we created a function with the width and height of (**900** x **600**) and stored it in a variable named **canvas**. So the code looks like this:

```
function setup() {
    canvas = createCanvas(900, 600);
}
```

22. Then, we wrote code to actually draw the shape on the canvas. For this, we defined the **draw()** function. And inside this function, we checked if the value of the **draw_circle** is equal to **set**. This is because if the value of the variable **draw_circle** is **set**, then only we had to draw the circle on the canvas:

```
function draw() {
  if(draw_circle == "set")
```

23. Then , to draw the circle we need to get the radius of the circle. So, we generated a random number and used it as radius which resulted in drawing a circle with random radius value. For this we used the **Math.floor()** function, and inside it we added the **Math.random()** function and then multiplied it by **100**. Then we stored the result in the variable **radius**:

```
radius = Math.floor(Math.random() * 100);
```

We have multiplied with 100 so the radius value will vary from 0 to 100. However, any other number can be used.

24. Then, we used the **circle()** function. This **circle()** function takes 3 parameters, **x** and **y** coordinates, and the **radius** of the circle:

```
radius = Math.floor(Math.random() * 100);
circle(x, y, radius);
```

25. Then using the **document.getElementById()** we updated the status Id of the span tag to text as "**Circle is drawn.**", and we set the variable **draw_circle** as **empty**.

If we don't set it as empty, then again the circle is drawn as **draw()** function is called after every millisecond and we don't want this. We wanted the circle to be drawn only when the user says it. Hence we set **draw_circle** to **empty**:

```
function draw() {
  if(draw_circle == "set")
  {
    radius = Math.floor(Math.random() * 100);
    circle(x, y, radius);
    document.getElementById("status").innerHTML = "Circle is drawn. ";
    draw_circle = "";
  }
```

26. Similarly to draw the rectangle, we checked if the value of the **draw_rect** is equal to **set**. This is because if the value of the variable **draw_rect** is **set**, then only we had to draw the rectangle on the canvas:

```
if(draw_rect == "set")
```

27. Then, to draw the rectangle we used the **rect()** function. This **rect()** function takes 4 parameters, **x and y coordinates (**which are the random values which we have generated in **recognition.onresult function**) and **width** and **height** as the parameters:

```
rect(x,y,70,50);
```

28. Then using the **document.getElementById()** we updated the status Id of the span tag to text as "**Rectangle is drawn.**", and we set the variable **draw_rect** as **empty**.

    If we don't set it as empty then again the rectangle is drawn as **draw()** function is called after every millisecond and we don't want this. We want the rectangle to be drawn only when the user says it. Hence, we set **draw_rect** to **empty**:

```
if(draw_rect == "set")
{
  rect(x,y,70,50);
  document.getElementById("status").innerHTML = "Rectangle is drawn. ";
  draw_rect = "";
}
```

Great! We have added the JS code to our application.

What's NEXT?

In the next class, we will start learning how to manipulate shapes using body parts.