JAVASCRIPT CODE

Volume 1

by

Christopher Topalian

Copyright 2000-2023
All Rights Reserved

Dedicated
to
God the Father

Where to Place Our Code

We Type or Paste our Code in the Console of the Web Browser

1. Open any Web Browser
2. Press F12

This opens our Web Browser CONSOLE

>> We Type the Code that we see on the next page into this Console Code Area.

Type the following Code in the Web Browser Console

We Type this Code in the Web Browser Console

We then Press Enter button

A Message Box will appear, as shown on the next page.

This is the Result of Our Code

This is an alert message box.

As we can see above, we typed our code and hit the enter button and it triggered an alert box to display the message Hi.

We can display any message that we want in the alert box.
We can also display variables, as shown in the next tutorial.

```
javascript:(
/* Code in a Script Editor, Paste Code in Web
Browser Console F12 */
function()
{
    function askName()
    {
        let name = prompt("Enter Name");

        return name;
    }

    alert("Hi " + askName());

}());

/*
Select All of the Code and
press Ctrl + C to Copy it.
```

Open Web Browser Console with F12

Paste the Code in the Browser Console using Ctrl + V to Paste.
Press the Enter Button.

A Message Box will appear that says Hi and the person's name that was entered.
*/

This is the Result of Our Code

```
name = prompt("Enter Name");
```

The function named prompt is assigned to a variable that we call name.

The name variable is given a value when the person types their name in the text area of the prompt box.

When the person hits the OK button, it will greet them by name using an alert message box.

Web Browser Console or Bookmarks?

Bookmarklets VS Web Browser Console to Activate our Code:

Activating Code in the Web Browser Console is very useful for quickly and easily testing code by Copying and Pasting from a Text File.

But, Activating Code in a Bookmarklet enables us to trigger the code with the click of the left mouse button, instead of having to copy and paste.

Either way is fine.
Either way will activate our code.

Some people may find it easier to keep their code in text files and then paste their code in the Web Browser Console when needed, but other people might more enjoy activating their code using Bookmarks.

Let's learn how to trigger our code in the URL Field of a Bookmark. This is called a BOOKMARKLET, and we make it by placing our JavaScript code in the location field of a Bookmark that we create.

The next pages show step by step, how to put our JavaScript code in a Bookmark, but remember that we can keep using the Web Browser Console method too.

We usually make bookmarklets of scripts that we use very often.

But for just testing things it is often much easier to just paste our code from our script editor into the Browser Console and then hit the enter button to trigger the code.

How to Make Bookmarklets

We Start by Making a Bookmark, which will contain our JavaScript Code in the Location Field.

Left Click the Bookmarks Menu

Choose Bookmark This Page or

Left Click the Star icon or

Shortcut is Control + D

We should now have a Bookmark created.

Edit Our Bookmark to Add Our JS

We open the Properties of our Bookmark to Edit it. Once open, we will see the Location Field, where we will place our JavaScript Code.

We Type Our JS Code in the URL Text Box of Our Bookmark

We Type this Code in the URL text box

javascript: alert("Hi");

We then Press Save button

Activating Our Bookmarklet

We use our Bookmarklet by Left Clicking on our Bookmark on the Browser Toolbar.
This will activate an ALERT dialog box, with the text message, Hi

Why do we make a Bookmark?

Our JavaScript code will be pasted into the URL field of a Bookmark. The bookmark contains our JavaScript code and makes triggering the code easy. Whenever we click on the bookmark, it will trigger our JavaScript code!
But, we don't do the typing of our code in the URL field of the bookmark, because it would be tough to edit on a one line URL field element.
So, instead, we do our coding work in a modern text editor, such as VSCode, and then we just copy our code from the script editor and paste it into the URL field of the Bookmark we create.
When we want to edit our Bookmarklet, we change the code in our Script editor and then paste it again into the URL field of our Bookmark and save it again. Since we have many thousands of scripts, that we want to keep organized, we keep our scripts in project folders, instead of relying on bookmarks in the browser. We keep our code organized in folders.

Show Bookmarks Panel - Control + B

We can show our Bookmarks folders on a side panel. We can organize many thousands of scripts easily into folders and use them when we want.

```
javascript:(
/* How to Add Comments */
function()
{
    function askName()
    {
        let name = prompt("Enter Name");

        /* Here is another comment. Comments can
be on multiple lines using this way */

        return name;
    }

    alert("Hi " + askName());
}());

/* Any code in between will be commented out, but
the code before and after will NOT be effected. */
```

```
javascript:(
/* prompt, if, toLowerCase, alert, == Equal To */
function()
{
    function askName()
    {
        let name = prompt("Enter
Name").toLowerCase();

        if (name == "john")
        {
            alert("Hi " + name);
        }
    }

    askName();
}());
/* John or JOHN or jOhN will activate */
/* Jo hn or JOH N or J O H N will NOT activate */
```

Christopher Topalian

/* if diagram */

Christopher Topalian

/* if diagram with a test condition */

```
javascript:(
/* prompt, if else, toLowerCase, == Equal To */
function()
{
    function askName()
    {
        let name = prompt("Enter
Name").toLowerCase();
        if (name == "john")
        {
            return "Hi John";
        }
        else
        {
            return "Hi " + name + " Where is John";
        }
    }
    alert(askName());
}());
```

```
javascript:(
/* prompt, if else, toLowerCase != Not Equal To */
function()
{
    function askName()
    {
        let name = prompt("Enter
Name").toLowerCase();
        if (name != "john")
        {
            return "Hi " + name + " Where is John";
        }
        else
        {
            return "Hi John";
        }
    }
    alert(askName());
}());
```

```
javascript:(
/* prompt OK and Cancel Button Responses */
function()
{
    function askName()
    {
        let name = prompt("Enter Name");

        if (name == null)
        {
            return "You pressed the Cancel button";
        }
        if (name == "")
        {
            return "You pressed OK without first
entering your name";
        }

        return name;
    }
```

```
    alert("Hi " + askName());

}());

/*
if the person enters their name and presses OK,
it says their name.

if the person presses the OK button without first
entering their name, it tells them that they didn't
enter their name

if the person presses the Cancel button it tells
them that they pressed the Cancel button.
*/
```

```
javascript:(
/* prompt if else if, < less than, > greater than */
function()
{
    function askQuestion()
    {
        let question = prompt("What is 5 x 5?");

        if (question < 25)
        {
            return "A bit higher";
        }
        else if (question > 25)
        {
            return "A bit lower";
        }
        else
        {
            return "Correct";
```

```
        }
    }

    alert(askQuestion());

}());
```

```
javascript:(
/* prompt, if else if, do while loop, <, >, != */
function()
{
    function askQuestion()
    {
        let message;

        do
        {
            let question = prompt("What is 5 x 5?");

            if (question < 25)
            {
                message = "A bit higher";
            }
            else if (question > 25)
            {
                message = "A bit lower";
```

```
        }
        else
        {
            message = "Correct";
        }
        alert(message);
    }
    while (question != 25);
}

askQuestion();

}());
```

Christopher Topalian

/* do while loop diagram */

```
javascript:(
/* prompt, if, while loop, < less than */
function()
{
    function askQuestion()
    {
        let players = prompt("Enter Number of
Players");

        let x = 0;

        while (x < players)
        {
            x++;
            console.log("Player " + x + ", ");
        }
    }
    askQuestion();
}());
```

Christopher Topalian

/* while loop diagram */

```
javascript:(
/* prompt, if, while loop, <= less than */
function()
{
    function askQuestion()
    {
        let players = prompt("Enter Number of
Players");

        let x = 1;

        while (x <= players)
        {
            console.log("Player " + x + ", ");

            x++;
        }
    }
    askQuestion();
}());
```

```
javascript:(
/* prompt, for loop, createElement */
function()
{
    function askQuestion()
    {
        let players = prompt("Enter Number of
Players");

        let mainDiv =
document.createElement("div");
        mainDiv.style.position = "absolute";
        mainDiv.style.width = 200 + "px";
        mainDiv.style.height = 200 + "px";
        mainDiv.style.display = "flex";
        mainDiv.style.flexDirection = "column";
        mainDiv.style.overflowY = "scroll";
        document.body.append(mainDiv);
```

```
        for (let x = 0; x < players; x++)
        {
            let player =
document.createElement("button");

            player.onclick = function()
            {
                alert("Player " + x );
            };

            player.innerHTML = "Player " + x;

            mainDiv.append(player);
        }
    }

    askQuestion();

}());
```

Christopher Topalian

/* for loop diagram */

```
javascript:(
/* prompt, for loop with break */
/* we use a for loop to display how many players
the user chose and use break if it is not enough
players chosen */
function()
{
    function askQuestion()
    {
        let players = prompt("Enter Number of
Players");

        for (let x = 1; x <= players; x++)
        {
            if (players <= 3)
            {
                alert(players + " people is not
enough");
```

```
            break;
        }

        console.log("Player " + x + ", ");
    }
}

askQuestion();

}());
```

```javascript:(
/* Date, Time, Time zone */
function()
{
    function getDateAndTime()
    {
        let currentDate = new Date();

        return currentDate;
    }

    console.log(getDateAndTime());

    alert(getDateAndTime());

}());

/* Sun Aug 27 2023 07:41:22 GMT-0400 (Eastern
Daylight Time) */
```

```
javascript:(
/* Date and Time - toLocaleString */
function()
{
    function getDateAndTime()
    {
        let currentDate = new Date();

        let dateString =
currentDate.toLocaleString();

        return dateString;
    }

    console.log(getDateAndTime());

    alert(getDateAndTime());

}());
/* 8/27/2023, 7:35:23 AM */
```

```javascript
javascript:(
/* Date - Year */
function()
{
    function getTheYear()
    {
        let currentDate = new Date();

        let year = currentDate.getUTCFullYear();

        return year;
    }

    console.log("Year: " + getTheYear());

    alert("Year: " + getTheYear());

}());

/* Year: 2023 */
```

```
javascript:(
/* Date - Month */
function()
{
    function getTheMonth()
    {
        let currentDate = new Date();

        let month = currentDate.getMonth();

        if (month == 0)
        {
            return "January";
        }
        else if (month == 1)
        {
            return "February";
        }
        else if (month == 2)
        {
```

```
        return "March";
    }
    else if (month == 3)
    {
        return "April";
    }
    else if (month == 4)
    {
        return "May";
    }
    else if (month == 5)
    {
        return "June";
    }
    else if (month == 6)
    {
        return "July";
    }
    else if (month == 7)
    {
```

```
        return "August";
    }
    else if (month == 8)
    {
        return "September";
    }
    else if (month == 9)
    {
        return "October";
    }
    else if (month == 10)
    {
        return "November";
    }
    else if (month == 11)
    {
        return "December";
    }

    return month;
```

```
    }

    console.log("Month is: " + getTheMonth());

    alert("Month is: " + getTheMonth());

}());

/*
August
*/
```

```
javascript:(
/* Date - Day */
function()
{
    function getTheDay()
    {
        let currentDate = new Date();

        let day = currentDate.getDay();

        if (day == 0)
        {
            return "Sunday";
        }
        else if (day == 1)
        {
            return "Monday";
        }
        else if (day == 2)
        {
```

```
        return "Tuesday";
    }
    else if (day == 3)
    {
        return "Wednesday";
    }
    else if (day == 4)
    {
        return "Thursday";
    }
    else if (day == 5)
    {
        return "Friday";
    }
    else if (day == 6)
    {
        return "Saturday";
    }

    return day;
```

```
    }

    console.log("Today is: " + getTheDay());

    alert("Today is: " + getTheDay());

}());

/*
Sunday
*/
```

```
javascript:(
/* Date - Time - Military - 24 Hour Format */
function()
{
    function getMilitaryTime()
    {
        let currentDate = new Date();
        let hours = currentDate.getHours();
        let minutes = currentDate.getMinutes();
        let seconds = currentDate.getSeconds();

        if (minutes < 10)
        {
            minutes = "0" + minutes;
        }

        if (seconds < 10)
        {
            seconds = "0" + seconds;
        }
```

```
        let timeString = hours + ":" + minutes + ":" +
seconds;

        return timeString;
    }

    console.log("Time: " + getMilitaryTime());

    alert("Time: " + getMilitaryTime());

}());

/*
Time: 17:17:28
*/
```

```
javascript:(
/* Date - Time - AM or PM - 12 Hour Format */
function()
{
    function getTheTime()
    {
        let currentDate = new Date();
        let hours = currentDate.getHours();
        let minutes = currentDate.getMinutes();
        let seconds = currentDate.getSeconds();
        let amOrPm;

        if (hours >= 12)
        {
            amOrPm = "PM";
            if (hours > 12)
            {
                hours -= 12;
            }
        }
```

```
else
{
    amOrPm = "AM";
    if (hours === 0)
    {
        hours = 12;
    }
}

if (minutes < 10)
{
    minutes = "0" + minutes;
}

if (seconds < 10)
{
    seconds = "0" + seconds;
}
```

```
        let timeString = hours + ":" + minutes + ":" +
seconds + " " + amOrPm;

        return timeString;
    }

    console.log("Time: " + getTheTime());

    alert("Time: " + getTheTime());

}());

/* Time: 8:50:17 AM */
```

```
javascript:(
/* Date - Time - AM or PM - 12 Hour Format –
Updates Time in a div */
function()
{
    function getTheTime()
    {
        let currentDate = new Date();
        let hours = currentDate.getHours();
        let minutes = currentDate.getMinutes();
        let seconds = currentDate.getSeconds();
        let amOrPm;

        if (hours >= 12)
        {
            amOrPm = "PM";
            if (hours > 12)
            {
                hours -= 12;
            }
```

```
}
else
{
    amOrPm = "AM";
    if (hours === 0)
    {
        hours = 12;
    }
}

if (minutes < 10)
{
    minutes = "0" + minutes;
}

if (seconds < 10)
{
    seconds = "0" + seconds;
}
```

```
        let timeString = hours + ":" + minutes + ":" +
seconds + " " + amOrPm;

        return timeString;
    }

    function createTimeText()
    {
        let timeDiv =
document.createElement("div");
        timeDiv.style.position = "fixed";
        timeDiv.style.right = "0px";
        timeDiv.style.top = "0px";
        timeDiv.style.paddingLeft = "15px";
        timeDiv.style.paddingRight = "15px";
        timeDiv.style.paddingTop = "5px";
        timeDiv.style.paddingBottom = "5px";
        timeDiv.style.borderRadius = "8px";
        timeDiv.style.backgroundColor = "rgb(0, 0,
0)";
```

```
        timeDiv.style.fontSize = "30px";
        timeDiv.style.color = "rgb(255, 255, 255)";

        setInterval(function()
        {
            timeDiv.innerHTML = getTheTime();
        }, 1000);

        document.body.append(timeDiv);
    }

    createTimeText();

}());

/* Time: 8:57:17 AM */
```

```
javascript:(
/* URL - Go to a Webpage */
function()
{
    let url001 = "https://www.google.com";

    function openWebpage()
    {
        window.location.href = url001;
    }

    openWebpage();

}());
```

```javascript
javascript:(
/* URL of the webpage */
function()
{
    function urlOfPage()
    {
        let url = window.location.href;

        return url;
    }

    console.log(urlOfPage());

    alert(urlOfPage());

}());
```

```
javascript:(
/* Title of the webpage */
function()
{
    function titleOfPage()
    {
        let title = document.title;

        return title;
    }

    console.log(titleOfPage());

    alert(titleOfPage());

}());
```

```
javascript:(
/* Title and URL of the webpage */
function()
{
    function titleOfPage()
    {
        let title = document.title;
        return title;
    }

    function urlOfPage()
    {
        let url = window.location.href;

        return url;
    }
    console.log(titleOfPage() + "\n" + urlOfPage());

    alert(titleOfPage() + "\n" + urlOfPage());
}());
```

```
javascript:(
/* Title and URL of the webpage as a single
function */
function()
{
    function getPageData()
    {
        let pageTitle = document.title;
        let pageURL = window.location.href;

        let pageData =
        "Title: " + pageTitle + "\n" +
        "URL: " + pageURL;

        return pageData;
    }

    console.log(getPageData());
    alert(getPageData());
}());
```

```
javascript:(
/* Elements - How Many Elements on a Page */
function()
{
    function howManyElements()
    {
        let elements =
document.getElementsByTagName("*");

        let elementCount = elements.length;

        return elementCount;
    }

    console.log(howManyElements());

    alert(howManyElements());

}());
```

```
javascript:(
/* Images - How many Images on a page */
function()
{
    function howManyImages()
    {
        let images =
document.getElementsByTagName('img');

        let imageCount = images.length;

        return imageCount;
    }

    console.log("Number of images is " +
howManyImages());

    alert("Number of images is " +
howManyImages());
}());
```

```
javascript:(
/* Links - How many Images on a page */
function()
{
    function howManyLinks()
    {
        let links =
document.getElementsByTagName('a');

        let linksCount = links.length;

        return linksCount;
    }

    console.log("Number of Links: " +
howManyLinks());

    alert("Number of Links: " + howManyLinks());

}());
```

```
javascript:(
/* How many specified elements */
function()
{
   function
howManySpecifiedElements(whichElementType)
   {
      let elements =
document.getElementsByTagName(whichEleme
ntType);

      let elementCount = elements.length;

      return elementCount;
   }

  console.log(howManySpecifiedElements("a"));
   alert(howManySpecifiedElements("a"));

}());
```

```
javascript:(
/* Style Links */
function()
{
    function styleLinks()
    {
        let links =
document.getElementsByTagName('a');

        for (let x = 0; x < links.length; x++)
        {
            links[x].style.backgroundColor = "rgb(0,
255, 255)";

            links[x].style.color = "rgb(0, 0, 0)";
        }
    }

    styleLinks();
}());
```

```
javascript:(
/* Style Specified Elements */
function()
{
    function
styleSpecifiedElements(whichElementType)
    {
        let elements =
document.getElementsByTagName(whichEleme
ntType);

        for (let x = 0; x < elements.length; x++)
        {
            elements[x].style.backgroundColor =
"rgb(0, 255, 255)";
            elements[x].style.color = "rgb(0, 0, 0)";
        }
    }
    styleSpecifiedElements("a");
}());
```

```
javascript:(
/* Text Color of a Webpage Changed to Aqua */
function()
{
    function textColorChange()
    {
        let style = document.createElement("style");

        style.textContent = "body * { color: aqua !
important; }";

        document.head.append(style);
    }

    textColorChange();

}());

/* !important makes a CSS rule a higher
specificity. It overrides any other style */
```

```
javascript:(
/* Text Color of a Webpage Changed to a
Specified Color */
function()
{
    function textColorChange(whichColor)
    {
        let style = document.createElement("style");

        style.textContent = "body * { color: " +
whichColor + " !important; }";

        document.head.append(style);
    }

    textColorChange("aqua");

}());
```

```
javascript:(
/* Style All Elements of a Specified Type */
function()
{
    function styleElement(whichElementType)
    {
        let elements =
document.getElementsByTagName(whichEleme
ntType);

        for (let x = 0; x < elements.length; x++)
        {
            elements[x].style.borderStyle = "solid";

            elements[x].style.borderWidth = 1 + "px";

            elements[x].style.borderRadius = 8 +
"px";
```

```
            elements[x].style.borderColor = "rgb(0,
255, 255)";

            elements[x].style.fontSize = 20 + "px";

            elements[x].style.fontWeight = "bold";
        }
    }

    styleElement("div");

}());
```

```
javascript:(
/* Number of Elements of a Specified Type on a
Page */
function()
{
    function
getNumberOfSpecifiedElement(whichElement)
    {
        let elements =
document.getElementsByTagName(whichEleme
nt);

        return elements.length;
    }

console.log(getNumberOfSpecifiedElement("div
"));
    alert(getNumberOfSpecifiedElement("div"));
}());
```

```
javascript:(
/* Show the InnerHTML of each specified
element type on a Page */
function()
{
   function
showElementInnerHTML(whichElementType)
   {
      let report = "";

      let elements =
document.getElementsByTagName(whichEleme
ntType);

      for (let x = 0; x < elements.length; x++)
      {
         report += elements[x].innerHTML;

         report += "\n";
```

```
        }

        console.log(report);

        alert(report);
    }

    showElementInnerHTML("p");

}());

/*
Show innerHTML of element types, such as:
/*
<p>: Paragraph elements.
<h1>, <h2>, <h3>, <h4>, <h5>, <h6>: Headings of
different levels.
<div>: Div elements.
<span>: Span elements.
```

<li>: List item elements within <ul> or <ol>.
<a>: Anchor links.
<button>: Buttons.
<label>: Labels for form elements.
<textarea>: Textareas within forms.
<option>: Options within <select> elements.
<blockquote>: Block quotes.
<cite>: Citations within block quotes.
<abbr>: Abbreviations.
<code>: Code snippets.
<pre>: Preformatted text.
*/

```
javascript:(
/* Show the src of each specified element type
on a Page */
function()
{
   function
showElementsSrc(whichElementType)
   {
      let report = "";

      let elements =
document.getElementsByTagName(whichEleme
ntType);

      for (let x = 0; x < elements.length; x++)
      {
         report += elements[x].src;

         report += "\n";
      }
```

```
        console.log(report);

        alert(report);
    }

    showElementsSrc("img");

}());

/*
Show src of element types, such as:
<img>: Image elements.
<input>: Input elements.
<script>: Script elements.
*/
```

```
javascript:(
/* Show the href of each specified element type
on a Page */
function()
{
    function
showElementsHref(whichElementType)
    {
        let report = "";

        let elements =
document.getElementsByTagName(whichEleme
ntType);

        for (let x = 0; x < elements.length; x++)
        {
            report += elements[x].href;

            report += "\n";
        }
```

```
        console.log(report);

        alert(report);
    }

    showElementsHref("link");

}());

/*
Show href of element types, such as:

<a>: Anchor elements.

<link>: Link elements, such as style sheets.
*/
```

```
javascript:(
/* Show the InnerHTML of the specified element
type and style it */
function()
{
   function
showElementInnerHTML(whichElementType)
   {
      let report = "";

      let elements =
document.getElementsByTagName(whichEleme
ntType);

      for (let x = 0; x < elements.length; x++)
      {
         elements[x].style.padding = 10 + "px";

         elements[x].style.borderStyle = "solid";
```

```
        elements[x].style.borderWidth = 2 + "px";

        elements[x].style.borderColor = "rgb(0,
255, 255)";

        elements[x].style.fontWeight = "bold";

        report += elements[x].innerHTML;

        report += "\n";
    }

    console.log(report);
    alert(report);
  }

  showElementInnerHTML("p");
}());
```

```
javascript:(
/* Show All Image URLS */
function()
{
    function showImageUrls()
    {
        let report = "";

        let theLinks =
document.getElementsByTagName("img");

        for (let x = 0; x < theLinks.length; x++)
        {
            report += theLinks[x].src;

            report += "\n";
        }

        console.log(report);
```

```
        alert(report);
    }

    showImageUrls();

}());


/*
alert shows a limited amount of characters.

Use console.log or other method to show all
image links found, when there are many.

Press F12 to see the console report.
*/
```

```
javascript:(
/* Show All Image URLS with style */
function()
{
    function showImageUrls()
    {
        let report = "";

        let images =
document.getElementsByTagName("img");

        for (let x = 0; x < images.length; x++)
        {
            images[x].style.borderStyle = "solid";

            images[x].style.borderWidth = 2 + "px";

            images[x].style.borderColor = "rgb(0, 255,
255)";
```

```
        report += images[x].src;

        report += "\n";
    }

    console.log(report);

    alert(report);
}

showImageUrls();

}());
```

```
javascript:(
/* Show All Buttons with style */
function()
{
    function showButons()
    {
        let report = "";

        let buttons =
document.getElementsByTagName("button");

        for (let x = 0; x < buttons.length; x++)
        {
            buttons[x].style.borderStyle = "solid";

            buttons[x].style.borderWidth = 2 + "px";

            buttons[x].style.borderColor = "rgb(0,
255, 255)";
```

```
            report += buttons[x].innerHTML;

            report += "\n";
        }

        console.log(report);

        alert(report);
    }

    showButons();

}());
```

```
javascript:(
/* Show All Links with style */
function()
{
    function showLinks()
    {
        let report = "";

        let links =
document.getElementsByTagName("a");

        for (let x = 0; x < links.length; x++)
        {
            links[x].style.borderStyle = "solid";

            links[x].style.borderWidth = 2 + "px";

            links[x].style.borderColor = "rgb(0, 255,
255)";
```

```
        report += links[x].href;

        report += "\n";
    }

    console.log(report);

    alert(report);
}

showLinks();

}());
```

```
javascript:(
/* Get Selected Text */
function()
{
    function getSelectedText()
    {
        let selectedText =
window.getSelection().toString();

        return selectedText;
    }

    console.log(getSelectedText());

    alert(getSelectedText());

}());
```

```
javascript:(
/* Replace Words on a Webpage */
function()
{
    function replaceWords()
    {
        let elements =
document.getElementsByTagName('*');

        for (let x = 0; x < elements.length; x++)
        {
            let theElement = elements[x];

            theElement.innerHTML =
theElement.innerHTML.replace(/\b(?:News)\b/gi,
'Howdy');

            theElement.innerHTML =
theElement.innerHTML.replace(/\b(?:The|Was|A|
To|Were)\b/gi, 'LOL');
```

```
            theElement.innerHTML =
theElement.innerHTML.replace(/\b(?:What|
Where|When|Why|How)\b/gi, 'Funny');
        }
    }

    replaceWords();

}());

/*

/ beginning and ending slashes show the start
and end of the regular expression pattern.

g means global and it replaces all occurrences
of the pattern, not only the first one.
```

i means case insensitive, meaning it matches uppercase and lowercase of the found pattern.

\b(?:News)\b The regular expression pattern being searched for

\b word boundary anchor means that the pattern is matched only as a whole word but not as part of a bigger word.

(?:News) This is a non-capturing group that matches the word "News." The ?: at the beginning of the group makes it non-capturing, which means it won't be found as part of a match.

\b word boundary anchor ensures the whole word is matched.
*/

```
javascript:(
/* Replace Words on a Webpage - Variation */
function()
{
    function replaceText(whichElement)
    {
        whichElement.innerHTML =
whichElement.innerHTML.replace(/\bgoogle\b/gi,
'Orc Factory');
    }

    let elements =
document.getElementsByTagName('*');

    for (let x = 0; x < elements.length; x++)
    {
        replaceText(elements[x]);
    }

}());
```

```
javascript:(
/* Images Display to None */
function()
{
    function hideImages()
    {
        let images =
document.getElementsByTagName('img');

        for (let x = 0; x < images.length; x++)
        {
            if
(window.getComputedStyle(images[x]).display
=== "none")
            {
                images[x].style.display = "block";
            }
            else
            {
                images[x].style.display = "none";
```

```
        }
    }
}

hideImages();

}());
```

```
javascript:(
/* Images - Gray Scale */
function()
{
    function makeImagesGrayScale()
    {
        let theImages =
document.getElementsByTagName('img');

        for (let x = 0; x < theImages.length; x++)
        {
            theImages[x].style.filter =
'grayscale(100%)';
        }
    }

    makeImagesGrayScale();

}());
```

```
javascript:(
/* Toggle Images On/Off */
function()
{
    function toggleImagesOnOrOff()
    {
        let images =
document.getElementsByTagName('img');

        for (let x = 0; x < images.length; x++)
        {
            let display =
window.getComputedStyle(images[x]).display;

            if (display === 'none')
            {
                images[x].style.display = 'block';
            }
            else
            {
```

```
            images[x].style.display = 'none';
        }
    }
}

toggleImagesOnOrOff();

}());

/*
if images are hidden, it will show them

if images are visible, it will hide them
*/
```

```
javascript:(
/* Mouse Click Changes Font Weight */
function()
{
    function mouseTriggersStyle()
    {
        document.onclick = function()
        {
            let elements =
document.getElementsByTagName('*');

            for (let x = 0; x < elements.length; x++)
            {
                elements[x].style.fontWeight = "bold";
            }
        };
    }

    mouseTriggersStyle();
}());
```

```
javascript:(
/* First Click Does X, Second Click Does Y */
function()
{
    function mouseTriggersStyle()
    {
        let toggleFlag = false;

        document.onclick = function()
        {
            let elements =
document.getElementsByTagName('*');

            for (let x = 0; x < elements.length; x++)
            {
                if (toggleFlag)
                {
                    elements[x].style.fontWeight =
"normal";
                }
```

```
            else
            {
                elements[x].style.fontWeight =
"bold";
            }
        }

        toggleFlag = !toggleFlag;
    };
  }

  mouseTriggersStyle();

}());

/*
The next tutorial shows a variation of this idea.
Choose the way that you find to be more natural.
*/
```

```
javascript:(
/* First Click Does X, Second Click Does Y -
Variation */
function()
{
    let toggleFlag = false;

    function mouseTriggersStyle()
    {
        document.onclick = function()
        {
            let elements =
document.getElementsByTagName('*');

            for (let x = 0; x < elements.length; x++)
            {
                if (toggleFlag == false)
                {
                    elements[x].style.fontWeight =
"normal";
```

```
                toggleFlag = true;
            }
            else
            {
                elements[x].style.fontWeight =
"bold";

                toggleFlag = false;
            }
        }
    };
}

mouseTriggersStyle();

}());

/* I consider this way to be more natural */
```

```
javascript:(
/* First Click do X, Second Click do Y, Third Click
do Z */
function()
{
    let state = 0;

    function toggleFontWeightAndColor(element)
    {
        if (state === 0)
        {
            element.style.fontWeight = "normal";

            element.style.color = "red";

            state = 1;
        }

        else if (state === 1)
        {
```

```
        element.style.fontWeight = "bold";

        element.style.color = "green";

        state = 2;
    }

    else
    {
        element.style.color = "blue";

        state = 0;
    }
}

function mouseTriggersStyle()
{
    document.onclick = function()
    {
```

```
        let elements =
document.getElementsByTagName('*');

        for (let x = 0; x < elements.length; x++)
        {
         toggleFontWeightAndColor(elements[x]);
        }
      };
   }

   mouseTriggersStyle();

}());
```

```
javascript:(
/* Random Number Generator from 1 to 100 */
function()
{
    function generateRandomNumber()
    {
        let randomNumber =
Math.floor(Math.random() * 100) + 1;

        return randomNumber;
    }

    console.log(generateRandomNumber());

    alert(generateRandomNumber());
}());

/* In this script, console.log will show one
random value and alert will show another
random value */
```

```
javascript:(
/* Random Background Color */
function()
{
    function randomBackgroundColor()
    {
        let randomColor = 'rgb(' +
Math.floor(Math.random() * 255) + ',' +
Math.floor(Math.random() * 255) + ',' +
Math.floor(Math.random() * 255) + ')';

        document.body.style.backgroundColor =
randomColor;
    }

    randomBackgroundColor();

}());
```

```
javascript:(
/* Random Background Color - Variation */
function()
{
    function randomValue()
    {
        return Math.floor(Math.random() * 255);
    }

    let red = randomValue();
    let green = randomValue();
    let blue = randomValue();

    let randomColor = 'rgb('+red+', '+green+', '+
blue+')';

    document.body.style.backgroundColor =
randomColor;

}());
```

```
javascript:(
/* Random Greeting */
function()
{
    let greetings =
    [
        "Hi",
        "Howdy.",
        "How ya doing?",
        "You having fun?",
        "Glad you are here.",
        "Nice weather today.",
        "It's such a nice day out."
    ];

    function makeRandomMessage(whichArray)
    {
        let randomGreeting =
whichArray[Math.floor(Math.random() *
whichArray.length)];
```

```
        return randomGreeting;
    }

    let message =
makeRandomMessage(greetings);

    console.log(message);

    alert(message);

}());
```

```
javascript:(
/* Random Quote Generator */
function()
{
    function generateRandomQuote()
    {
        let quotes =
        [
            "Always be kind",
            "Be excellent to each other",
            "Live long and prosper"
        ];

        let randomIndex = Math.floor(Math.random()
* quotes.length);

        let randomQuote = quotes[randomIndex];

        return randomQuote;
    }
```

```
    let quote = generateRandomQuote();

    console.log(quote);

    alert(quote);

}());
```

```
javascript:(
/* Random Font for the entire Webpage */
function()
{
    let fonts =
    [
        'Arial',
        'Times New Roman',
        'Courier New',
        'Garamond',
        'Avant Garde',
        'Palatino Linotype'
    ];

    function makeRandomFont()
    {
        let elements =
document.getElementsByTagName('*');
```

```
        let randomFont =
fonts[Math.floor(Math.random() * fonts.length)];

        for (let x = 0; x < elements.length; x++)
        {
            elements[x].style.fontFamily =
randomFont;
        }

        console.log('Changed font to ' +
randomFont);

        alert('Changed font to ' + randomFont);
    }

    makeRandomFont();

}());
```

```
javascript:(
/* Random Background Color */
function()
{
    function getRandomColor()
    {
        let r = Math.floor(Math.random() * 255);
        let g = Math.floor(Math.random() * 255);
        let b = Math.floor(Math.random() * 255);

        return 'rgb('+r+', '+g+ ', '+b+')';
    }

    document.body.style.backgroundColor =
getRandomColor();

}());
```

```
javascript:(
/* Random BG to Specified Element */
function()
{
    function
styleSpecifiedElements(whichElementType)
    {
        let elements =
document.getElementsByTagName(whichEleme
ntType);

        for (let x = 0; x < elements.length; x++)
        {
            let randomColor = 'rgb(' +
Math.floor(Math.random() * 255) + ',' +
Math.floor(Math.random() * 255) + ',' +
Math.floor(Math.random() * 255) + ')';

            elements[x].style.backgroundColor =
randomColor;
```

```
        }
    }

    styleSpecifiedElements("a");

}());
```

```
javascript:(
/* Random BG to Specified Elements – Variation
*/
function()
{
   function
styleSpecifiedElements(whichElementType,
whichRgbColor)
   {
      let theElements =
document.getElementsByTagName(whichEleme
ntType);

      for (let x = 0; x < theElements.length; x++)
      {
         theElements[x].style.backgroundColor =
whichRgbColor;
      }
   }
```

```
    styleSpecifiedElements("p", 'rgb(' +
Math.floor(Math.random() * 255) + ',' +
Math.floor(Math.random() * 255) + ',' +
Math.floor(Math.random() * 255) + ')');

}());
```

```
javascript:(
/* Random Positions for a Circle */
function()
{
    function moveObjectRandomPositions()
    {
        let obj = document.createElement("div");
        obj.style.position = "absolute";
        obj.style.width = "20px";
        obj.style.height = "20px";
        obj.style.background = "aqua";
        obj.style.borderRadius = "50%";
        document.body.append(obj);

        function getRandomPosition()
        {
            let x = Math.floor(Math.random() *
window.innerWidth);
```

```
        let y = Math.floor(Math.random() *
window.innerHeight);
        return {
            x: x,
            y: y
        };
    }

    function moveObject()
    {
        let pos = getRandomPosition();
        obj.style.left = pos.x + "px";
        obj.style.top = pos.y + "px";
    }

    /* move every 1 second */
    setInterval(moveObject, 1000);
  }
  moveObjectRandomPositions();
}());
```

```
javascript:(
/* Random Circles Pos and Color */
function()
{
    function randomCirclePosAndColor()
    {
        let circle = document.createElement("div");
        circle.style.width = "20px";
        circle.style.height = "20px";
        circle.style.borderRadius = "50%";
        circle.style.position = "absolute";
        circle.style.backgroundColor =
getRandomColor();
        document.body.append(circle);

        /*----*/

        function getRandomPosition()
        {
```

```
        let x = Math.floor(Math.random() *
window.innerWidth);
        let y = Math.floor(Math.random() *
window.innerHeight);

        return {
            x: x,
            y: y
        };
    }

    function getRandomColor()
    {
        let r = Math.floor(Math.random() * 255);
        let g = Math.floor(Math.random() * 255);
        let b = Math.floor(Math.random() * 255);

        return "rgb("+r+", "+g+", "+b+")";
    }
```

```
        function moveCircle()
        {
            let pos = getRandomPosition();

            circle.style.left = pos.x + "px";

            circle.style.top = pos.y + "px";

            /* move every 2 seconds */
            setTimeout(moveCircle, 2000);
        }

        moveCircle();
    }

    randomCirclePosAndColor();

}());
```

```
javascript:(
/* Random Circles Pos and Color - How many to
Create */
function()
{
    function randomCirclePosAndColor()
    {
        let circleCount = prompt("Number of Circles
to Create?");

        circleCount = parseInt(circleCount);

        if (isNaN(circleCount) || circleCount <= 0)
        {
            alert("Enter a valid number more than 0");

            return;
        }

        function getRandomPosition()
```

```
    {
        let x = Math.floor(Math.random() *
window.innerWidth);

        let y = Math.floor(Math.random() *
window.innerHeight);

        return {
            x: x,
            y: y
        };
    }

    function getRandomColor()
    {
        let r = Math.floor(Math.random() * 255);
        let g = Math.floor(Math.random() * 255);
        let b = Math.floor(Math.random() * 255);

        return "rgb("+r+", "+g+", "+b+")";
```

```
}

function moveCircle(circle)
{
    let pos = getRandomPosition();

    circle.style.left = pos.x + "px";

    circle.style.top = pos.y + "px";

    /* move every 2 seconds */
    setTimeout(function()
    {
        moveCircle(circle);
    }, 2000);
}

for (let x = 0; x < circleCount; x++)
{
```

```
        let circle =
document.createElement("div");
        circle.style.width = "20px";
        circle.style.height = "20px";
        circle.style.borderRadius = "50%";
        circle.style.position = "absolute";
        circle.style.backgroundColor =
getRandomColor();
        document.body.append(circle);

        moveCircle(circle);
    }
  }

  randomCirclePosAndColor();

}());
```

```
javascript:(
/* Random Pos, Size, Color Objects on Screen */
function()
{
    function generateRandomObjects()
    {
        /* array to hold the random objects */
        let objects = [];

        /* function to create a random object */
        function createRandomObject()
        {
            let object =
document.createElement('div');
            object.style.position = 'absolute';
            object.style.left = Math.random() *
window.innerWidth + 'px';
            object.style.top = Math.random() *
window.innerHeight + 'px';
```

```
        object.style.width = Math.random() * 50 +
'px';
        object.style.height = Math.random() * 50 +
'px';
        object.style.backgroundColor =
getRandomColor();
        document.body.append(object);

        /* add the object to the array */
        objects.push(object);
    }

    /* function to generate a random color */
    function getRandomColor()
    {
        let r = Math.floor(Math.random() * 255);
        let g = Math.floor(Math.random() * 255);
        let b = Math.floor(Math.random() * 255);

        return 'rgb('+r+', '+g+', '+b+')';
```

```
}

// function to move the objects randomly
function moveObjects()
{
    for (let x = 0; x < objects.length; x++)
    {
        const object = objects[x];

        const newX = Math.random() *
window.innerWidth;

        const newY = Math.random() *
window.innerHeight;

        object.style.left = newX + 'px';

        object.style.top = newY + 'px';
    }
}
```

```
        /* create random objects */
        setInterval(createRandomObject, 1000);

        /* move random objects */
        setInterval(moveObjects, 2000);
    }

    generateRandomObjects();

}());
```

```
/*
More objects will continue to appear on the
screen with random position, size and color.
*/
```

```
javascript:(
/* Random Border Color with Timer */
function()
{
    let theTimer;
    let isRunning = false;

    function getRandomColor()
    {
        let r = Math.floor(Math.random() * 255);
        let g = Math.floor(Math.random() * 255);
        let b = Math.floor(Math.random() * 255);
        return 'rgb('+r+', '+g+', '+b+')';
    }

    function toggleColorChange()
    {
        if (!isRunning)
        {
```

```
        document.body.style.borderStyle =
"solid";

        document.body.style.borderColor =
"rgb(255, 255, 255)";

        theTimer = setInterval(function()
        {
            document.body.style.borderColor =
getRandomColor();
        }, 1000);
    }
    else
    {
        clearInterval(theTimer);

        document.body.style.backgroundColor =
";
    }
    isRunning = !isRunning;
```

```
    }

    toggleColorChange();

    document.body.addEventListener('click',
toggleColorChange);

}());
```

```
javascript:(
/* Replace Images with a Random Image from
our Array */
function()
{
    const textures =
    [
'https://collegeofscripting.weebly.com/uploads/
6/4/4/8/64482293/wood1.jpg',

'https://collegeofscripting.weebly.com/uploads/
6/4/4/8/64482293/copper.jpg',

'https://collegeofscripting.weebly.com/uploads/
6/4/4/8/64482293/silver.jpg',

'https://collegeofscripting.weebly.com/uploads/
6/4/4/8/64482293/gold.jpg',
    ];
```

```
    function replaceImages()
    {
        const images =
document.getElementsByTagName('img');

        for (let x = 0; x < images.length; x++)
        {
            let imageIndex = x % textures.length;

            images[x].src = textures[imageIndex];

            images[x].alt = 'Replaced Image';
        }
    }

    replaceImages();

}());
/* replaces all images on the current page with
random images from our array */
```

```
javascript:(
/* Replace Images with specified Image */
function()
{
    function replaceImages()
    {
        let images =
document.getElementsByTagName('img');

        for (let x = 0; x < images.length; x++)
        {
            images[x].src =
'https://collegeofscripting.weebly.com/uploads/6/
4/4/8/64482293/asphalt.jpg';

            images[x].alt = 'Asphalt';
        }
    }
    replaceImages();
}());
```

```
javascript:(
/* Video Pause - Pauses the Video */
function()
{
    function videoPause()
    {
        let theVideo =
document.querySelector('video');

        theVideo.pause();
    }

    videoPause();

}());

/* When Activated, this Bookmarklet will PAUSE
the first video found on the page.
*/
```

```
javascript:(
/* Video Play */
function()
{
    function videoPlay()
    {
        let theVideo =
document.querySelector('video');

        theVideo.play();
    }

    videoPlay();

}());
```

```
javascript:(
/* Video Back 2 Seconds */
function()
{
    function videoSkipBackwards()
    {
        let theTime =
document.querySelector('video').currentTime +=
-2;
    }

    videoSkipBackwards();

}());
```

```
javascript:(
/* Video Forward 2 Seconds */
function()
{
    function videoSkipForward()
    {
        let theTime =
document.querySelector('video').currentTime +=
2;
    }

    videoSkipForward();

}());
```

```
javascript:(
/* Video Forward 2 Seconds, Keep Activating */
function()
{
    function videoSkipForward()
    {
      document.querySelector('video').currentTime
+= 2;
    }

    setInterval(videoSkipForward, 1000);

}());
```

```
javascript:(
/* Video currentTime */
function()
{
    function videoCurrentTime()
    {
        let time =
document.querySelector('video').currentTime;

        return time;
    }

    console.log(videoCurrentTime());

    alert(videoCurrentTime());

}());
```

```
javascript:(
/* Video currentTime with round */
function()
{
    function videoCurrentTime()
    {
        let time =
document.querySelector('video').currentTime;

        let roundedTime = Math.round(time);

        return roundedTime;
    }

    console.log(videoCurrentTime());

    alert(videoCurrentTime());

}());
```

```
javascript:(
/* Video currentTime using toFixed */
function()
{
    function videoCurrentTime()
    {
        let time =
document.querySelector('video').currentTime;

        let formattedTime = time.toFixed(2);

        return formattedTime;
    }

    console.log(videoCurrentTime());

    alert(videoCurrentTime());

}());
```

```
javascript:(
/* Video - Set the currentTime */
function()
{
    function videoSetTime(whichSeconds)
    {
      document.querySelector('video').currentTime
= whichSeconds;
    }

    videoSetTime(60);

}());
```

```
javascript:(
/* Video - Duration - Total Length in Seconds */
function()
{
    function videoGetDuration()
    {
        let theDuration =
document.querySelector('video').duration;

        return theDuration;
    }

    console.log(videoGetDuration());

    alert(videoGetDuration());

}());
```

```
javascript:(
/* Video - loop if true, non loop if false */
function()
{
    function videoLoop(whichLoopSetting)
    {
        document.querySelector('video').loop =
whichLoopSetting;
    }

    videoLoop(true);

}());
```

```
javascript:(
/* Video muted if true, unmuted if false */
function()
{
    function videoMute(whichMuteSetting)
    {
        document.querySelector('video').muted =
whichMuteSetting;
    }

    videoMute(true);

}());
```

```
javascript:(
/* Video URL - window.location.href */
function()
{
    function videoSource()
    {
        let theSrc = window.location.href;

        return theSrc;
    }

    alert(videoSource());

}());
```

```
javascript:(
/* Video Volume - 0.0 Mute, 1.0 Full */
function()
{
    function videoVolume(whichVolume)
    {
        document.querySelector('video').volume =
whichVolume;
    }

    videoVolume(0.5);

}());
```

```
javascript:(
/* Video Volume - Get Volume Level 0.0 to 1.0 */
function()
{
    function videoVolumeGet()
    {
        let theSrc =
document.querySelector('video').volume;

        return theSrc;
    }

    alert(videoVolumeGet());

}());
```

```
javascript:(
/* Video Speed to Custom Value */
/* Set the speed value of the video, such as: 0.0,
0.25, 0.5, 0.75, 1.00, 1.25, 1.5, 1.75, 2.00, 20.0, or
super slow 0.01 */
function()
{
    function videoSpeedSet(whichSpeed)
    {
   document.querySelector('video').playbackRate
= whichSpeed;
    }

    videoSpeedSet(0.5);

}());
```

```
javascript:(
/* Video - Custom Speed */
function()
{
    function videoSpeedSet()
    {
        const speedInput = prompt('Enter a Speed',
'0.50');

        const videoSpeed =
parseFloat(speedInput);

   document.querySelector('video').playbackRate
= videoSpeed;
    }

    videoSpeedSet();

}());
```

```
javascript:(
/* YouTube - Style Description Border color */
function()
{
    function createBorderAroundVideoInfo()
    {
        let theBorder =
document.querySelector(".style-scope ytd-
watch-metadata");

        theBorder.style.borderStyle = 'solid';

        theBorder.style.borderColor = 'aqua';
    }

    createBorderAroundVideoInfo();

}());
```

Topalian Comic Book Creator Coded in JavaScript

ScriptingCollege
668 subscribers

Subscribe

1 Share

44 views 3 months ago

The long awaited JavaScript application that allows you to create your own comic books in your web browser is finally here! You can include video backgrounds, video animations, animated gifs and so much more. In this video, I show you the gallery feature, which lets you choose item ...more

```
javascript:(
/* YouTube - Get URL, Title, Description, Date */
function()
{
    function getInfo()
    {
        let nameList =
document.querySelectorAll(".style-scope ytd-
video-primary-info-renderer");

        let theNames = [];

        for (let x = 0; x < nameList.length; x++)
        {
            theNames += nameList[x].textContent;

            theNames += "\n";
        }
```

```
let theTextArea = document.createElement("textarea");

theTextArea.style.position = "absolute";
theTextArea.style.left = 100 + 'px';
theTextArea.style.top = 200 + 'px';
theTextArea.style.width = 700 + 'px';
theTextArea.style.height = 200 + 'px';
theTextArea.style.zIndex = "1000";
theTextArea.style.border = "solid 2px rgba(0,0,100,1.0)";
theTextArea.style.background = "white";
theTextArea.style.fontFamily = "arial";
theTextArea.style.fontWeight = "normal";
theTextArea.style.fontSize = "medium";
theTextArea.style.color = "black";
theTextArea.style.textAlign = "center";
```

```
        theTextArea.setAttribute("readonly",
"true");

        theTextArea.value = theNames;

        document.body.append(theTextArea);
    }

    getInfo();

}());
```

```
javascript:(
/* Shows Mouse Position when person clicks the
screen */
function()
{
    function mousePos()
    {
        let mouseX = event.pageX;
        let mouseY = event.pageY;

        let mousePos =
        "Mouse" + "\n" +
        "X " + mouseX + "\n" +
        "Y " + mouseY;

        console.log(mousePos);

        document.title = mousePos;
    }
```

```
    window.addEventListener("click", mousePos,
false);

    mousePos();
}());
```

```
javascript:(
/* Create a Video Game Player on Any Webpage!
*/
function()
{
    function createPlayer()
    {
        let player = document.createElement("div");
        player.id = "thePlayer";
        player.style.position = "absolute";
        player.style.left = 0;
        player.style.top = 0;
        player.style.width = "50px";
        player.style.height = "50px";
        player.style.zIndex = "15";
        player.style.fontFamily = "exo";
        player.style.fontSize = "20px";
        player.style.fontWeight = "bold";
        player.style.color = "rgb(255, 255, 255)";
```

```
    player.style.textAlign = "center";
    player.style.background = "rgba(76, 175,
180, 0.5)";
    document.body.append(player);

    /*----*/

    /* keyboard Letter Codes Being Pressed */
    let keyboard = {};
    keyboard.UP = 87;
    keyboard.DOWN = 83;
    keyboard.LEFT = 65;
    keyboard.RIGHT = 68;

    /*----*/

    /* player's start position and id */
    let ourPlayer =
    {
```

```
        x: 100,
        y: 300,
        speedMultiplier: 2,
        playerId:
document.getElementById("thePlayer")
    };

    /*----*/

    /* key press detection */
    document.body.onkeyup =
document.body.onkeydown = function(e)
{
    if (e.keyCode == 80) /* letter p */
    {
        ourPlayer.playerId.style.background =
"rgba(0, 175, 80, 0.5)";
        ourPlayer.speedMultiplier = 4;
    }
```

```
        if (e.keyCode == 79) /* letter o */
        {
            ourPlayer.playerId.style.background =
"rgba(76, 0, 80, 0.5)";
            ourPlayer.speedMultiplier = 3;
        }
        if (e.keyCode == 73) /* letter i */
        {
            ourPlayer.playerId.style.background =
"rgba(76, 175, 180, 0.5)";
            ourPlayer.speedMultiplier = 2;
        }

        /* find out which key was pressed */
        let theKeyCode = e.keyCode || e.which;

        keyboard[theKeyCode] = e.type ==
'keydown';
    };
```

```
/*----*/

/* character movement updating */
let movePlayer = function(theX, theY)
{
    ourPlayer.x += (theX || 0) *
ourPlayer.speedMultiplier;

    ourPlayer.y += (theY || 0) *
ourPlayer.speedMultiplier;

    ourPlayer.playerId.style.left = ourPlayer.x +
'px';

    ourPlayer.playerId.style.top = ourPlayer.y +
'px';
};
```

```
/*----*/

/* player controls */
let sensePlayerMotion = function()
{
    if (keyboard[keyboard.LEFT])
    {
        movePlayer(-1, 0);
    }
    if (keyboard[keyboard.RIGHT])
    {
        movePlayer(1, 0);
    }
    if (keyboard[keyboard.UP])
    {
        movePlayer(0, -1);
    }
    if (keyboard[keyboard.DOWN])
    {
```

```
        movePlayer(0, 1);
    }
};

/*----*/

/* update the Position of the player */
movePlayer();

/*----*/

function scrollIt()
{
document.getElementById("thePlayer").scrollIntoView(
    {
        block: "center",
        inline: "center"
    });
```

```
    }

    let loop001;

    function gameLoop()
    {
        sensePlayerMotion();

        scrollIt();

        loop001 =
requestAnimationFrame(gameLoop);
    }

    gameLoop();
}

createPlayer();
}());
```

```
javascript:(
/* createElement, append, Make a div, style */
function()
{
    function createInfoDiv()
    {
        let ourDiv = document.createElement("div");
        ourDiv.style.position = "absolute";
        ourDiv.style.left = "100px";
        ourDiv.style.top = "100px";
        ourDiv.style.paddingLeft = "10px";
        ourDiv.style.paddingRight = "10px";
        ourDiv.style.paddingTop = "5px";
        ourDiv.style.paddingBottom = "5px";
        ourDiv.style.zIndex = "1000";
        ourDiv.style.borderStyle = "solid";
        ourDiv.style.borderWidth = "1px";
        ourDiv.style.borderRadius = "8px";
```

```
        ourDiv.style.borderColor = "rgb(0, 255,
255)";
        ourDiv.style.background = "rgb(0, 0, 0)";
        ourDiv.style.fontSize = "30px";
        ourDiv.style.fontWeight = "bold";
        ourDiv.style.color = "rgb(0, 255, 255)";
        ourDiv.style.textAlign = "center";
        ourDiv.innerHTML = "Copper";
        ourDiv.innerHTML += "<br>";
        ourDiv.innerHTML += "29";
        document.body.append(ourDiv);
    }
    createInfoDiv();
}());
```

```
javascript:(
/* createElement - id - Right click on the Div and
choose Inspect */
function()
{
    function createDiv()
    {
        let ourDiv = document.createElement("div");
        ourDiv.id = "ourDiv";
        ourDiv.style.position = "absolute";
        ourDiv.style.left = 100 + "px";
        ourDiv.style.top = 100 + "px";
        ourDiv.style.paddingLeft = "10px";
        ourDiv.style.paddingRight = "10px";
        ourDiv.style.paddingTop = "5px";
        ourDiv.style.paddingBottom = "5px";
        ourDiv.style.background = "rgb(0, 0, 0)";
        ourDiv.style.borderStyle = "solid";
        ourDiv.style.borderWidth = "1px";
```

```
        ourDiv.style.borderRadius = "8px";
        ourDiv.style.borderColor = "rgb(0, 255,
255)";
        ourDiv.style.background = "rgb(0, 0, 0)";
        ourDiv.style.fontSize = "30px";
        ourDiv.style.fontWeight = "bold";
        ourDiv.style.color = "rgb(0, 255, 255)";
        ourDiv.style.textAlign = "center";
        ourDiv.innerHTML = "Right click this div.
Choose Inspect.";
        document.body.append(ourDiv);
    }

    createDiv();

}());
```

Right click this div. Choose Inspect.

```
javascript:(
/* createElement, append, Make a Paragraph */
function()
{
    function createParagraph()
    {
        let ourParagraph =
document.createElement("p");
        ourParagraph.style.position = "absolute";
        ourParagraph.style.top = 100 + 'px';
        ourParagraph.style.left = 100 + 'px';
        ourParagraph.innerHTML = "Here is the
text.";
        document.body.append(ourParagraph);
    }

    createParagraph();

}());
```

```
javascript:(
/* createElement, append, Make a div, style, url
new tab */
function()
{
    function createInfoDiv()
    {
        let copper =
        {
            name: "copper",
            number: "29",
            link:
"https://en.wikipedia.org/wiki/Copper"
        };

        let ourDiv = document.createElement("div");
        ourDiv.style.position = "absolute";
        ourDiv.style.left = "100px";
        ourDiv.style.top = "100px";
        ourDiv.style.paddingLeft = "10px";
```

```
        ourDiv.style.paddingRight = "10px";
        ourDiv.style.paddingBottom = "15px";
        ourDiv.style.paddingTop = "9px";
        ourDiv.style.zIndex = "1000";
        ourDiv.style.borderRadius = "8px";
        ourDiv.style.background = "rgb(0, 0, 0)";
        ourDiv.style.fontSize = 30 + "px";
        ourDiv.style.color = "rgb(0, 255, 255)";
        ourDiv.style.textAlign = "center";
        ourDiv.innerHTML = copper.name;
        ourDiv.innerHTML += "<br>";
        ourDiv.innerHTML += copper.number;
        ourDiv.innerHTML += "<br>";
        ourDiv.innerHTML += '<a href = \" +
copper.link + '\' target = "_blank"> Link </a>';
        document.body.append(ourDiv);
    }

    createInfoDiv();
```

}());

copper
29
Link

```
javascript:(
/* createElement, Array Of Objects - link */
function()
{
    function createInfoDiv()
    {
        let myRecord =
        [
            {
                name: "Key",
                weight: 140,
                link: "https://google.com/news"
            },
            {
                name: "Donald",
                weight: 160,
                link: "https://google.com"
            }
        ];
```

```
let ourDiv = document.createElement("div");
ourDiv.style.position = "absolute";
ourDiv.style.left = "100px";
ourDiv.style.top = "100px";
ourDiv.style.width = "100px";
ourDiv.style.paddingLeft = "10px";
ourDiv.style.paddingRight = "10px";
ourDiv.style.paddingBottom = "15px";
ourDiv.style.paddingTop = "9px";
ourDiv.style.borderRadius = "8px";
ourDiv.style.background = "rgb(0, 0, 0)";
ourDiv.style.zIndex = "1000";
ourDiv.style.fontSize = 30 + "px";
ourDiv.style.color = "rgb(0, 255, 255)";
ourDiv.style.textAlign = "center";
ourDiv.innerHTML = myRecord[0].name;
ourDiv.innerHTML += "<br>";
ourDiv.innerHTML += myRecord[0].weight;
ourDiv.innerHTML += "<br>";
```

```
        ourDiv.innerHTML += '<a href = \" +
myRecord[0].link + '\' target = "_blank"> Link
</a>';

        document.body.append(ourDiv);
    }

    createInfoDiv();

}());
```

```
/*
We make an Array of Objects that we named
myRecord
```

```
The 0 entry of the myRecord array is Key.

The 1 entry of the myRecord array is Donald.
In the above script, we only display one record,
which is the entry that has the name Key.

To display the first entry name we use
myRecord[0].name
To display the first entry weight we use
myRecord[0].weight

To display the second entry name we use
myRecord[1].name
To display the second entry weight we use
myRecord[1].weight
*/
```

```
javascript:(
/* createElement, Array of Objects - Show All */
function()
{
    function createInfoDiv()
    {
        let myRecord =
        [
            {
                name: "Key",
                weight: 140,
                link: "https://google.com/news"
            },
            {
                name: "Donald",
                weight: 160,
                link: "https://google.com"
            }
        ];
```

```
/*----*/

let ourDiv = document.createElement("div");
ourDiv.style.position = "absolute";
ourDiv.style.left = "100px";
ourDiv.style.top = "100px";
ourDiv.style.paddingLeft = "10px";
ourDiv.style.paddingRight = "10px";
ourDiv.style.paddingBottom = "2px";
ourDiv.style.paddingTop = "2px";
ourDiv.style.borderRadius = "8px";
ourDiv.style.background = "rgb(0, 0, 0)";
ourDiv.style.zIndex = "1000";
ourDiv.style.fontSize = 30 + "px";
ourDiv.style.color = "rgb(0, 255, 255)";
ourDiv.style.textAlign = "right";
document.body.append(ourDiv);

let output = "";
```

```
        for (let x = 0; x < myRecord.length; x++)
        {
            output += "<hr>";
            output += myRecord[x].name;
            output += " ";
            output += myRecord[x].weight;
            output += " ";
            output += '<a href = \" + myRecord[x].link
+ '\' target = "_blank"> Link </a>';
        }

        output += '<hr>';

        ourDiv.innerHTML = output;

    }

    createInfoDiv();

}());
```

```
/*
Creates a New div on the page with style & url
link of an array of objects, displaying all entries

We loop through the entries of the myRecords
array to display all of them at once.
The for loop will only continue for as many
entries that are found. In this case, there are
only two entries.

We display them all of the entries on a div that
we have made using createElement.
This div that we create is very useful technology.
*/
```

```
javascript:(
/* createElement, Make a Button */
function()
{
    function textMessage(whichText)
    {
        alert(whichText);
    }

    function createMessage(whichText)
    {
        let ourButton =
document.createElement("button");
        ourButton.style.position = "absolute";
        ourButton.style.left = "100px";
        ourButton.style.top = "100px";
        ourButton.style.padding = "2px";
        ourButton.style.zIndex = "1000";
        ourButton.style.background = "rgb(0, 0, 0)";
```

```
ourButton.style.fontSize = 22 + "px";
ourButton.style.color = "rgb(255, 255, 255)";
ourButton.innerHTML = "Greeting";

ourButton.onmouseover = function()
{
    ourButton.style.color = "rgb(0, 255, 255)";
};

ourButton.onmouseout = function()
{
    ourButton.style.color = "rgb(255, 255,
255)";
};

ourButton.onclick = function()
{
    textMessage(whichText);
};
```

```
        document.body.append(ourButton);
    }
    createMessage("Hi Everyone");
}());
```

```
javascript:(
/* Scene - Position - Get Current Position of the
square object by left clicking it */
function()
{
    let horizon = document.createElement('div');
    horizon.style.position = 'absolute';
    horizon.style.left = '0px';
    horizon.style.top = '0px';
    horizon.style.width = '100%';
    horizon.style.height = '50%';
    horizon.style.backgroundColor = 'rgb(0, 100,
200)';
    document.body.append(horizon);

    /*----*/

    let grass = document.createElement('div');
    grass.style.position = 'absolute';
    grass.style.left = '0px';
```

```
grass.style.bottom = '0px';
grass.style.width = '100%';
grass.style.height = '50%';
grass.style.backgroundColor = 'rgb(0, 120, 0)';
document.body.append(grass);

/*----*/

let road = document.createElement('div');
road.style.position = 'absolute';
road.style.left = '30%';
road.style.bottom = '0%';
road.style.width = '20%';
road.style.height = '100%';
road.style.clipPath = 'polygon(0 100%, 50%
50%, 100% 100%)';
road.style.backgroundColor = 'rgb(0, 20, 0)';
document.body.append(road);

/*----*/
```

```
let object001 = document.createElement('div');
object001.style.position = 'absolute';
object001.style.left = '400px';
object001.style.top = '500px';
object001.style.width = '20px';
object001.style.height = '20px';
object001.style.backgroundColor = 'tan';
object001.style.cursor = 'pointer';
object001.style.zIndex = '10000';

object001.onclick = function(event)
{
    /* get position */
    let objectRect =
object001.getBoundingClientRect();

    console.log('X: ' + objectRect.x + ', Y: ' +
objectRect.y);
```

```
        alert('X: ' + objectRect.x + ', Y: ' +
objectRect.y);
    };

    document.body.append(object001);

    /*----*/

    let sun = document.createElement('div');
    sun.style.position = 'absolute';
    sun.style.left = '300px';
    sun.style.bottom = '400px';
    sun.style.width = '100px';
    sun.style.height = '100px';
    sun.style.borderRadius = "50%";
    sun.style.backgroundColor = 'yellow';
    sun.style.cursor = 'pointer';
    sun.style.zIndex = '10000';
    document.body.append(sun);
```

```
}());
```

```
/*
Result:
X: 400, Y: 500
*/
```

```
javascript:(
/* createElement - Get link textContent from
Webpage - right click removes element */
function()
{
    function createInfoDiv()
    {
        let report = "";

        let theLinks =
document.getElementsByTagName("a");

        for (let x = 0; x < theLinks.length; x++)
        {
            report += '<a href = " ' + theLinks[x] + ' "
target = "_blank">' + theLinks[x].textContent +
'</a>';

            report += "<hr>";
```

```
}

let ourDiv = document.createElement("div");
ourDiv.id = "theDiv";
ourDiv.style.position = "absolute";
ourDiv.style.left = "100px";
ourDiv.style.top = "100px";
ourDiv.style.width = "300px";
ourDiv.style.height = "200px";
ourDiv.style.paddingLeft = "10px";
ourDiv.style.paddingRight = "10px";
ourDiv.style.paddingBottom = "5px";
ourDiv.style.paddingTop = "5px";
ourDiv.style.borderRadius = "8px";
ourDiv.style.background = "rgb(0, 0, 0)";
ourDiv.style.zIndex = "1000";
ourDiv.style.fontSize = 22 + "px";
ourDiv.style.color = "rgb(0, 255, 255)";
ourDiv.style.overflowY = "scroll";
```

```
    ourDiv.innerHTML = report;

    ourDiv.oncontextmenu = function()
    {
  document.getElementById("theDiv").remove();
    };

    document.body.append(ourDiv);
  }

  createInfoDiv();

}());

/* For example: we can search Google for the
word News and then activate our script */
```

/*
The div is created on the page and the links of the page are shown in the div and are clickable. The div is also scrollable.
*/

/*
Choose the sytax style that you like using more:

```
report += "<a href = ' " + theLinks[x] + " ' target =
'_blank'>" + theLinks[x].textContent + "</a>";

report += '<a href = " ' + theLinks[x] + ' " target =
"_blank">' + theLinks[x].textContent + '</a>';

report += '<a href = \" + theLinks[x] + '\' target =
"_blank"> '+theLinks[x].textContent+' </a>';

report += `<a href = "${theLinks[x]}" target =
"_blank">${theLinks[x].textContent}</a>`;

*/
```

```
javascript:(
/* createElement - Get URL of Page when button
is clicked - onmouseover, onmouseout */
function()
{
    function showUrl()
    {
        let ourButton =
document.createElement("button");
        ourButton.style.position = "fixed";
        ourButton.style.right = 5 + "px";
        ourButton.style.top = 5 + "px";
        ourButton.style.borderRadius = 8 + "px";
        ourButton.style.paddingLeft = 10 + "px";
        ourButton.style.paddingRight = 10 + "px";
        ourButton.style.paddingBottom = 5 + "px";
        ourButton.style.paddingTop = 5 + "px";
        ourButton.style.zIndex = "1000";
        ourButton.style.background = "rgb(0, 0, 0)";
```

```
ourButton.style.fontSize = 20 + "px";
ourButton.style.fontWeight = "bold";
ourButton.style.color = "rgb(255, 255, 255)";

ourButton.onclick = function()
{
    console.log(window.location.href);
    alert(window.location.href);
};

ourButton.onmouseover = function()
{
    ourButton.style.borderColor = "aqua";
    ourButton.style.color = "aqua";
};

ourButton.onmouseout = function()
{
    ourButton.style.borderColor = "white";
```

```
        ourButton.style.color = "white";
    };

    ourButton.innerHTML = "Show URL";

    document.body.append(ourButton);
  }

  showUrl();

}());
```

/* onmouseover the button changes style, onmouseout the button changes style again. onclick it shows the url of the page in a console message and alert message box */

```
javascript:(
/* createElement, Video pause button */
function()
{
    function videoPause()
    {
        let ourButton =
document.createElement("button");
        ourButton.id = "theButton";
        ourButton.style.position = "absolute";
        ourButton.style.left = "100px";
        ourButton.style.top = "100px";
        ourButton.style.paddingLeft = 10 + "px";
        ourButton.style.paddingRight = 10 + "px";
        ourButton.style.paddingTop = 5 + "px";
        ourButton.style.paddingBottom = 5 + "px";
        ourButton.style.borderRadius = 8 + "px";
        ourButton.style.zIndex = "1000";
        ourButton.style.background = "rgb(0, 0, 0)";
```

```
ourButton.style.fontSize = 20 + "px";
ourButton.style.fontWeight = "bold";
ourButton.style.color = "rgb(255, 255, 255)";
ourButton.innerHTML = "Pause Button";

ourButton.onmouseover = function()
{
    ourButton.style.color = "rgb(0, 255, 255)";
};

ourButton.onmouseout = function()
{
    ourButton.style.color = "rgb(255, 255,
255)";
};

ourButton.onclick = function()
{
```

```
        let theVideo =
document.querySelectorAll("video");

        theVideo.pause();
      };

      document.body.append(ourButton);

    }

    videoPause();

}());

/* When we have a video playing in our browser
we can pause it using this button. */
```

```
javascript:(
/* createElement, Video, Play, Pause Buttons */
function()
{
    function createPausePlayButtons()
    {
        let playButton =
document.createElement("button");
        playButton.id = "playButton";
        playButton.style.position = "absolute";
        playButton.style.left = 100 + "px";
        playButton.style.top = 100 + "px";
        playButton.style.paddingLeft = 10 + "px";
        playButton.style.paddingRight = 10 + "px";
        playButton.style.paddingTop = 5 + "px";
        playButton.style.paddingBottom = 5 + "px";
        playButton.style.borderRadius = 8 + "px";
        playButton.style.zIndex = 1000;
       playButton.style.background = "rgb(0, 0, 0)";
```

```
playButton.style.fontSize = 20 + "px";
playButton.style.fontWeight = "bold";
playButton.style.color = "rgb(255, 255, 255)";
playButton.innerHTML = "Play Button";

playButton.onmouseover = function()
{
   playButton.style.color = "rgb(0, 255, 255)";
};

playButton.onmouseout = function()
{
playButton.style.color = "rgb(255, 255, 255)";
};

playButton.onclick = function()
{
    let theVideo =
document.querySelectorAll("video");
```

```
        theVideo.play();
    };

    document.body.append(playButton);

    /*----*/

    let pauseButton =
document.createElement("button");
    pauseButton.id = "pauseButton";
    pauseButton.style.position = "absolute";
    pauseButton.style.left = 100 + "px";
    pauseButton.style.top = 150 + "px";
    pauseButton.style.paddingLeft = 10 + "px";
    pauseButton.style.paddingRight = 10 +
"px";
    pauseButton.style.paddingTop = 5 + "px";
```

```
        pauseButton.style.paddingBottom = 5 +
"px";
        pauseButton.style.borderRadius = 8 + "px";
        pauseButton.style.zIndex = 1000;
        pauseButton.style.background = "rgb(0, 0,
0)";
        pauseButton.style.fontSize = 20 + "px";
        pauseButton.style.fontWeight = "bold";
        pauseButton.style.color = "rgb(255, 255,
255)";
        pauseButton.innerHTML = "Pause Button";

        pauseButton.onmouseover = function()
        {
            pauseButton.style.color = "rgb(0, 255,
255)";
        };

        pauseButton.onmouseout = function()
```

```
        {
            pauseButton.style.color = "rgb(255, 255,
255)";
        };

        pauseButton.onclick = function()
        {
            let theVideo =
document.querySelectorAll("video");

            theVideo.pause();
        };
        document.body.append(pauseButton);
    }

    createPausePlayButtons();

}());
```

```
javascript:(
/* createElement Video, play, pause, back,
forward, Mute, UnMute */
function()
{
    function createVideoControls()
    {
        let thePaddingLeft = 10;
        let thePaddingRight = 10;
        let thePaddingTop = 5;
        let thePaddingBottom = 5;
        let theMargin = 1;
        let theBorderRadius = 8;
        let theZIndex = 1000;
        let theFontSize = 16;
        let theTextColor = "rgb(255, 255, 255)";
        let theBackgroundColor = "rgba(0, 0, 0,
0.3)";
```

```
    let theBorderColor = "rgba(255, 255, 255,
0.3)";
    let theFontWeight = "bold";

    /*----*/

    let mainDiv =
document.createElement("div");
    mainDiv.style.position = "absolute";
    mainDiv.style.left = 5 + "px";
    mainDiv.style.top = 70 + "px";
    mainDiv.style.display = "flex";
    mainDiv.style.flexDirection = "column";
    document.body.append(mainDiv);

    /*----*/

    let playButton =
document.createElement("button");
```

```
playButton.id = "playButton";
playButton.style.paddingLeft =
thePaddingLeft + "px";
playButton.style.paddingRight =
thePaddingRight + "px";
playButton.style.paddingTop =
thePaddingTop + "px";
playButton.style.paddingBottom =
thePaddingBottom + "px";
playButton.style.margin = theMargin + "px";
playButton.style.borderRadius =
theBorderRadius + "px";
playButton.style.borderColor =
theBorderColor;
playButton.style.zIndex = theZIndex;
playButton.style.background =
theBackgroundColor;
playButton.style.fontSize = theFontSize +
"px";
```

```
playButton.style.fontWeight = theFontWeight;
playButton.style.color = theTextColor;
playButton.innerHTML = "Play";

playButton.onmouseover = function()
{
    playButton.style.color = "rgb(0, 255, 255)";
};

playButton.onmouseout = function()
{
    playButton.style.color = theTextColor;
};

playButton.onclick = function()
{
```

```
        let theVideo =
document.querySelector("video");

        theVideo.play();
    };

    mainDiv.append(playButton);

    /*----*/

    let pauseButton =
document.createElement("button");
    pauseButton.id = "pauseButton";
    pauseButton.style.paddingLeft =
thePaddingLeft + "px";
    pauseButton.style.paddingRight =
thePaddingRight + "px";
    pauseButton.style.paddingTop =
thePaddingTop + "px";
```

```
        pauseButton.style.paddingBottom =
thePaddingBottom + "px";
        pauseButton.style.margin = theMargin +
"px";
        pauseButton.style.borderRadius =
theBorderRadius + "px";
        pauseButton.style.borderColor =
theBorderColor;
        pauseButton.style.zIndex = theZIndex;
        pauseButton.style.background =
theBackgroundColor;
        pauseButton.style.fontSize = theFontSize +
"px";
        pauseButton.style.fontWeight =
theFontWeight;
        pauseButton.style.color = theTextColor;
        pauseButton.innerHTML = "Pause";

        pauseButton.onmouseover = function()
```

```
        {
            pauseButton.style.color = "rgb(0, 255,
255)";
        };

        pauseButton.onmouseout = function()
        {
            pauseButton.style.color = theTextColor;
        };

        pauseButton.onclick = function()
        {
            let theVideo =
document.querySelector("video");

        theVideo.pause();
        };

        mainDiv.append(pauseButton);
```

```
/*----*/

let backButton =
document.createElement("button");
backButton.id = "backButton";
backButton.style.paddingLeft =
thePaddingLeft + "px";
backButton.style.paddingRight =
thePaddingRight + "px";
backButton.style.paddingTop =
thePaddingTop + "px";
backButton.style.paddingBottom =
thePaddingBottom + "px";
backButton.style.margin = theMargin +
"px";
backButton.style.borderRadius =
theBorderRadius + "px";
```

```
        backButton.style.borderColor =
theBorderColor;
        backButton.style.zIndex = theZIndex;
        backButton.style.background =
theBackgroundColor;
        backButton.style.fontSize = theFontSize +
"px";
        backButton.style.fontWeight =
theFontWeight;
        backButton.style.color = theTextColor;
        backButton.innerHTML = "Back";

        backButton.onmouseover = function()
        {
            backButton.style.color = "rgb(0, 255,
255)";
        };

        backButton.onmouseout = function()
```

```
        {
            backButton.style.color = theTextColor;
        };

        backButton.onclick = function()
        {
            let theVideo =
document.querySelector("video");

            theVideo.currentTime += -2;
        };

        mainDiv.append(backButton);

        /*----*/

        let forwardButton =
document.createElement("button");
        forwardButton.id = "forwardButton";
```

```
        forwardButton.style.paddingLeft =
thePaddingLeft + "px";
        forwardButton.style.paddingRight =
thePaddingRight + "px";
        forwardButton.style.paddingTop =
thePaddingTop + "px";
        forwardButton.style.paddingBottom =
thePaddingBottom + "px";
        forwardButton.style.margin = theMargin +
"px";
        forwardButton.style.borderRadius =
theBorderRadius + "px";
        forwardButton.style.borderColor =
theBorderColor;
        forwardButton.style.zIndex = theZIndex;
        forwardButton.style.background =
theBackgroundColor;
        forwardButton.style.fontSize = theFontSize
+ "px";
```

```
        forwardButton.style.fontWeight =
theFontWeight;
        forwardButton.style.color = theTextColor;
        forwardButton.innerHTML = "Forward";

        forwardButton.onmouseover = function()
        {
            forwardButton.style.color = "rgb(0, 255,
255)";
        };

        forwardButton.onmouseout = function()
        {
            forwardButton.style.color = theTextColor;
        };

        forwardButton.onclick = function()
        {
```

```
        let theVideo =
document.querySelector("video")[0];

        theVideo.currentTime += 2;
    };

    mainDiv.append(forwardButton);

    /*----*/

    let speedButton =
document.createElement("button");
    speedButton.id = "speedButton";
    speedButton.style.paddingLeft =
thePaddingLeft + "px";
    speedButton.style.paddingRight =
thePaddingRight + "px";
    speedButton.style.paddingTop =
thePaddingTop + "px";
```

```
        speedButton.style.paddingBottom =
thePaddingBottom + "px";
        speedButton.style.margin = theMargin +
"px";
        speedButton.style.borderRadius =
theBorderRadius + "px";
        speedButton.style.borderColor =
theBorderColor;
        speedButton.style.zIndex = theZIndex;
        speedButton.style.background =
theBackgroundColor;
        speedButton.style.fontSize = theFontSize +
"px";
        speedButton.style.fontWeight =
theFontWeight;
        speedButton.style.color = theTextColor;
        speedButton.innerHTML = "Speed";

        speedButton.onmouseover = function()
```

```
        {
            speedButton.style.color = "rgb(0, 255,
255)";
        };

        speedButton.onmouseout = function()
        {
            speedButton.style.color = theTextColor;
        };

        speedButton.onclick = function()
        {
            let video =
document.querySelector("video");

            let speedInput = prompt("Enter speed",
"");
```

```
        video.playbackRate =
parseFloat(speedInput);
    };

    mainDiv.append(speedButton);

    /*----*/

    let muteButton =
document.createElement("button");
    muteButton.id = "muteButton";
    muteButton.style.paddingLeft =
thePaddingLeft + "px";
    muteButton.style.paddingRight =
thePaddingRight + "px";
    muteButton.style.paddingTop =
thePaddingTop + "px";
    muteButton.style.paddingBottom =
thePaddingBottom + "px";
```

```
        muteButton.style.margin = theMargin +
"px";
        muteButton.style.borderRadius =
theBorderRadius + "px";
        muteButton.style.borderColor =
theBorderColor;
        muteButton.style.zIndex = theZIndex;
        muteButton.style.background =
theBackgroundColor;
        muteButton.style.fontSize = theFontSize +
"px";
        muteButton.style.fontWeight =
theFontWeight;
        muteButton.style.color = theTextColor;
        muteButton.innerHTML = "Mute";

        muteButton.onmouseover = function()
        {
```

```
        muteButton.style.color = "rgb(0, 255,
255)";
    };

    muteButton.onmouseout = function()
    {
        muteButton.style.color = theTextColor;
    };

    muteButton.onclick = function()
    {
       document.querySelector("video").muted =
true;
    };

    mainDiv.append(muteButton);

    /*----*/
```

```
    let unmuteButton =
document.createElement("button");
    unmuteButton.id = "unmuteButton";
    unmuteButton.style.paddingLeft =
thePaddingLeft + "px";
    unmuteButton.style.paddingRight =
thePaddingRight + "px";
    unmuteButton.style.paddingTop =
thePaddingTop + "px";
    unmuteButton.style.paddingBottom =
thePaddingBottom + "px";
    unmuteButton.style.margin = theMargin +
"px";
    unmuteButton.style.borderRadius =
theBorderRadius + "px";
    unmuteButton.style.borderColor =
theBorderColor;
    unmuteButton.style.zIndex = theZIndex;
```

```
        unmuteButton.style.background =
theBackgroundColor;
        unmuteButton.style.fontSize = theFontSize
+ "px";
        unmuteButton.style.fontWeight =
theFontWeight;
        unmuteButton.style.color = theTextColor;
        unmuteButton.innerHTML = "Unmute";

        unmuteButton.onmouseover = function()
        {
            unmuteButton.style.color = "rgb(0, 255,
255)";
        };

        unmuteButton.onmouseout = function()
        {
            unmuteButton.style.color = theTextColor;
        };
```

```
        unmuteButton.onclick = function()
        {
            document.querySelector("video").muted
= false;
        };

        mainDiv.append(unmuteButton);
    }

    createVideoControls();

}());
```

Play
Pause
Back
Forward
Speed
Mute
Unmute
CollegeOfScripting.weebly.com
0:00 / 3:55
Topalian Comic Book Creator Coded in JavaScript
ScriptingCollege
668 subscribers
Subscribe
1
Share
All
From ScriptingCollege
Indie games

```
javascript:(
/* Show elements that have a certain class name
on https://CollegeOfScripting.weebly.com */
function()
{
    function showData()
    {
        let nameList =
document.querySelectorAll(".buttonStyle");

        let theNames = [];

        for (let x = 0; x < nameList.length; x++)
        {
            nameList[x].style.borderColor = "rgb(0,
255, 255)";

            theNames += nameList[x].textContent;
```

```
            theNames += "\n";
        }

        return theNames.toString();
    }

    console.log(showData());

    alert(showData());

}());

/*
This Bookmarklet is designed to work ONLY on
https://CollegeOfScripting.weebly.com
Right Click on the buttons, such as the
JavaScript button and then
Choose Inspect Element
```

We see that the Element has a css style class called .buttonStyle
We use the dot syntax before the class name.
The css style class name is .buttonStyle
Inspecting an Element
After Right Clicking on the Button and choosing Inspect Element, the Inspector Opens and allows us to see the name of the class of that Button.
In this case, the buttons of this Weebly Website have a class called
buttonStyle
Thus, we simply use that class name, when we want to reference only elements that use that class. This allows us to examine any webpage and inspect any element to find their class name, which therefore allows us the ability to reference any elements on the page that use that class.
*/

Inspecting an Element

/* After Right Clicking on the button and choosing Inspect Element, the Inspector Opens and shows the name of the class of that button. In this case, the button of this Weebly Website has a class named **buttonStyle** */

```
javascript:(
/* Solar Wind Speed -
https://www.swpc.noaa.gov/ */
function()
{
    function getSolarWindSpeed()
    {
        let solarWindSpeed =
document.querySelector("#WindSpeed").textCo
ntent;

        return solarWindSpeed;
    }

    function showData()
    {
        let solarWindSpeed = "Solar Wind Speed: "
+ getSolarWindSpeed() + " km/second";
```

```
        console.log(solarWindSpeed);

        alert(solarWindSpeed);
    }

    showData();

}());
```

https://www.swpc.noaa.gov

Solar Wind Speed: 354 km/sec

Solar Wind Magnetic Fields: Bt **6** nT, Bz **-1** nT

Inspector | Console | Debugger | Network | Style Editor

earch HTML

```
<li>
<span class="summary-field-label">Solar Wind Speed:</span>
<span id="WindSpeed" class="summary-field-value">354</span>
km/sec
</li>
```

```
/*
Solar Wind Speed: 354 km/second

When we go to https://www.swpc.noaa.gov and
right click on the Solar Wind Speed and choose
Inspect, we find out many things about the
element, including the id, which we use in this
script to get the textContent of that id.
*/
```

```
javascript:(
/* Solar Wind Speed -
https://www.swpc.noaa.gov - Timer */
function()
{
    function getSolarWindSpeed()
    {
        let solarWindSpeed =
document.querySelector("#WindSpeed").textCo
ntent;

        return solarWindSpeed;
    }

    function showData()
    {
        let solarWindSpeed = "Solar Wind Speed: "
+ getSolarWindSpeed() + " km/second";
```

```
        console.log(solarWindSpeed);
    }

    showData();

    let theTimer = setInterval(function()
    {
        showData();
    }, 60 * 1000);

}());

/*
We go to https://www.swpc.noaa.gov and then
activate this script.

This script will keep getting the Solar Wind
Speed from the element id "WindSpeed"
textContent. */
```

```
javascript:(
/* Solar Wind Speed -
https://www.swpc.noaa.gov - Timer - Array */
function()
{
    let solarWindArray = [];

    function getSolarWindSpeed()
    {
        let solarWindSpeed =
document.querySelector("#WindSpeed").textCo
ntent;

        let date = new Date();

        let dateLocal = date.toLocaleString();

        return dateLocal + " - Solar Wind Speed: " +
solarWindSpeed + " km/second";
    }
```

```
    /* push data into the array */
    solarWindArray.push(getSolarWindSpeed());

    console.log(JSON.stringify(solarWindArray));

    let theTimer = setInterval(function()
    {
      /* push data into the array on each iteration */
       solarWindArray.push(getSolarWindSpeed());

      console.log(JSON.stringify(solarWindArray));

    }, 60 * 1000);

}());

/*
When activated on https://www.swpc.noaa.gov
this script will get the Solar Wind Speed every 1
```

minute and add the solar wind speed to the array, in addition to the date and time.

The first time that the script gathers the solar wind speed it looks like this:
["8/28/2023, 6:54:56 AM - Solar Wind Speed: 327 km/second"]

The second time that the script gathers the solar wind speed it looks like this:
["8/28/2023, 6:54:17 AM - Solar Wind Speed: 327 km/second","8/28/2023, 6:55:17 AM - Solar Wind Speed: 325 km/second"]
*/

```
javascript:(
/* Solar Wind Speed -
https://www.swpc.noaa.gov - Timer - Array of
Objects -*/
function()
{
    let solarWindArray = [];

    function getSolarWindSpeed()
    {
        let solarWindSpeed =
document.querySelector("#WindSpeed").textCo
ntent;

        let date = new Date();

        let dateLocal = date.toLocaleString();

        return {
            date: dateLocal,
```

```
        speed: solarWindSpeed
    };
}

/* push data into the array */
solarWindArray.push(getSolarWindSpeed());

console.log(JSON.stringify(solarWindArray));

let theTimer = setInterval(function()
{
  /* push data into the array on each iteration */
  solarWindArray.push(getSolarWindSpeed());

  console.log(JSON.stringify(solarWindArray));

}, 60 * 1000);

}());
```

/*

When activated on https://www.swpc.noaa.gov this script will get the solar wind speed every 1 minute and add the solar wind speed to the array of objects, in addition to the date and time.

The first time that the script gathers the solar wind speed it looks like this:
[{"date":"8/28/2023, 7:15:49 AM","speed":"323"}]

The second time that the script gathers the solar wind speed it looks like this:
[{"date":"8/28/2023, 7:15:49 AM","speed":"323"},
{"date":"8/28/2023, 7:16:49 AM","speed":"325"}]

The third time that the script gathers the solar wind speed it looks like this:
[{"date":"8/28/2023, 7:15:49 AM","speed":"323"},
{"date":"8/28/2023, 7:16:49 AM","speed":"325"},
{"date":"8/28/2023, 7:17:49 AM","speed":"324"}]

The fourth time that the script gathers the solar wind speed it looks like this:
[{"date":"8/28/2023, 7:15:49 AM","speed":"323"},
{"date":"8/28/2023, 7:16:49 AM","speed":"325"},
{"date":"8/28/2023, 7:17:49 AM","speed":"324"},
{"date":"8/28/2023, 7:18:49 AM","speed":"323"}]

*/

```
javascript:(
/* Timer - Every 5 Seconds, Trigger function */
function()
{
    let counter = 0;

    function showCounterData()
    {
        console.log(counter);

        counter += 1;
    }

    setInterval(showCounterData, 5000);

}());
```

```
javascript:(
/* Timer - Every 1 Second, COUNT UP */
function()
{
    let counter = 0;

    function updateIt()
    {
        counter += 1;

        return counter;
    }

    function createCounterDiv()
    {
        let ourDiv = document.createElement("div");
        ourDiv.id = "theDiv";
        ourDiv.style.position = "absolute";
        ourDiv.style.left = 20 + "px";
```

```
        ourDiv.style.top = 150 + "px";
        ourDiv.style.width = 100 + "px";
        ourDiv.style.height = 25 + "px";
        ourDiv.style.padding = 10 + "px";
        ourDiv.style.backgroundColor = "rgb(0, 0,
0)";
        ourDiv.style.fontSize = 20 + "px";
        ourDiv.style.color = "rgb(255, 255, 255)";
        ourDiv.innerHTML = updateIt();
        document.body.append(ourDiv);
    }

    setInterval(createCounterDiv, 1000);

}());
```

```
javascript:(
/* Timer - Every 1 Second, COUNT DOWN,
clearInterval */
function()
{
    let counter = prompt("Enter Count Down Start
Time");

    function countDown()
    {
        counter -= 1;

        if (counter == 0)
        {
            clearInterval(ourTimer);
        }
        return counter;
    }
```

```
function createCounterDiv()
{
    let ourDiv = document.createElement("div");
    ourDiv.id = "theDiv";
    ourDiv.style.position = "absolute";
    ourDiv.style.left = 100 + "px";
    ourDiv.style.top = 100 + "px";
    ourDiv.style.padding = 10 + "px";
    ourDiv.style.borderRadius = 8 + "px";
  ourDiv.style.backgroundColor = "rgb(0, 0, 0)";
    ourDiv.style.fontSize = 25 + "px";
    ourDiv.style.fontWeight = "bold";
    ourDiv.style.color = "rgb(255, 255, 255)";
    ourDiv.innerHTML = countDown();
    document.body.append(ourDiv);
}
let ourTimer = setInterval(createCounterDiv,
1000);
}());
```

```
javascript:(
/* Array of Objects - JSON.stringify */
function()
{
    let people =
    [
        {
            name: 'Melissa',
            date: '1980/03/01',
        },
        {
            name: 'Tabitha',
            date: '1983/04/05'
        }
    ];

    function showData(whichArray)
    {
        console.log(whichArray);
```

```
        console.log(JSON.stringify(whichArray));
    }

    showData(people);

}());

/*
Here is the Output

console.log(people);
0: Object { name: "Melissa", date: "1980/03/01" }

1: Object { name: "Tabitha", date: "1983/04/05" }

console.log(JSON.stringify(people));
[{"name":"Melissa","date":"1980/03/01"},
{"name":"Tabitha","date": "1983/04/05"}]
*/
```

```
javascript:(
/* Array of Objects - for loop */
function()
{
    let people =
    [
        {
            name: 'Melissa',
            date: '1980/03/01',
        },

        {
            name: 'Tabitha',
            date: '1983/04/05'
        }
    ];

    function showData(whichArray)
    {
```

```
        for (let x = 0; x < whichArray.length; x++)
        {
            console.log(whichArray[x]);
        }
    }

    showData(people);

}());

/*

Object { name: "Melissa", date: "1980/03/01" }

Object { name: "Tabitha", date: "1983/04/05" }

*/
```

```
javascript:(
/* Array of Objects - while loop */
function()
{
    let people =
    [
        {
            name: 'Melissa',
            date: '1980/03/01',
        },

        {
            name: 'Tabitha',
            date: '1983/04/05'
        }
    ];

    function showData(whichArray)
    {
```

```
        let x = 0;

        while (x < whichArray.length)
        {
            console.log(whichArray[x]);

            x += 1;
        }
    }

    showData(people);

}());

/*
{ name: 'Melissa', date: '1980/03/01' }
{ name: 'Tabitha', date: '1983/04/05' }
*/
```

```
javascript:(
/* Array of Objects - Names only */
function()
{
    let people =
    [
        {
            name: 'Melissa',
            date: '1980/03/01',
        },

        {
            name: 'Tabitha',
            date: '1983/04/05'
        }
    ];

    function showData(whichArray)
    {
```

```
        let x = 0;

        while (x < whichArray.length)
        {
            console.log(whichArray[x].name);

            x += 1;
        }
    }

    showData(people);

}());

/*
Melissa
Tabitha
*/
```

```
javascript:(
/* Array of Objects - Dates only */
function()
{
    let people =
    [
        {
            name: 'Melissa',
            date: '1980/03/01',
        },

        {
            name: 'Tabitha',
            date: '1983/04/05'
        }
    ];

    let x = 0;
```

```
    while (x < people.length)
    {
        console.log(people[x].date);

        x += 1;
    }

}());

/*
1980/03/01
1983/04/05
*/
```

```
javascript:(
/* Array of Objects - for loop */
function()
{
    let people =
    [
        {
            name: 'Melissa',
            date: '1980/03/01',
        },
        {
            name: 'Tabitha',
            date: '1983/04/05'
        }
    ];

    function showData(whichArray)
    {
        let output = 'Name\t\tDate\n';
```

```
        for (let x = 0; x < whichArray.length; x++)
        {
            output = output + whichArray[x].name;

            output += "\t";

            output += whichArray[x].date;

            output += "\n";
        }

        console.log(output);

        alert(output);
    }

    showData(people);

}());
```

```
/*
Name        Date
Melissa  1980/03/01
Tabitha  1983/04/05
*/
```

```
javascript:(
/* Array of Objects - First Letter Initial */
function()
{
    let people =
    [
        {
            name: 'Melissa',
            date: '1980/03/01',
        },

        {
            name: 'Tabitha',
            date: '1983/04/05'
        }
    ];

    function showFirstLetterOfName(whichArray)
    {
```

```
        let x = 0

        while (x < whichArray.length)
        {
            console.log(whichArray[x].name[0]);

            x += 1;
        }
    }

    showFirstLetterOfName(people);

}());

/*
M
T
*/
```

```
javascript:(
/* Array of Objects - First 3 Letter Initials  */
function()
{
    let people =
    [
        {
            name: 'Melissa',
            date: '1980/03/01',
        },

        {
            name: 'Tabitha',
            date: '1983/04/05'
        }
    ];

    function showData(whichArray)
    {
```

```
        let x = 0;

        while (x < whichArray.length)
        {
            console.log(whichArray[x]['name'][0] +
            whichArray[x]['name'][1] +
            whichArray[x]['name'][2]);

            x += 1;
        }
    }

    showData(people);

}());

/*
Mel
Tab */
```

```
javascript:(
/* Array of Objects - Filter by Year */
function()
{
    let people =
    [
        {
            name: 'Melissa',
            date: '1980/03/01',
        },

        {
            name: 'Tabitha',
            date: '1983/04/05'
        },

        {
            name: 'Jane',
            date: '1987/08/12'
        }
```

```
];

function sortByDate(whichArray, direction)
{
    if (direction == "up")
    {
        whichArray.sort(function(a, b)
        {
            return new Date(a.date) - new
Date(b.date);
        });
    }

    else if (direction == "down")
    {
        whichArray.sort(function(a, b)
        {
            return new Date(b.date) - new
Date(a.date);
        });
```

```
        }
    }

    function showData(whichArray)
    {
        for (let x = 0; x < whichArray.length; x++)
        {
            if (whichArray[x].date >= "1981")
            {
                console.log(whichArray[x].name + "\n"
+ whichArray[x].date);
            }
        }
    }

    sortByDate(people, "up");

    showData(people);

}());
```

```
/*
Tabitha
1983/04/05

Jane
1987/08/12
*/
```

```
javascript:(
/* Array of Objects - Filter by Year and Month */
function()
{
    let people =
    [
        {
            name: 'Melissa',
            date: '1980/03/01',
        },

        {
            name: 'Tabitha',
            date: '1983/04/05'
        },

        {
            name: 'Joan',
            date: '1983/05/17'
        },
```

```
        {
            name: 'Jane',
            date: '1987/08/12'
        }
    ];

    function sortByDate(whichArray, direction)
    {
        if (direction == "up")
        {
            whichArray.sort(function(a, b)
            {
                return new Date(a.date) - new
Date(b.date);
            });
        }

        else if (direction == "down")
        {
```

```
            whichArray.sort(function(a, b)
            {
                return new Date(b.date) - new
Date(a.date);
            });
        }
    }

    function showData(whichArray)
    {
        for (let x = 0; x < whichArray.length; x++)
        {
            if (whichArray[x].date >= "1983/05")
            {
                console.log(whichArray[x].name + "\n"
+ whichArray[x].date);
            }
        }
    }
```

```
    sortByDate(people, "up");

    showData(people);

}());

/*
Joan
1983/05/17

Jane
1987/08/12
*/
```

```
javascript:(
/* Array of Objects - Filter by Year, Month and
Day */
function()
{
    let people =
    [
        {
            name: 'Melissa',
            date: '1980/03/01',
        },

        {
            name: 'Tabitha',
            date: '1983/04/05'
        },

        {
            name: 'Joan',
            date: '1983/05/17'
```

```
        },

        {
            name: 'Jane',
            date: '1987/08/12'
        }
    ];

    function sortByDate(whichArray, direction)
    {
        if (direction == "up")
        {
            whichArray.sort(function(a, b)
            {
                return new Date(a.date) - new
Date(b.date);
            });
        }

        else if (direction == "down")
```

```
        {
            whichArray.sort(function(a, b)
            {
                return new Date(b.date) - new
Date(a.date);
            });
        }
    }

    function showData(whichArray)
    {
        for (let x = 0; x < whichArray.length; x++)
        {
            if (whichArray[x].date == "1983/04/05")
            {
                console.log(whichArray[x].name + "\n"
+ whichArray[x].date);
            }
        }
    }
```

```
    sortByDate(people, "up");

    showData(people);

}());

/*
Tabitha
1983/04/05
*/
```

```
javascript:(
/* Array of Objects - Filter by Year, Month, Day
and Time */
function()
{
    let people =
    [
        {
            name: 'Melissa',
            date: '1980/03/01 12:00PM',
        },

        {
            name: 'Tabitha',
            date: '1983/04/05 2:57PM'
        },

        {
            name: 'Jennifer',
            date: '1983/05/17 3:45PM'
```

```
    },

    {
        name: 'Joan',
        date: '1983/05/17 4:07PM'
    },

    {
        name: 'Jane',
        date: '1987/08/12 8:23PM'
    }
];

function sortByDate(whichArray, direction)
{
    if (direction == "up")
    {
        whichArray.sort(function(a, b)
        {
```

```
            return new Date(a.date) - new
Date(b.date);
        });
    }

    else if (direction == "down")
    {
        whichArray.sort(function(a, b)
        {
            return new Date(b.date) - new
Date(a.date);
        });
    }
}

function showData(whichArray)
{
    for (let x = 0; x < whichArray.length; x++)
    {
```

```
            if (whichArray[x].date >= "1983/05/17
3:50PM")
            {
                console.log(whichArray[x].name + "\n"
+ whichArray[x].date);
            }
        }
    }

    sortByDate(people, "up");

    showData(people);
}());

/*
Joan
1983/05/17 4:07PM
Jane
1987/08/12 8:23PM
*/
```

```
javascript:(
/* Array of Objects - Filter Date A to Date B,
YYYY/MM/DD */
function()
{
    let people =
    [
        {
            name: 'Melissa',
            date: '1980/03/01 12:00PM',
        },

        {
            name: 'Tabitha',
            date: '1983/04/05 2:57PM'
        },

        {
            name: 'Jennifer',
            date: '1983/05/17 3:45PM'
```

```
    },

    {
        name: 'Joan',
        date: '1983/05/17 4:07PM'
    },

    {
        name: 'Jane',
        date: '1987/08/12 8:23PM'
    }
];

function sortByDate(whichArray, direction)
{
    if (direction == "up")
    {
        whichArray.sort(function(a, b)
        {
```

```
            return new Date(a.date) - new
Date(b.date);
        });
    }

    else if (direction == "down")
    {
        whichArray.sort(function(a, b)
        {
            return new Date(b.date) - new
Date(a.date);
        });
    }
}

function showData(whichArray)
{
    for (let x = 0; x < whichArray.length; x++)
    {
```

```
            if (whichArray[x].date >= "1983/05/17
3:45PM" && whichArray[x].date <= "1983/05/17
4:00PM")
            {
                console.log(whichArray[x].name + "\n"
+ whichArray[x].date);
            }
        }
    }

    sortByDate(people, "up");

    showData(people);

}());

/*
Jennifer
1983/05/17 3:45PM
*/
```

```
javascript:(
/* Array of Objects - Special way to show keys
and values */
function()
{
    let people =
    [
        {
            name: 'Melissa',
            date: '2021/04/01',
        },
        {
            name: 'Tabitha',
            date: '2021/04/05'
        }
    ];

    for (let x = 0; x < people.length; x++)
    {
        let person = people[x];
```

```
        let entries = Object.entries(person);

        for (let m = 0; m < entries.length; m++)
        {
            let key = entries[m][0];

            let value = entries[m][1];

            console.log(key + ': ' + value);
        }
    }

}());

/*
we use a for loop to iterate through this array
and access both the key and value for each
property in the object
```

```
name: Melissa
date: 2021/04/01
name: Tabitha
date: 2021/04/05

*/
```

```
javascript:(
/* Array of Objects with Nested Arrays */
function()
{
    function showData()
    {
        let data =
        [
            {
                name: 'James',
                age: 30,
                hobbies: ['Gardening', 'Gaming']
            },

            {
                name: 'Joan',
                age: 25,
                hobbies: ['Hiking', 'Painting']
            }
        ];
```

```
        let message = 'Array of Objects:\n';

        for (let x = 0; x < data.length; x++)
        {
            let person = data[x];

            message += 'Name: ' + person.name +
', Age: ' + person.age + ', Hobbies: [' +
person.hobbies.join(', ') + ']\n';
        }

        console.log(message);

        alert(message);
    }

    showData();

}());
```

```
javascript:(
/* Array of Objects - Show Data on Rows of Divs
*/
function()
{
    function showData()
    {
        let data =
        [
            {
                name: 'Item 1',
                description: 'Description 1'
            },

            {
                name: 'Item 2',
                description: 'Description 2'
            },

            {
```

```
        name: 'Item 3',
        description: 'Description 3'
    },

    {
        name: 'Item 4',
        description: 'Description 4'
    },

    {
        name: 'Item 5',
        description: 'Description 5'
    },

    {
        name: 'Item 6',
        description: 'Description 6'
    },

    {
```

```
            name: 'Item 7',
            description: 'Description 7'
        },

        {
            name: 'Item 8',
            description: 'Description 8'
        },

        {
            name: 'Item 9',
            description: 'Description 9'
        },
    ];

    /* create a container for the rows of divs */
    let container =
document.createElement('div');
    container.style.position = "absolute";
    container.style.left = 10 + "px";
```

```
    container.style.top = 10 + "px";
    container.style.maxWidth = '800px';
    container.style.padding = '10px';
    container.style.display = 'flex';
    container.style.flexWrap = 'wrap';
    container.style.justifyContent = 'space-
between';
    document.body.append(container);

    /* function to create a div for each object */
    function createDivForObject(object)
    {
        let dataDiv =
document.createElement('div');
        dataDiv.style.width = '30%';
        dataDiv.style.paddingLeft = '10px';
        dataDiv.style.paddingRight = '10px';
        dataDiv.style.paddingBottom = '5px';
        dataDiv.style.marginLeft = '1px';
        dataDiv.style.marginRight = '1px';
```

```
        dataDiv.style.marginBottom = '7px';
        dataDiv.style.marginTop = '7px';
        dataDiv.style.border = '1px solid';
        dataDiv.style.borderColor = 'rgb(255, 255,
255)';

        dataDiv.innerHTML = '<h3>' +
object.name + '</h3>' + '<p>' + object.description
+ '</p>';

        container.append(dataDiv);
    }

    /* create divs for each object in the array */
    for (let x = 0; x < data.length; x++)
    {
        createDivForObject(data[x]);
    }
}
```

```
    showData();

}());
```

Item 1
Description 1
Item 2
Description 2
Item 3
Description 3
Item 4
Description 4
Item 5
Description 5
Item 6
Description 6
Item 7
Item 8
Item 9

```
javascript:(
/* Array of Objects - map, filter, reduce - Finds
Ages Over Specified Age */
function()
{
    function
showNamesOverSpecifiedAge(whichAge)
    {
        let data =
        [
            {
                name: 'John',
                age: 30
            },
            {
                name: 'Jane',
                age: 25
            },
            {
                name: 'James',
```

```
        age: 35
    },
    {
        name: 'Alice',
        age: 28
    },
    {
        name: 'Joan',
        age: 40
    }
];

/* map to extract names */
let names = [];

for (let x = 0; x < data.length; x++)
{
    names.push(data[x].name);
}
console.log('Mapped Names:', names);
```

```
/*----*/

/* filter to find people over 30 */
let ageArray = [];

for (let x = 0; x < data.length; x++)
{
    if (data[x].age >= whichAge)
    {
        ageArray.push(data[x]);
    }
}
console.log('People Over ' + whichAge + '
Years Old:', ageArray);

/* reduce to calculate total age */
let totalAge = 0;

for (let x = 0; x < data.length; x++)
```

```
        {
            totalAge += data[x].age;
        }

        console.log('Total Age:', totalAge);

        /*----*/

        /* show each item */
        for (let x = 0; x < data.length; x++)
        {
            console.log('Name:', data[x].name, '|
Age:', data[x].age);
        }
    }

    showNamesOverSpecifiedAge(30);

}());
```

```
javascript:(
/* Array of Objects - Emoji Fun */
function()
{
    let funObjects =
    [
        {
            name: "Smiling Face",
            description: "A face with a big smile 😃",
        },

        {
            name: "Big Hat",
            description: "A big hat 🎩",
        },

        {
            name: "SunShades",
            description: "A pair of sunshades 😎",
        },
```

```
    ];

    let message = "Fun Objects:\n\n";

    for (let x = 0; x < funObjects.length; x++)
    {
        message += "Name: " + funObjects[x].name
+ "\n";

        message += "Description: " +
funObjects[x].description + "\n\n";
    }

    console.log(message);

    alert(message);

}());
```

```
javascript:(
/* Array of Objects - Random Object –
Programming Languages */
function()
{
    let programmingFacts =
    [
        {
            name: "C",
            fact: "Invented in 1972",
        },
        {
            name: "C++",
            fact: "Invented in 1985",
        },
        {
            name: "JavaScript",
            fact: "Invented in 1995",
        }
    ];
```

```
    function generateRandomFact(whichArray)
    {
        let randomIndex = Math.floor(Math.random()
* whichArray.length);

        let randomFact = whichArray[randomIndex];

        let message = randomFact.name + "\n" +
randomFact.fact + "\n";

        console.log(message);

        alert(message);
    }

    generateRandomFact(programmingFacts);

}());
```

```
javascript:(
/* Array of Objects - Random Object - Periodic
Elements */
function()
{
    let periodicTable =
    [
        {
            symbol: "H",
            name: "Hydrogen",
            atomicNumber: 1,
        },

        {
            symbol: "He",
            name: "Helium",
            atomicNumber: 2,
        },

        {
```

```
        symbol: "Li",
        name: "Lithium",
        atomicNumber: 3,
    },

    {
        symbol: "Be",
        name: "Beryllium",
        atomicNumber: 4,
    }
];

function generateRandomFact(whichArray)
{
    let randomIndex = Math.floor(Math.random()
* whichArray.length);

    let randomFact = whichArray[randomIndex];
```

```
        let message = randomFact.symbol + "\n" +
randomFact.name + "\n" +
randomFact.atomicNumber;

        console.log(message);

        alert(message);
    }

    generateRandomFact(periodicTable);

}());
```

```
javascript:(
/* Array of Objects - Calculate the Average Score
*/
function()
{
    function calculateAverageScore()
    {
        let students =
        [
            {
                name: "Alice",
                score: 85
            },

            {
                name: "Bob",
                score: 92
            },

            {
```

```
        name: "James",
        score: 78
    },

    {
        name: "David",
        score: 88
    },

    {
        name: "Jennifer",
        score: 95
    },
];

function calculateAverage(scores)
{
    let total = 0;

    for (let x = 0; x < scores.length; x++)
```

```
        {
            total += scores[x].score;
        }

        return total / scores.length;
    }

    /* calculate the average score */
    let averageScore =
calculateAverage(students);

    /* display the result */
    alert("Average Exam Score: " +
averageScore.toFixed(2));
  }

  calculateAverageScore();

}());
```

```
javascript:(
/* Array of Objects - Calculate Average Score,
Standard Deviation */
function()
{
    let students =
    [
        {
            name: "Alice",
            score: 85
        },

        {
            name: "Bob",
            score: 92
        },

        {
            name: "Jennifer",
            score: 78
```

```
    },

    {
        name: "David",
        score: 88
    },

    {
        name: "Joan",
        score: 95
    },
];

function calculateAverage(scores)
{
    let total = 0;

    for (let x = 0; x < scores.length; x++)
    {
        total += scores[x].score;
```

```
    }

    return total / scores.length;
}

function calculateStandardDeviation(scores)
{
    let average = calculateAverage(scores);

    let variance = 0;

    for (let x = 0; x < scores.length; x++)
    {
        variance += Math.pow(scores[x].score -
average, 2);
    }

    let stdDeviation = Math.sqrt(variance /
scores.length);
```

```
        return stdDeviation;
    }

    /* calculate the average score and standard
deviation */
    let averageScore =
calculateAverage(students);

    let standardDeviation =
calculateStandardDeviation(students);

    /* display the results */
    alert("Average Score: " +
averageScore.toFixed(2) + "\nStandard
Deviation: " + standardDeviation.toFixed(2));

}());
```

```
javascript:(
/* Array of Objects - Find Max/Min Temperature */
function()
{
    function calculateMinMaxTemperature()
    {
        let cities =
        [
            {
                name: "New York",
                temperature: 90
            },

            {
                name: "Los Angeles",
                temperature: 82
            },

            {
                name: "Chicago",
```

```
        temperature: 76
    },

    {
        name: "Boston",
        temperature: 95
    },

    {
        name: "Denver",
        temperature: 87
    },
];

function findMaxTemperature(whichArray)
{
    let max = whichArray[0].temperature;

    for (let x = 1; x < whichArray.length; x++)
    {
```

```
        if (whichArray[x].temperature > max)
        {
            max = whichArray[x].temperature;
        }
    }
    return max;
}

function findMinTemperature(whichArray)
{
    let min = whichArray[0].temperature;

    for (let x = 1; x < whichArray.length; x++)
    {
        if (whichArray[x].temperature < min)
        {
            min = whichArray[x].temperature;
        }
    }
    return min;
```

```
        }

        /* find the maximum and minimum
temperatures */
        let maxTemperature =
findMaxTemperature(cities);

        let minTemperature =
findMinTemperature(cities);

        /* show the results */
        alert("Maximum Temperature: " +
maxTemperature + "°F\nMinimum Temperature: "
+ minTemperature + "°F");
    }

    calculateMinMaxTemperature();

}());
```

```
javascript:(
/* Array of Objects - Calculate Sales Total */
function()
{
    function calculateSalesTotal()
    {
        let products =
        [
            {
                name: "Store A",
                sales: 250
            },
            {
                name: "Store B",
                sales: 180
            },
            {
                name: "Store C",
                sales: 320
            },
```

```
    {
        name: "Store D",
        sales: 420
    },
];

function calculateTotalSales(salesData)
{
    let total = 0;

    for (let x = 0; x < salesData.length; x++)
    {
        total += salesData[x].sales;
    }

    return total;
}

/* calculate the total sales */
```

```
        let totalSales =
calculateTotalSales(products);

        return "Total Sales: $" +
totalSales.toFixed(2);
    }

    /* display the result */
    console.log(calculateSalesTotal());

    alert(calculateSalesTotal());

}());
```

```
javascript:(
/* Array of Objects - Count occurrences of each
item */
function()
{
    function countOccurrencesOfEachItem()
    {
        let food =
        [
            {
                name: "Apple",
                quantity: 3
            },

            {
                name: "Banana",
                quantity: 2
            },

            {
```

```
        name: "Strawberry",
        quantity: 4
    },

    {
        name: "Apple",
        quantity: 2
    },

    {
        name: "Pear",
        quantity: 3
    },

    {
        name: "Banana",
        quantity: 1
    },
];
```

```
function countFruitOccurrences(items)
{
    let fruitCounts = {};

    for (let x = 0; x < items.length; x++)
    {
        let itemName = items[x].name;

        if (fruitCounts[itemName])
        {
            fruitCounts[itemName] +=
items[x].quantity;
        }
        else
        {
            fruitCounts[itemName] =
items[x].quantity;
        }
    }
    return fruitCounts;
```

```
    }

    /* count the occurrences of each fruit */
    let fruitOccurrences =
countFruitOccurrences(food);

    /* display the results */
    let message = "Fruit Occurrences:\n\n";

    let fruitKeys =
Object.keys(fruitOccurrences);

    for (let x = 0; x < fruitKeys.length; x++)
    {
        let fruit = fruitKeys[x];
message += fruit + ": " + fruitOccurrences[fruit]
+ "\n";
    }

    console.log(message);
```

```
        alert(message);
    }

    countOccurrencesOfEachItem();

}());

/*

Fruit Occurrences:

Apple: 5
Banana: 3
Strawberry: 4
Pear: 3

*/
```

```
javascript:(
/* Links - List and Count */
function()
{
    function listLinksAndCount()
    {
        let links = document.links;

        let linkCount = links.length;

        if (linkCount === 0)
        {
            alert('No links found on the page');
        }
        else
        {
            let linkSources = [];

            for (let x = 0; x < linkCount; x++)
            {
```

```
            linkSources.push(links[x].href);
        }

        let data = 'Total Links: ' +
        linkCount + '\n\n' +
        'Link Sources: ' +
        linkSources.join('\n');

        console.log(data);

        alert(data);
    }
}

listLinksAndCount();

}());
```

```
javascript:(
/* Element - List and Count Specified Type */
function()
{
    function listAndCount(elementType)
    {
        let elements =
document.getElementsByTagName(elementType
);

        let elementCount = elements.length;

        if (elementCount === 0)
        {
            alert('No ' + elementType + 's found on
this page');
        }
        else
        {
            let elementContents = [];
```

```
            for (let x = 0; x < elementCount; x++)
            {
elementContents.push(elements[x].textContent);
            }

            let data = 'Total ' +
            elementType + 's: ' +
            elementCount + '\n\n' +
            elementType + ' ' +
            'Contents:' + '\n' +
            elementContents.join('\n');

            console.log(data);
            alert(data);
        }
    }

    listAndCount('a');
}());
```

```
javascript:(
/* Bouncing Ball - Linear Motion - Bounces
Up/Down - No Trigonometry required */
function()
{
    function createBallWithMotion()
    {
        let ball = document.createElement('div');
        ball.id = 'ball';
        ball.style.position = "absolute";
        ball.style.width = "50px";
        ball.style.height = "50px";
        ball.style.borderRadius = "50%";
        ball.style.margin = "0px";
        ball.style.overflow = "hidden";
        ball.style.backgroundColor = "aqua";
        document.body.append(ball);

        /* 1 goes down, -1 goes up */
        let direction = 1;
```

```
let position = 0;

let speed = 2;

function animate()
{
    position += direction * speed;

    ball.style.top = position + 'px';

    if (position >= (window.innerHeight - 50) ||
position <= 0)
    {
        /* reverse direction at top or bottom */
        direction *= -1;
    }
}

let animationFrameId001;
```

```
        function gameLoop()
        {
            animate();

            animationFrameId001 =
requestAnimationFrame(gameLoop);
        }

        gameLoop();
    }

    createBallWithMotion();

}());
```

```
javascript:(
/* Bouncing Ball - Linear Motion - Starts at an
Angle - No Trigonometry Required */
function()
{
    function createBallWithMotion()
    {
        let ball = document.createElement('div');
        ball.id = 'ball';
        ball.style.position = "absolute";
        ball.style.width = "50px";
        ball.style.height = "50px";
        ball.style.borderRadius = "50%";
        ball.style.margin = "0px";
        ball.style.overflow = "hidden";
        ball.style.backgroundColor = "aqua";
        document.body.append(ball);

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

```
let dx = 5;
let dy = 5;

function animate()
{
    x += dx;
    y += dy;

    if (x < 0 || x > window.innerWidth -
ball.clientWidth)
    {
        dx = -dx;
    }

    if (y < 0 || y > window.innerHeight -
ball.clientHeight)
    {
        dy = -dy;
    }
```

```
        ball.style.left = x + 'px';
        ball.style.top = y + 'px';
    }

    let animationFrameId001;

    function gameLoop()
    {
        animate();

        animationFrameId001 =
requestAnimationFrame(gameLoop);
    }

    gameLoop();
  }

  createBallWithMotion();
}());
```

```
javascript:(
/* Bouncing Ball - Linear Motion - Starts at Angle
- Trigonometry Angle Calculations */
function()
{
    function createBouncingBall()
    {
        let ball = document.createElement('div');
        ball.id = 'ball';
        ball.style.position = "absolute";
        ball.style.width = "50px";
        ball.style.height = "50px";
        ball.style.borderRadius = "50%";
        ball.style.margin = "0px";
        ball.style.overflow = "hidden";
        ball.style.backgroundColor = "aqua";
        document.body.append(ball);

        /* initial angle in degrees */
        let angle = 45;
```

```
let speed = 2;

/* convert degrees to radians for Math.sin
and Math.cos */
let radians = angle * (Math.PI / 180);

let xSpeed = Math.cos(radians) * speed;
let ySpeed = Math.sin(radians) * speed;

let positionX = 0;
let positionY = 0;

function animate()
{
    positionX += xSpeed;
    positionY += ySpeed;

    ball.style.left = positionX + 'px';
    ball.style.top = positionY + 'px';
```

```
        /* bounce off the walls */
        if (positionX >= (window.innerWidth - 50)
|| positionX <= 0)
        {
            xSpeed *= -1;
        }

        if (positionY >= (window.innerHeight - 50)
|| positionY <= 0)
        {
            ySpeed *= -1;
        }
    }

    let animationFrameId001;

    function gameLoop()
    {
        animate();
```

```
        animationFrameId001 =
requestAnimationFrame(gameLoop);
      }

      gameLoop();
   }

   createBouncingBall();

}());
```

```
javascript:(
/* Bouncing Circles - Linear Motion - Starts at an
Angle - No Trigonometry Required  */
function()
{
    let circles = [];

    let numberOfCircles = 10;

    function getRandomPosition()
    {
        let x = Math.floor(Math.random() *
window.innerWidth);

        let y = Math.floor(Math.random() *
window.innerHeight);

        return {
            x: x,
            y: y
```

```
    };
}

for (let x = 0; x < numberOfCircles; x++)
{
    let circle = document.createElement('div');
    circle.style.position = 'absolute';
    circle.style.width = '20px';
    circle.style.height = '20px';

    circle.style.backgroundColor = 'none';
    circle.style.borderStyle = 'solid';
    circle.style.borderWidth = '1px';
    circle.style.borderRadius = '50%';
    circle.style.pointerEvents = 'none';

    let position = getRandomPosition();
    circle.style.left = position.x + 'px';
    circle.style.top = position.y + 'px';
```

```
        circle.dx = Math.random() * 4 - 2;
        circle.dy = Math.random() * 4 - 2;

        document.body.append(circle);

        circles.push(circle);
    }

    function animate()
    {
        for (let x = 0; x < circles.length; x++)
        {
            let circle = circles[x];

            circle.style.left = (parseInt(circle.style.left)
+ circle.dx) + 'px';

            circle.style.top =
(parseInt(circle.style.top) + circle.dy) + 'px';
```

```
            if (parseInt(circle.style.left) <= 0 ||
parseInt(circle.style.left) >= window.innerWidth -
20)
            {
                circle.dx *= -1;
            }

            if (parseInt(circle.style.top) <= 0 ||
parseInt(circle.style.top) >= window.innerHeight
- 20)
            {
                circle.dy *= -1;
            }
        }
    }

    let animationFrameId001;

    function gameLoop()
    {
```

```
        animate();

        animationFrameId001 =
requestAnimationFrame(gameLoop);
    }

    gameLoop();

}());
```

```
javascript:(
/* Airplane - Projectile - Simulated Drop of
projectile from airplane where user clicks */
function()
{
    let airplane = document.createElement('div');
    airplane.style.position = 'absolute';
    airplane.style.left = '-50px';
    airplane.style.top = '50px';
    airplane.style.width = '50px';
    airplane.style.height = '20px';
    airplane.style.backgroundColor = 'blue';
    airplane.style.transition = 'all 2s linear';
    document.body.append(airplane);

    /*----*/

    let projectile = document.createElement('div');
    projectile.style.position = 'absolute';
    projectile.style.width = '15px';
```

```
    projectile.style.height = '15px';
    projectile.style.borderRadius = "50%";
    projectile.style.backgroundColor = 'aqua';
    projectile.style.transition = 'all 1s linear';
    projectile.style.display = 'none';
    document.body.append(projectile);

    let flying = false;

    document.onclick = function(event)
    {
        if (!flying)
        {
            flying = true;

            airplane.style.transform = 'translate(' +
window.innerWidth + 'px, 0)';

            setTimeout(function()
            {
```

```
            let x = event.clientX;
            let y = event.clientY;

            projectile.style.left = x + 'px';
            projectile.style.top = '25px';

            projectile.style.display = 'block';

            /* projectile path */
            projectile.style.transform = 'translateY('
+ (y - 25) + 'px)';
          }, 1000);
      }
   };

}());
```

```
/*
When a person left clicks their mouse on the
screen an airplane will fly to the right and
simulate dropping a projectile below, which will
remain on the screen.
*/
```

```
javascript:(
/* Airplane - Projectile - Realistic - Projectile
drops from bottom middle of airplane  */
function()
{
    let airplane = document.createElement('div');
    airplane.id = "airplane";
    airplane.style.position = 'absolute';
    airplane.style.left = '-50px';
    airplane.style.top = '50px';
    airplane.style.width = '50px';
    airplane.style.height = '20px';
    airplane.style.backgroundColor = 'blue';
    airplane.style.transition = 'all 2s linear';
    document.body.append(airplane);

    /*----*/

    let projectile = document.createElement('div');
    projectile.id = "projectile";
```

```
projectile.style.position = 'absolute';
projectile.style.width = '15px';
projectile.style.height = '15px';
projectile.style.borderRadius = "50%";
projectile.style.backgroundColor = 'aqua';
/* initially hide the object */
projectile.style.display = 'none';
document.body.append(projectile);

let flying = false;

document.onclick = function(event)
{
    if (!flying)
    {
        flying = true;

        let x = event.clientX + 15;
        let y = event.clientY;
```

```
        /* calculate the adjustment needed for the
airplane's center */
        let airplaneCenterAdjustment =
airplane.offsetWidth / 2;

        /* set the plane's starting position to the
left of the viewport */
        airplane.style.left = -
airplaneCenterAdjustment + 'px';

        /* set the plane's final position (where it
was clicked) */
        airplane.style.transform = 'translate(' + (x
- airplaneCenterAdjustment) + 'px, 0)';

        /* set the initial position of the projectile
at the bottom middle of the airplane */
        let airplaneRect =
airplane.getBoundingClientRect();
```

```
        let projectileLeft = x -
airplaneCenterAdjustment;

        /* place the projectile at the bottom of the
airplane */
        let projectileTop = airplaneRect.bottom;

        projectile.style.left = projectileLeft + 'px';
        projectile.style.top = projectileTop + 'px';

        /* calculate the distance to fall (from top
to the user's click Y-coordinate) */
        let distanceToFall = y - projectileTop;

        /* set a timeout to start the object's
descent and reveal it */
        setTimeout(function()
        {
         /* display the object to make it visible */
           projectile.style.display = 'block';
```

```
/* set the initial time and speed */
let startTime = null;

/* falling speed: higher values make it
fall faster */
let speed = 200;

function animateProjectile(timestamp)
{
    if (!startTime)
    {
        startTime = timestamp;
    }

    /* calculate the elapsed time */
    let elapsedTime = timestamp -
startTime;
```

```
                /* calculate the new top position
based on the elapsed time and speed */
                let newY = (elapsedTime / 1000) *
speed;

                if (newY < distanceToFall)
                {
                    /* continue the animation */
        requestAnimationFrame(animateProjectile);
                }
                else
                {
                    /* finish the animation when it
reaches the target Y coordinate */
                    newY = distanceToFall;
                }

                /* apply the new top position */
                projectile.style.transform =
'translateY(' + newY + 'px)';
```

```
            }

            /* start the animation */
        requestAnimationFrame(animateProjectile);
            }, 2000);

            /* adjust the timeout delay based on
the airplane's transition duration */
        }
    };

}());
```

```
javascript:(
/* Mouse Arrow Effect - Circles */
function()
{
    function changeMouseArrow()
    {
        const theEffect =
document.createElement('style');

        theEffect.innerText = `
     * {
            cursor: url('data:image/svg+xml;utf8,<svg
xmlns="http://www.w3.org/2000/svg" width="32"
height="32" viewBox="0 0 24 24"><circle
cx="12" cy="12" r="10" stroke="white" stroke-
width="2" fill="transparent"/><circle cx="12"
cy="12" r="6" stroke="white" stroke-width="2"
fill="transparent"/><circle cx="12" cy="12" r="2"
fill="white"/></svg>'), auto !important;
        }`;
```

```
        document.head.append(theEffect);
    }

    changeMouseArrow();

}());
```

```
javascript:(
/* Animate Words Typing Effect */
function()
{
    function animateText()
    {
        let text = "Hi Everyone";
        let x = 0;
        let animationSpeed = 400;

        let textDiv =
document.createElement("div");
        textDiv.style.position = "fixed";
        textDiv.style.top = "50%";
        textDiv.style.left = "50%";
        textDiv.style.transform = "translate(-50%, -
50%)";
        textDiv.style.fontSize = "25px";
        document.body.append(textDiv);
```

```
        let animation = setInterval(function()
        {
            if (x < text.length)
            {
                textDiv.textContent = text.substring(0,
x + 1);

                x++;
            }
            else
            {
                clearInterval(animation);
            }
        }, animationSpeed);
    }

    animateText();
}());
/* displays a text of "Hi Everyone" with an
animation. Each letter appears one by one. */
```

```
javascript:(
/* Detect which Web Browser is being used */
function()
{
    function detectWebBrowser()
    {
        if
(navigator.userAgent.toLowerCase().indexOf('fir
efox') !== -1)
        {
            return 'You are using Firefox';
        }

        if
(navigator.userAgent.toLowerCase().indexOf('ch
rome') !== -1)
        {
            return 'You are using Chrome';
        }
```

```
        if
(navigator.userAgent.toLowerCase().indexOf('sa
fari') !== -1)
        {
            return 'You are using Safari';
        }

        // if none of the above, display a generic
message
        return 'No known browser detected';
    }

    console.log(detectWebBrowser());

    alert(detectWebBrowser());

}());
```

```
javascript:(
/* Pythagorean Theorem */
function()
{
    /* create a function to calculate the
hypotenuse length */
    function calculateHypotenuse(a, b)
    {
        return Math.sqrt(a * a + b * b);
    }

    /* create a function to display the result */
    function displayResult(a, b, c)
    {
        alert(
            "Side A: " + a + "\n" +
            "Side B: " + b + "\n" +
            "Hypotenuse C: " + c.toFixed(2)
        );
    }
```

```
    /* prompt the user for the lengths of side A
and side B */
    const sideA = parseFloat(prompt("Enter the
length of side A:"));

    const sideB = parseFloat(prompt("Enter the
length of side B:"));

    /* check if the input is valid */
    if (isNaN(sideA) || isNaN(sideB))
    {
        alert("Enter only valid numbers");
    }
    else
    {
        /* calculate hypotenuse with the
Pythagorean Theorem */
        const hypotenuse =
calculateHypotenuse(sideA, sideB);
```

```
        /* display the result */
        displayResult(sideA, sideB, hypotenuse);
    }

}());
```

```
javascript:(
/* Calculate Factorial */
function()
{
  /* create a function to calculate the factorial */
   function calculateFactorial(whichNumber)
   {
      if (whichNumber < 0)
      {
         return "The factorial is not defined for
negative numbers";
      }
      else if (whichNumber === 0)
      {
         return 1;
      }
      else
      {
         let factorial = 1;
```

```
        for (let x = 1; x <= whichNumber; x++)
        {
            factorial *= x;
        }

        return factorial;
    }
}

/* prompt the user for a number */
const number = parseInt(prompt("Enter a non-
negative integer to calculate its factorial:"));

/* check if the input is valid */
if (isNaN(number) || number < 0)
{
    alert("Enter a non-negative integer");
}
else
{
```

```
        /* calculate the factorial */
        const factorialResult =
calculateFactorial(number);

        /* display the result */
        alert("Factorial of " + number + " is: " +
factorialResult);
    }

}());
```

```
javascript:(
/* Calculate Circle Circumference and Area */
function()
{
    function calculateCircleProperties()
    {
        let radius = parseFloat(prompt("Enter the
radius of the circle:"));

        if (!isNaN(radius) && radius >= 0)
        {
            let area = Math.PI * Math.pow(radius, 2);

            let circumference = 2 * Math.PI * radius;

            alert(
                "Radius: " + radius.toFixed(2) + "\n" +
                "Area: " + area.toFixed(2) + "\n" +
                "Circumference: " +
circumference.toFixed(2)
```

```
            );
        }
        else
        {
            alert("Enter a valid positive number for
the radius.");
        }
    }

    calculateCircleProperties();

}());
```

```
javascript:(
/* Button Beep Sound using Oscillator */
function()
{
    /* create a button element */
    let button = document.createElement('button');
    button.textContent = 'Click me for a beep';

    /* create a new AudioContext */
    let AudioContext = window.AudioContext ||
window.webkitAudioContext;

    let audioContext = new AudioContext();

    /* add click event to play the beep sound */
    button.addEventListener('click', function()
    {
      /* create oscillator node for the beep sound */
        let oscillator =
audioContext.createOscillator();
```

```
        oscillator.type = 'sine';

        /* frequency of the beep sound */
        oscillator.frequency.value = 1000;

      oscillator.connect(audioContext.destination);

        oscillator.start();

        setTimeout(function()
        {
            oscillator.stop();
        }, 100);
    });

    /* append the button to the body */
    document.body.append(button);

}());
```

```
javascript:(
/* Draggable Square */
function()
{
    function createDraggableSquare()
    {
        let square = document.createElement('div');
        square.style.position = 'absolute';
        square.style.left = '100px';
        square.style.top = '100px';
        square.style.width = '50px';
        square.style.height = '50px';
        square.style.backgroundColor = 'aqua';
        square.style.cursor = 'move';

        /* initialize variables for tracking the drag
state */
        let isDragging = false;
        let offsetX;
        let offsetY;
```

```
        /* function to handle mouse down event and
start dragging */
        square.addEventListener('mousedown',
function(event)
        {
            isDragging = true;

            offsetX = event.clientX -
square.getBoundingClientRect().left;

            offsetY = event.clientY -
square.getBoundingClientRect().top;
        });

        /* function to handle mouse up event and
stop dragging */
        window.addEventListener('mouseup',
function()
        {
```

```
        isDragging = false;
    });

    /* function to handle mouse move event and
update square position */
    window.addEventListener('mousemove',
function(event)
    {
        if (isDragging)
        {
            let x = event.clientX - offsetX;
            let y = event.clientY - offsetY;

            /* make sure the square stays within
the viewport */
            x = Math.min(Math.max(x, 0),
window.innerWidth - 50);

            y = Math.min(Math.max(y, 0),
window.innerHeight - 50);
```

```
            square.style.left = x + 'px';
            square.style.top = y + 'px';
        }
    });

    document.body.append(square);
}

createDraggableSquare();

}());
```

```
javascript:(
/* Drag any element on any webpage */
function()
{
    function makePageElementsDraggable()
    {
        let dragItem = null;
        let offsetX;
        let offsetY;

    document.addEventListener('mousedown',
function(e)
        {
            dragItem = e.target;

            offsetX = e.clientX -
dragItem.getBoundingClientRect().left;

            offsetY = e.clientY -
dragItem.getBoundingClientRect().top;
```

```
            dragItem.style.position = 'absolute';
document.body.append(dragItem);
        });

    document.addEventListener('mousemove',
function(e)
        {
            if (dragItem)
            {
                dragItem.style.left = e.clientX - offsetX
+ 'px';

                dragItem.style.top = e.clientY - offsetY
+ 'px';
            }
        });

        document.addEventListener('mouseup',
function()
```

```
        {
            dragItem = null;
        });
    }

    makePageElementsDraggable();

}());
```

```
javascript:(
/* Newton's First Law of Motion - The Law of
Inertia.*/
function()
{
    function createMovingSquare()
    {
        let box = document.createElement("div");
        box.style.position = "absolute";
        box.style.top = "50px";
        box.style.left = "50px";
        box.style.width = "100px";
        box.style.height = "100px";
        box.style.backgroundColor = "aqua";
        document.body.append(box);

        /* starting velocity */
        let velocity = 1;

        function animate()
```

```
    {
        /* move the box horizontally */
        box.style.left = parseInt(box.style.left) +
velocity + "px";

        /* check if box has reached right edge */
        if (parseInt(box.style.left) +
parseInt(box.style.width) >= window.innerWidth)
        {
            /* reverse direction */
            velocity = -1;
        }

        /* check if box has reached left edge */
        if (parseInt(box.style.left) <= 0)
        {
            /* reverse direction */
            velocity = 1;
        }
    }
```

```
        let animationFrameId001;

        function gameLoop()
        {
            animate();

            animationFrameId001 =
requestAnimationFrame(gameLoop);
        }

        gameLoop();
    }

    createMovingSquare();

}());

/*
```

Newton's First Law of Motion - an object will keep going in its current state of motion unless an external force acts on it.

In this script, the object continues moving freely until it encounters the screen's edges.

When it hits a wall, it changes direction by bouncing off with an equal and opposite force to its original motion.
*/

```
javascript:(
/* Calculate Distance of Galactus to Earth */
function()
{
    function calculateGalactusDistanceToEarth()
    {
        /* starting distance in miles of Galactus to
Earth */
        let distance = 1000000;

        /* speed in mph - how fast Galactus is
traveling */
        let speed = 10000;

        /* update interval - how often to update us
on the distance */
        let updateTime = 1 * 1000;

        function updateDistance()
        {
```

```
        /* update distance */
        distance -= speed / 60;

        if (distance <= 0)
        {
            document.getElementById("galactus-
update").innerHTML = "Galactus has arrived!";

            clearInterval(interval);
        }
        else
        {
            document.getElementById("galactus-
update").innerHTML = "Galactus is " +
distance.toFixed(2) + " miles away.";
        }
    }

    /* create a div element for the updates */
```

```
    let updateDiv =
document.createElement("div");
    updateDiv.id = "galactus-update";
    updateDiv.style.position = "fixed";
    updateDiv.style.right = "10px";
    updateDiv.style.bottom = "10px";
    updateDiv.style.padding = "10px";
    updateDiv.style.backgroundColor = "rgba(0,
0, 0, 0.7)";
    updateDiv.style.color = "white";
    updateDiv.style.fontFamily = "Arial";
    updateDiv.style.fontSize = "16px";
    updateDiv.style.zIndex = "10000";
    document.body.append(updateDiv);

    /* update the distance initially */
    updateDistance();

    /* set up a timer to update the distance
every specified seconds */
```

```
        let interval = setInterval(updateDistance,
updateTime);
    }

    calculateGalactusDistanceToEarth();

}());
```

Galactus is 945833.33 miles away.

```
javascript:(
/* Calculate Distance of Galactus to Earth and
How Long Until Arrival */
function()
{
   function
calculateGalactusDistanceAndTimeToEarth()
   {
      /* starting distance in miles of Galactus to
Earth */
      let distance = 1000000;

      /* speed in mph - how fast Galactus is
traveling */
      let speed = 10000;

      /* update interval - how often to update us
on the distance */
      let updateTime = 1 * 1000;
```

```
function updateDistance()
{
    /* update distance */
    distance -= speed / 60;

    if (distance <= 0)
    {
        document.getElementById("galactus-
update").innerHTML = "Galactus has arrived!";
clearInterval(interval);
    }
    else
    {
        /* calculate time in hours */
        let timeInHours = distance / speed;

        /* convert time to hours and minutes */
        let hours = Math.floor(timeInHours);
```

```
            let minutes = Math.floor((timeInHours -
hours) * 60);

            document.getElementById("galactus-
update").innerHTML =
"Galactus is " + distance.toFixed(2) +
" miles away.<br>Arrival in " +
hours + " hours and " + minutes + " minutes.";
        }
    }

    /* create a div element for the updates */
    let updateDiv =
document.createElement("div");
    updateDiv.id = "galactus-update";
    updateDiv.style.position = "fixed";
    updateDiv.style.right = "10px";
    updateDiv.style.bottom = "10px";
    updateDiv.style.padding = "10px";
```

```
        updateDiv.style.backgroundColor = "rgba(0,
0, 0, 0.7)";
        updateDiv.style.color = "white";
        updateDiv.style.fontFamily = "Arial";
        updateDiv.style.fontSize = "16px";
        updateDiv.style.zIndex = "10000";
        document.body.append(updateDiv);

        /* update the distance initially */
        updateDistance();

        /* set up a timer to update the distance
every specified seconds */
        let interval = setInterval(updateDistance,
updateTime);
    }

   calculateGalactusDistanceAndTimeToEarth();

}());
```

Galactus is 989500.00 miles away.
Arrival in 98 hours and 57 minutes.

```
javascript:(
/* Square Expands with Each Click */
function()
{
    function createExpandingSquare()
    {
        /* create a square element */
        const square =
document.createElement('div');
        square.id = "square";
        square.style.position = "absolute";
        square.style.left = 100 + "px";
        square.style.top = 50 + "px";
        square.style.width = '100px';
        square.style.height = '100px';
        square.style.backgroundColor = 'aqua';
        square.style.transition = 'width 0.3s, height
0.3s';
        square.style.cursor = 'pointer';
```

```
        /* starting width and height */
        let width = 100;
        let height = 100;

        /* add click event listener to grow the
square on each click */
        square.onclick = function()
        {
            width += 20;
            height += 20;

            square.style.width = width + 'px';
            square.style.height = height + 'px';
        };

        /* add the square to the page */
        document.body.append(square);
    }
    createExpandingSquare();
}());
```

```
javascript:(
/* Square Expands/Contracts by Click */
function()
{
    function createExpandingContractingSquare()
    {
        /* create a square element */
        const square =
document.createElement('div');
        square.id = 'square';
        square.style.position = 'absolute';
        square.style.left = '100px';
        square.style.top = '50px';
        square.style.width = '100px';
        square.style.height = '100px';
        square.style.backgroundColor = 'aqua';
        square.style.transition = 'width 0.3s, height
0.3s';
        square.style.cursor = 'pointer';
```

```
/* starting width and height */
let width = 100;
let height = 100;

/* initialize a variable to keep track of the
direction (expanding or contracting) */
let expanding = true;

/* onclick adjust the square's size */
square.onclick = function()
{
    if (expanding)
    {
        width += 20;
        height += 20;

        if (width >= 300)
        {
            expanding = false;
        }
```

```
        }
        else
        {
            width -= 20;
            height -= 20;

            if (width <= 100)
            {
                expanding = true;
            }
        }
        square.style.width = width + 'px';
        square.style.height = height + 'px';
    };

    /* add the square to the page */
    document.body.append(square);
  }
  createExpandingContractingSquare();
}());
```

```
javascript:(
/* Square Expands/Contracts by Timer */
function()
{
    function createExpandingContractingSquare()
    {
        /* create a square element */
        const square =
document.createElement('div');
        square.id = 'square';
        square.style.position = 'absolute';
        square.style.left = '100px';
        square.style.top = '50px';
        square.style.width = '100px';
        square.style.height = '100px';
        square.style.backgroundColor = 'aqua';
        square.style.transition = 'width 0.3s, height
0.3s';
        square.style.cursor = 'pointer';
```

```
/* stop the timer when the square is clicked */
square.onclick = function()
{
    clearInterval(timerInterval);
};

/* add the square to the page */
document.body.append(square);

/*----*/

/* initialize width and height */
let width = 100;
let height = 100;

/* initialize a variable to keep track of the
direction (expanding or contracting) */
let expanding = true;

function updateSquareSize()
```

```
{
    if (expanding)
    {
        width += 20;
        height += 20;

        if (width >= 300)
        {
            expanding = false;
        }
    }
    else
    {
        width -= 20;
        height -= 20;

        if (width <= 100)
        {
            expanding = true;
        }
```

```
        }

        square.style.width = width + 'px';
        square.style.height = height + 'px';
    }

    /* set up a timer to update the square's size
every 1 second */
    const timerInterval =
setInterval(updateSquareSize, 1000);
  }

  createExpandingContractingSquare();

}());
```

```
javascript:(
/* Square Moves in Square Pattern by Timer */
function()
{
    function createSquareMoveInSquarePattern()
    {
        /* create a square element */
        let square = document.createElement('div');
        square.style.position = 'absolute';
        square.style.left = '20px';
        square.style.top = '20px';
        square.style.width = '50px';
        square.style.height = '50px';
        square.style.backgroundColor = 'aqua';
        square.style.transition = 'transform 2s';

        /* initialize the starting position */
        let x = 20;
        let y = 20;
```

```
/* 0 is right, 1 is down, 2 is left, 3 is up */
let direction = 0;

/* add the square to the page */
document.body.append(square);

/* function to move the square in a square
pattern */
function moveSquare()
{
    if (direction === 0)
    {
        x += 100;
    }
    else if (direction === 1)
    {
        y += 100;
    }
    else if (direction === 2)
    {
```

```
            x -= 100;
        }
        else if (direction === 3)
        {
            y -= 100;
        }

        square.style.transform = 'translate(' + x +
'px, ' + y + 'px)';

        /* update direction */
        direction = (direction + 1) % 4;
    }

    /* start moving the square */
    moveSquare();

    /* repeat square pattern continuously */
    setInterval(moveSquare, 2000);
}
```

```
    createSquareMoveInSquarePattern();

}());
```

```
javascript:(
/* Special Efx - when clicked div turns to
particles */
function()
{
    function makeClickableSpecialEfx()
    {
        let specialEfxBox =
document.createElement('div');
        specialEfxBox.style.position = 'absolute';
        specialEfxBox.style.left = "100px";
        specialEfxBox.style.top = "50px";
        specialEfxBox.style.width = '50px';
        specialEfxBox.style.height = '50px';
        specialEfxBox.style.backgroundColor =
'blue';
        specialEfxBox.style.cursor = 'pointer';

        specialEfxBox.onclick = function()
        {
```

```
let particles = [];

let amount = 500;

for (let x = 0; x < amount; x++)
{
    let particle =
document.createElement('div');
    particle.style.position = 'absolute';
    particle.style.width = '5px';
    particle.style.height = '5px';
    particle.style.backgroundColor =
'aqua';
    particle.style.animation = 'explode 1s
ease-in-out';

    /* random X coordinate within a range */
    let randomX = Math.random() * 200 - 100;

    /* random Y coordinate within a range */
```

```
            let randomY = Math.random() * 200 - 100;

            particle.style.left =
specialEfxBox.getBoundingClientRect().left +
randomX + 'px';

            particle.style.top =
specialEfxBox.getBoundingClientRect().top +
randomY + 'px';

            particles.push(particle);

            document.body.append(particle);
        }

        /* hide the original div */
        specialEfxBox.style.display = 'none';

        setTimeout(function()
        {
```

```
            for (let i = 0; i < particles.length; i++)
            {
                /* remove particles after the
animation */
                particles[i].remove();
            }
        }, 1000);
    };

    document.body.append(specialEfxBox);

    /*----*/

    let style001 =
document.createElement('style');

    style001.innerHTML = `
      @keyframes explode {
       0% {
           opacity: 1;
```

```
            transform: scale(1);
        }
        100% {
            opacity: 0;
            transform: scale(2);
        }
    }`;

    document.head.append(style001);
}

makeClickableSpecialEfx();

}());
```

```
javascript:(
/* Special Efx - Random Color Squares */
function()
{
    function makeClickableSpecialEfx()
    {
        let specialEfxBox =
document.createElement('div');
        specialEfxBox.style.position = 'absolute';
        specialEfxBox.style.left = '100px';
        specialEfxBox.style.top = '50px';
        specialEfxBox.style.width = '50px';
        specialEfxBox.style.height = '50px';
        specialEfxBox.style.backgroundColor =
'blue';
        specialEfxBox.style.cursor = 'pointer';

        specialEfxBox.onclick = function()
        {
            let particles = [];
```

```
let amount = 500;

for (let x = 0; x < amount; x++)
{
    let particle =
document.createElement('div');
    particle.style.position = 'absolute';
    particle.style.width = '10px';
    particle.style.height = '10px';
    particle.style.backgroundColor =
getRandomColor();
    particle.style.animation = 'explode 1s
ease-in-out';

   /* random X coordinate within a range */
    let randomX = Math.random() * 200 -
100;

   /* random Y coordinate within a range */
```

```
            let randomY = Math.random() * 200 -
100;

            particle.style.left =
specialEfxBox.getBoundingClientRect().left +
randomX + 'px';

            particle.style.top =
specialEfxBox.getBoundingClientRect().top +
randomY + 'px';

            particles.push(particle);

            document.body.append(particle);
        }

        /* hide the original div */
        specialEfxBox.style.display = 'none';

        setTimeout(function()
```

```
        {
            for (let i = 0; i < particles.length; i++)
            {
                /* remove particles after animation */
                particles[i].remove();
            }
        }, 500);
    };

    document.body.append(specialEfxBox);

    /*----*/

    let style001 =
document.createElement('style');

    style001.innerHTML = `
        @keyframes explode {
        0% {
            opacity: 1;
```

```
            transform: scale(1);
        }
        100% {
            opacity: 0;
            transform: scale(5);
        }
    }`;

    document.head.append(style001);
}

function getRandomColor()
{
    let r = Math.floor(Math.random() * 255);
    let g = Math.floor(Math.random() * 255);
    let b = Math.floor(Math.random() * 255);

    return 'rgb('+r+', '+g+', '+b+')';
}
```

```
    makeClickableSpecialEfx();

}());
```

```
javascript:(
/* Special Efx - Circles Random Color and Size */
function()
{
    function makeClickableSpecialEfx()
    {
        let specialEfxBox =
document.createElement('div');
        specialEfxBox.style.position = 'absolute';
        specialEfxBox.style.left = '100px';
        specialEfxBox.style.top = '50px';
        specialEfxBox.style.width = '100px';
        specialEfxBox.style.height = '100px';
        specialEfxBox.style.backgroundColor =
'blue';
        specialEfxBox.style.cursor = 'pointer';

        specialEfxBox.onclick = function()
        {
            let particles = [];
```

```
        let amount = 500;

        for (let x = 0; x < amount; x++)
        {
            let particle =
document.createElement('div');
            particle.style.position = 'absolute';

            /* get a random size for the circle */
            let size = getRandomSize();

            particle.style.width = size + 'px';
            particle.style.height = size + 'px';
            /* make it a circle */
            particle.style.borderRadius = '50%';
            particle.style.backgroundColor =
getRandomColor();
            particle.style.animation = 'explode 1s
ease-in-out';
```

```
            /* random X coordinate within a range */
            let randomX = Math.random() * 200 -
100;

            /* random Y coordinate within a range */
            let randomY = Math.random() * 200 -
100;
             particle.style.left =
specialEfxBox.getBoundingClientRect().left +
randomX + 'px';

             particle.style.top =
specialEfxBox.getBoundingClientRect().top +
randomY + 'px';

             particles.push(particle);

             document.body.append(particle);
        }
```

```
        /* hide the original div */
        specialEfxBox.style.display = 'none';

        setTimeout(function()
        {
            for (let i = 0; i < particles.length; i++)
            {
                /* remove particles after the
animation */
                particles[i].remove();
            }
        }, 500);
    };

    document.body.append(specialEfxBox);

    /*----*/
```

```
        let style001 =
document.createElement('style');

        style001.innerHTML = `
            @keyframes explode {
            0% {
                opacity: 1;
                transform: scale(1);
            }
            100% {
                opacity: 0;
                transform: scale(5);
            }
        }`;

        document.head.append(style001);
    }

    function getRandomColor()
    {
```

```
        let r = Math.floor(Math.random() * 255);
        let g = Math.floor(Math.random() * 255);
        let b = Math.floor(Math.random() * 255);

        return 'rgb('+r+', '+g+', '+b+')';
    }

    /* function to get a random size for circle */
    function getRandomSize()
    {
        return Math.floor(Math.random() * 20) + 5;
    }

    makeClickableSpecialEfx();

}());
```

```
javascript:(
/* Special Efx - Circles in a Circle */
function()
{
    function makeClickableSpecialEfx()
    {
        let specialEfxBox =
document.createElement('div');
        specialEfxBox.style.position = 'absolute';
        specialEfxBox.style.left = '50%';
        specialEfxBox.style.top = '50%';
        specialEfxBox.style.transform = 'translate(-
50%, -50%)';
        specialEfxBox.style.width = '50px';
        specialEfxBox.style.height = '50px';
        specialEfxBox.style.backgroundColor =
'blue';
        /* make it a circle */
        specialEfxBox.style.borderRadius = '50%';
        specialEfxBox.style.cursor = 'pointer';
```

```
specialEfxBox.onclick = function()
{
    let particles = [];

    let amount = 500;

    for (let angle = 0; angle < 360; angle +=
360 / amount)
    {
        let particle =
document.createElement('div');
        particle.style.position = 'absolute';
        particle.style.width = '50px';
        particle.style.height = '50px';
        /* make it a circle */
        particle.style.borderRadius = '50%';
        particle.style.backgroundColor =
getRandomColor();
```

```
            particle.style.animation = 'explode 1s
ease-in-out';

            let radius = 200;

            /*----*/

            let x = Math.cos((angle * Math.PI) / 180)
* radius;

            let y = Math.sin((angle * Math.PI) / 180) *
radius;

            /*----*/

            particle.style.left = 'calc(50% + ' + x +
'px)';

            particle.style.top = 'calc(50% + ' + y +
'px)';
```

```
        /*----*/

        particles.push(particle);

        document.body.append(particle);
    }

    /* hide the original div */
    specialEfxBox.style.display = 'none';

    setTimeout(function()
    {
        for (let i = 0; i < particles.length; i++)
        {
            /* remove particles after animation */
            particles[i].remove();
        }
    }, 500);
};
```

```
document.body.append(specialEfxBox);

/*----*/

let style001 =
document.createElement('style');

style001.innerHTML = `
    @keyframes explode {
    0% {
        opacity: 1;
        transform: scale(1);
    }
    100% {
        opacity: 0;
        transform: scale(5);
    }
}`;
```

```
        document.head.append(style001);
    }

    function getRandomColor()
    {
        let r = Math.floor(Math.random() * 255);
        let g = Math.floor(Math.random() * 255);
        let b = Math.floor(Math.random() * 255);

        return 'rgb('+r+', '+g+', '+b+')';
    }

    makeClickableSpecialEfx();

}());
```

Christopher Topalian

```
javascript:(
/* Special Efx - Circle Expands */
function()
{
    function makeSpecialEfx()
    {
        let particles = [];

        let amount = 1000;

        for (let angle = 0; angle < 360; angle += 360 /
amount)
        {
            let particle =
document.createElement('div');
            particle.style.position = 'absolute';
            particle.style.width = '50px';
            particle.style.height = '50px';
            particle.style.borderRadius = "50%";
            particle.style.backgroundColor = 'aqua';
```

```
        particle.style.animation = 'explode 1s
ease-in-out';

        /* start particles from the center */
        let radius = 0;

        /*----*/

        let x = Math.cos((angle * Math.PI) / 180) *
radius;

        let y = Math.sin((angle * Math.PI) / 180) *
radius;

        /*----*/

        particle.style.left = 'calc(50% + ' + x +
'px)';
```

```
            particle.style.top = 'calc(50% + ' + y +
'px)';

            /*----*/

            particles.push(particle);

            document.body.append(particle);
        }

        setTimeout(function()
        {
            for (let i = 0; i < particles.length; i++)
            {
               /* remove particles after the animation */
                particles[i].remove();
            }
        }, 1000);
    }
```

```
/*-----*/

let style001 =
document.createElement('style');

style001.innerHTML = `
    @keyframes explode {
    0% {
        opacity: 1;
        transform: scale(1);
    }
    100% {
        opacity: 0;
        transform: scale(5);
    }
}`;

document.head.append(style001);

makeSpecialEfx();
```

```
}());
```

```
/*
The Circle expands with a smooth animation.
*/
```

```
javascript:(
/* Special Efx - Circle Expands to Concentric
Circles */
function()
{
    function makeSpecialEfx()
    {
        let circles = [];

        /* number of concentric circles */
        let numCircles = 10;

        /* initial size of the innermost circle */
        let circleSize = 50;

        for (let i = 0; i < numCircles; i++)
        {
            let circle =
document.createElement('div');
            circle.style.position = 'absolute';
```

```
circle.style.width = circleSize + 'px';
circle.style.height = circleSize + 'px';
circle.style.borderRadius = '50%';
circle.style.backgroundColor = 'aqua';
circle.style.animation = 'expandCircle 1s
ease-in-out';

/*----*/

/* calculate position to center the circles */
let leftPosition = (window.innerWidth -
circleSize) / 2;

let topPosition = (window.innerHeight -
circleSize) / 2;

/*----*/

circle.style.left = leftPosition + 'px';
circle.style.top = topPosition + 'px';
```

```
        /*----*/

        /* increase the circle size for the next
concentric circle */
        circleSize += 50;

        circles.push(circle);

        document.body.append(circle);
    }

    setTimeout(function()
    {
        for (let i = 0; i < circles.length; i++)
        {
            /* remove circles after animation */
            circles[i].remove();
        }
    }, 1000);
```

```
}

/*----*/

let style001 =
document.createElement('style');

style001.innerHTML = `
    @keyframes expandCircle {
    0% {
        opacity: 1;
        transform: scale(1);
    }
    100% {
        opacity: 0;
        transform: scale(1.5);
    }
}`;

document.head.append(style001);
```

```
    makeSpecialEfx();

}());
```

```
javascript:(
/* Special Efx - Array of Objects - Random
Choice */
function()
{
    function generateSpecialEffects()
    {
        let specialEfx =
        [
            {
                name: 'Effect 1',
                color: 'aqua',
                size: 50,
            },

            {
                name: 'Effect 2',
                color: 'yellow',
                size: 200,
            },
```

```
        {
            name: 'Effect 3',
            color: 'blue',
            size: 400,
        },
    ];

    function createRandomEfx()
    {
        let whichArray = specialEfx;

        let randomIndex =
Math.floor(Math.random() * whichArray.length);

        let effectChosen =
whichArray[randomIndex];

        /*----*/
```

```
        let particle =
document.createElement('div');
        particle.style.position = 'absolute';
        particle.style.width = effectChosen.size +
'px';
        particle.style.height = effectChosen.size +
'px';
        particle.style.borderRadius = '50%';
        particle.style.backgroundColor =
effectChosen.color;
        particle.style.animation = 'expandCircle
1s ease-in-out';

        /*----*/

        /* calculate position to center the circle */
        let leftPosition = (window.innerWidth -
effectChosen.size) / 2;
```

```
        let topPosition = (window.innerHeight -
effectChosen.size) / 2;

        /*----*/

        particle.style.left = leftPosition + 'px';
        particle.style.top = topPosition + 'px';

        /*----*/

        document.body.append(particle);

        setTimeout(function()
        {
            /* remove the particle after animation */
            particle.remove();
        }, 700);
    }

    /*----*/
```

```
let style001 = document.createElement('style');

style001.innerHTML = `
    @keyframes expandCircle {
    0% {
        opacity: 1;
        transform: scale(1);
    }
    100% {
        opacity: 0;
        transform: scale(1.5);
    }
}`;

document.head.append(style001);

/* timer to create random efx at regular intervals */
```

```
        setInterval(function()
        {
            createRandomEfx(specialEfx);
        }, 2000);
    }

    generateSpecialEffects();

}());
```

```
javascript:(
/* Cards - Shuffle Deck - Create the deck as an
Array */
function()
{
    function shuffleDeck()
    {
        const suits = ['Hearts', 'Diamonds', 'Clubs',
'Spades'];

        const values = ['2', '3', '4', '5', '6', '7', '8', '9',
'10', 'Jack', 'Queen', 'King', 'Ace'];

        const deck = [];

        /* create the deck as an array */
        for (let i = 0; i < suits.length; i++)
        {
            for (let j = 0; j < values.length; j++)
            {
```

```
            deck.push(values[j] + ' of ' + suits[i]);
        }
    }

    /* shuffle the deck */
    for (let i = 0; i < deck.length - 1; i++)
    {
        let j = i + Math.floor(Math.random() *
(deck.length - i));

        /* swap the elements at indices i and j */
        let temp = deck[i];

        deck[i] = deck[j];

        deck[j] = temp;
    }

    return deck;
}
```

```
    let shuffledDeck = shuffleDeck();

    /* show as objects */
    console.log(shuffledDeck);

    /* show as JSON */
    console.log(JSON.stringify(shuffledDeck));

    /* show as JSON */
    alert(JSON.stringify(shuffledDeck));

}());


/* The Result of this code is shown on the next
two pages */
```

/* show as Objects */

Inspector | Console | Debugger | Network | Style Editor

Filter Output | Errors | Warnings | Logs | Info | Debug

```
    alert(JSON.stringify(shuffledDeck));

  }());
```

```
Array(52) [ "4 of Hearts", "3 of Clubs", "9 of Clubs", "Queen of Spades", "5 of Spades", "2 of Diamonds", "5 of Clubs", "King of Clubs", "Jack of Hearts", "Ace of Spades", … ]
    0: "4 of Hearts"
    1: "3 of Clubs"
    2: "9 of Clubs"
    3: "Queen of Spades"
    4: "5 of Spades"
    5: "2 of Diamonds"
    6: "5 of Clubs"
    7: "King of Clubs"
```

/* show as JSON */

Inspector Console Debugger Network Style Editor

Filter Output Errors Warnings Logs Info Debug

["4 of Hearts","3 of Clubs","9 of Clubs","Queen of Spades","5 of Spades","2 of Diamonds","5 of Clubs","King of Clubs","Jack of Hearts","Ace of Spades","Jack of Clubs","King of Hearts","King of Spades","7 of Hearts","3 of Hearts","9 of Hearts","10 of Clubs","5 of Diamonds","6 of Clubs","6 of Diamonds","8 of Diamonds","Queen of Hearts","9 of Diamonds","3 of Spades","Jack of Diamonds","5 of Hearts","2 of Hearts","7 of Diamonds","6 of Spades","4 of Diamonds","Ace of Diamonds","8 of Spades","7 of Clubs","2 of Spades","Ace of Hearts","King of Diamonds","7 of Spades","Queen of Clubs","Ace of Clubs","4 of Spades","10 of Diamonds","3 of Diamonds","6 of Hearts","10 of Spades","8 of Hearts","2 of Clubs","Jack of Spades","9 of Spades","Queen of Diamonds","10 of Hearts","8 of Clubs","4 of Clubs"]

```
javascript:(
/* Cards - Shuffle Deck - Create the deck as an
Array - Rows of 4 */
function()
{
    function shuffleDeck()
    {
        const suits = ['Hearts', 'Diamonds', 'Clubs',
'Spades'];

        const values = ['2', '3', '4', '5', '6', '7', '8', '9',
'10', 'Jack', 'Queen', 'King', 'Ace'];

        const deck = [];

        /* create the deck */
        for (let i = 0; i < suits.length; i++)
        {
            for (let j = 0; j < values.length; j++)
            {
```

```
            deck.push(values[j] + ' of ' + suits[i]);
        }
    }

    /* shuffle the deck */
    for (let i = 0; i < deck.length - 1; i++)
    {
        let j = i + Math.floor(Math.random() *
(deck.length - i));

        /* swap the elements at indices i and j */
        let temp = deck[i];

        deck[i] = deck[j];

        deck[j] = temp;
    }

    return deck;
}
```

```
    let shuffledDeck = shuffleDeck();

    /* create a container div to hold the cards */
    let container = document.createElement('div');
    container.style.position = "absolute";
    container.style.left = "100px";
    container.style.top = "10px";
    container.style.display = 'grid';
    container.style.gridTemplateColumns =
'repeat(4, 1fr)';
    container.style.gridGap = '10px';
    container.style.margin = '20px';

    /* add each card to the container */
    for (let i = 0; i < shuffledDeck.length; i++)
    {
        let cardDiv =
document.createElement('div');
        cardDiv.textContent = shuffledDeck[i];
```

```
        cardDiv.style.padding = '10px';
        cardDiv.style.border = '1px solid';
        cardDiv.style.borderColor = 'rgb(255, 255,
255)';
        cardDiv.style.borderRadius = '5px';
        cardDiv.style.fontSize = '20px';
        cardDiv.style.textAlign = 'center';
        container.append(cardDiv);
    }

    /* append the container to the body */
    document.body.append(container);

}());
```

King of Spades	King of Clubs	6 of Diamonds	3 of Spades
9 of Clubs	5 of Diamonds	8 of Diamonds	Jack of Diamonds
10 of Hearts	Jack of Spades	3 of Diamonds	6 of Clubs
5 of Spades	Jack of Hearts	10 of Clubs	Queen of Hearts
Jack of Clubs	7 of Hearts	Ace of Diamonds	7 of Diamonds
8 of Clubs	7 of Spades	Ace of Hearts	4 of Spades
2 of Clubs	3 of Clubs	King of Diamonds	4 of Diamonds
5 of Hearts	4 of Hearts	9 of Hearts	Queen of Diamonds
6 of Hearts	5 of Clubs	2 of Spades	10 of Spades
2 of Diamonds	10 of Diamonds	Queen of Clubs	8 of Hearts
9 of Diamonds	Ace of Clubs	8 of Spades	King of Hearts
Ace of Spades	9 of Spades	4 of Clubs	2 of Hearts
6 of Spades	Queen of Spades	7 of Clubs	3 of Hearts

```
javascript:(
/* Cards - Shuffle Deck - Create the deck as an
Array of Objects */
function()
{
    function shuffleDeck()
    {
        const suits = ['Hearts', 'Diamonds', 'Clubs',
'Spades'];

        const values = ['2', '3', '4', '5', '6', '7', '8', '9',
'10', 'Jack', 'Queen', 'King', 'Ace'];

        const deck = [];

        /* create the deck as an array of objects */
        for (let i = 0; i < suits.length; i++)
        {
            for (let j = 0; j < values.length; j++)
            {
```

```
            deck.push(
            {
                suit: suits[i],
                number: values[j]
            });
        }
    }

    /* shuffle the deck */
    for (let i = 0; i < deck.length - 1; i++)
    {
        let j = i + Math.floor(Math.random() *
(deck.length - i));

        let temp = deck[i];

        deck[i] = deck[j];

        deck[j] = temp;
    }
```

```
    return deck;
  }

  let shuffledDeck = shuffleDeck();

  /* show as objects */
  console.log(shuffledDeck);

  /* show as JSON */
  console.log(JSON.stringify(shuffledDeck));

  /* show as JSON */
  alert(JSON.stringify(shuffledDeck));

}());

/* The Result of this code is shown on the next
page */
```

/* show as Objects */

Inspector | Console | Debugger | Network | { }

Filter Output | Errors | Warnings | Logs | Info | Debu

```
▼ Array(52) [ {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…},
{…}, … ]
  ▶ 0: Object { suit: "Hearts", number: "8" }
  ▶ 1: Object { suit: "Spades", number: "Jack" }
  ▶ 2: Object { suit: "Spades", number: "10" }
  ▶ 3: Object { suit: "Hearts", number: "Jack" }
  ▶ 4: Object { suit: "Spades", number: "7" }
  ▶ 5: Object { suit: "Diamonds", number: "5" }
```

/* show as JSON */

Inspector | Console | Debugger | Network | { }

Filter Output | Errors | Warnings | Logs | Info | Debu


```
[{"suit":"Hearts","number":"8"},
{"suit":"Spades","number":"Jack"},
{"suit":"Spades","number":"10"},
{"suit":"Hearts","number":"Jack"},
{"suit":"Spades","number":"7"},
{"suit":"Diamonds","number":"5"},
{"suit":"Clubs","number":"6"},{"suit":"Clubs","number":"4"
{"suit":"Spades","number":"King"},
```

```
javascript:(
/* Cards - Shuffle Deck - Create the deck as an
Array of Objects - Rows of 4 */
function()
{
    function shuffleDeck()
    {
        const suits = ['Hearts', 'Diamonds', 'Clubs',
'Spades'];

        const values = ['2', '3', '4', '5', '6', '7', '8', '9',
'10', 'Jack', 'Queen', 'King', 'Ace'];

        const deck = [];

        /* create the deck as an array of objects */
        for (let i = 0; i < suits.length; i++)
        {
            for (let j = 0; j < values.length; j++)
            {
```

```
            deck.push(
            {
                suit: suits[i],
                number: values[j]
            });
        }
    }

    /* shuffle the deck */
    for (let i = 0; i < deck.length - 1; i++)
    {
        let j = i + Math.floor(Math.random() *
(deck.length - i));

        let temp = deck[i];

        deck[i] = deck[j];

        deck[j] = temp;
    }
```

```
    return deck;
}

let shuffledDeck = shuffleDeck();

/* show as objects */
console.log(shuffledDeck);

/* show as JSON */
console.log(JSON.stringify(shuffledDeck));

/*----*/

/* create a container div to hold the cards */
let container = document.createElement('div');
container.style.position = "absolute";
container.style.left = "100px";
container.style.top = "10px";
container.style.display = 'grid';
```

```
    container.style.gridTemplateColumns =
'repeat(4, 1fr)';
    container.style.gridGap = '10px';
    container.style.margin = '20px';

    /* add each card to the container */
    for (let i = 0; i < shuffledDeck.length; i++)
    {
        let cardDiv =
document.createElement('div');
        cardDiv.textContent =
shuffledDeck[i].number + " of " +
shuffledDeck[i].suit;
        cardDiv.style.padding = '10px';
        cardDiv.style.border = '1px solid';
        cardDiv.style.borderColor = 'rgb(255, 255,
255)';
        cardDiv.style.borderRadius = '5px';
        cardDiv.style.fontSize = '20px';
        cardDiv.style.textAlign = 'center';
```

```
        container.append(cardDiv);
    }

    /* append the container to the body */
    document.body.append(container);

}());
```

Queen of Clubs	4 of Diamonds	Ace of Hearts	2 of Diamonds
Ace of Diamonds	Queen of Diamonds	King of Clubs	4 of Spades
2 of Clubs	Ace of Spades	4 of Clubs	Jack of Diamonds
Jack of Spades	Jack of Clubs	9 of Spades	8 of Clubs
3 of Spades	9 of Clubs	5 of Spades	Jack of Hearts
7 of Clubs	3 of Diamonds	9 of Diamonds	7 of Hearts
King of Diamonds	8 of Diamonds	6 of Hearts	9 of Hearts
5 of Clubs	5 of Hearts	3 of Hearts	Ace of Clubs
7 of Diamonds	3 of Clubs	2 of Spades	5 of Diamonds
Queen of Hearts	4 of Hearts	6 of Spades	10 of Hearts
6 of Diamonds	10 of Diamonds	King of Hearts	7 of Spades
2 of Hearts	10 of Spades	Queen of Spades	8 of Hearts
King of Spades	6 of Clubs	10 of Clubs	8 of Spades

```
javascript:(
/* Trigonometry - Convert Degrees To Radians */
function()
{
    function getAngleInDegree()
    {
        let angle = parseFloat(prompt("Enter an
angle in degrees:"));

        return angle;
    }

    function convertDegreeToRadian(angle)
    {
        if (!isNaN(angle))
        {
            /* angle in degree converted to radian */
            let radians = angle * (Math.PI / 180);

            return radians;
```

```
        }
        else
        {
            alert("Angle wasn't entered in numbers");
        }
    }

    let result =
convertDegreeToRadian(getAngleInDegree());

    console.log(result);

    alert(result);

}());
```

```
javascript:(
/* Trigonometry - Sine */
function()
{
    function getAngleInDegree()
    {
        let angle = parseFloat(prompt("Enter an
angle in degrees:"));

        return angle;
    }

    function convertDegreeToRadian(angle)
    {
        if (!isNaN(angle))
        {
            /* angle in degree converted to radian */
            let radians = angle * (Math.PI / 180);

            return radians;
```

```
        }
        else
        {
            alert("Angle wasn't entered in numbers");
        }
    }

    let theRadians =
convertDegreeToRadian(getAngleInDegree());

    function sineOfAngle()
    {
        let sine = Math.sin(theRadians).toFixed(2);

        return sine;
    }
    console.log(sineOfAngle());
    alert(sineOfAngle());
}());
```

```
javascript:(
/* Trigonometry - Cosine */
function()
{
    function getAngleInDegree()
    {
        let angle = parseFloat(prompt("Enter an
angle in degrees:"));

        return angle;
    }

    function convertDegreeToRadian(angle)
    {
        if (!isNaN(angle))
        {
            /* angle in degree converted to radian */
            let radians = angle * (Math.PI / 180);

            return radians;
```

```
        }
        else
        {
            alert("Angle wasn't entered in numbers");
        }
    }

    let theRadians =
convertDegreeToRadian(getAngleInDegree());

    function cosineOfAngle()
    {
        let cosine =
Math.cos(theRadians).toFixed(2);

        return cosine;
    }
    console.log(cosineOfAngle());
    alert(cosineOfAngle());
}());
```

```
javascript:(
/* Trigonometry - Tangent */
function()
{
    function getAngleInDegree()
    {
        let angle = parseFloat(prompt("Enter an
angle in degrees:"));

        return angle;
    }

    function convertDegreeToRadian(angle)
    {
        if (!isNaN(angle))
        {
            /* angle in degree converted to radian */
            let radians = angle * (Math.PI / 180);

            return radians;
```

```
        }
        else
        {
            alert("Angle wasn't entered in numbers");
        }
    }

    let theRadians =
convertDegreeToRadian(getAngleInDegree());

    function tangentOfAngle()
    {
        let tangent =
Math.tan(theRadians).toFixed(2);

        return tangent;
    }
    console.log(tangentOfAngle());
    alert(tangentOfAngle());
}());
```

```
javascript:(
/* Trigonometry - Sine, Cosine, Tangent
Calculator */
function()
{
    function calculateTrigonometry()
    {
        let angle = parseFloat(prompt("Enter an
angle in degrees:"));

        if (!isNaN(angle))
        {
            let radians = angle * (Math.PI / 180);
            let sine = Math.sin(radians);
            let cosine = Math.cos(radians);
            let tangent = Math.tan(radians);

            let message = "Angle: " + angle + "
degrees\n";
```

```
            message += "Sine: " + sine.toFixed(4) + "\
n";

            message += "Cosine: " +
cosine.toFixed(4) + "\n";

            message += "Tangent: " +
tangent.toFixed(4);

            alert(message);
        }
        else
        {
            alert("Angle wasn't entered in numbers");
        }
    }

    calculateTrigonometry();

}());
```

```
javascript:(
/* Trigonometry - Circle made using Divs */
function()
{
    function generateCircleOfDivs()
    {
        /* number of divs to create */
        let numberOfDivs = 360;

        /* the radius of the circle */
        let radius = 50;

        /* x pos of the circle's center */
        let centerX = 100;

        /* y pos of the circle's center */
        let centerY = 100;

        let container =
document.createElement("div");
```

```
container.style.width = "200px";
container.style.height = "200px";
container.style.position = "fixed";
container.style.left = 100 + "px";
container.style.top = 100 + "px";
container.style.border = "1px solid black";
document.body.append(container);

for (let x = 0; x < numberOfDivs; x++)
{
    let angle = (x / numberOfDivs) * 2 *
Math.PI;

    let xPos = centerX + radius *
Math.cos(angle);

    let yPos = centerY + radius *
Math.sin(angle);

    /*----*/
```

```
        let div = document.createElement("div");
        div.style.width = "10px";
        div.style.height = "10px";
        div.style.background = "blue";
        div.style.borderRadius = "50%";
        div.style.position = "absolute";
        div.style.left = xPos + "px";
        div.style.top = yPos + "px";
        container.append(div);
    }
  }

  generateCircleOfDivs();

}());
```

```
javascript:(
/* Trigonometry - Spinning Circle of Divs */
function()
{
    function generateCircleOfMovingDivs()
    {
        /* number of divs to create */
        let numberOfDivs = 36;

        /* radius of the circle */
        let radius = 50;

        /* center position */
        let centerX = 100;
        let centerY = 100;

        let container =
document.createElement("div");
        container.style.position = "fixed";
        container.style.left = "20px";
```

```
container.style.top = "20px";
container.style.width = "200px";
container.style.height = "200px";
document.body.append(container);

for (let x = 0; x < numberOfDivs; x++)
{
    let angle = (x / numberOfDivs) * 2 *
Math.PI;

    let xPos = centerX + radius *
Math.cos(angle);

    let yPos = centerY + radius *
Math.sin(angle);

    let div = document.createElement("div");
    div.style.position = "absolute";
    div.style.left = xPos + "px";
    div.style.top = yPos + "px";
```

```
div.style.width = "10px";
div.style.height = "10px";
div.style.background = "blue";
div.style.borderRadius = "50%";
container.append(div);

/* animate the divs */
setInterval(function()
{
    let rotateAngle = (x / numberOfDivs) * 2
* Math.PI + (new Date().getTime() / 1000);

    xPos = centerX + radius *
Math.cos(rotateAngle);

    yPos = centerY + radius *
Math.sin(rotateAngle);

    div.style.left = xPos + "px";
```

```
            div.style.top = yPos + "px";

        }, 50);
    }
}

generateCircleOfMovingDivs();

}());
```

```
javascript:(
/* Trigonometry - Pulsating Circle of Divs */
function()
{
    function circleOfDivsPulsating()
    {
        /* number of divs to create */
        let numberOfDivs = 60;

        /* radius of the circle */
        let radius = 50;

        /* center position */
        let centerX = 100;
        let centerY = 100;

        /* container for the circle */
        let container =
document.createElement("div");
        container.style.position = "fixed";
```

```
        container.style.left = "100px";
        container.style.top = "100px";
        container.style.width = "200px";
        container.style.height = "200px";
        container.style.margin = "0 auto";
        container.style.overflow = "hidden";
        document.body.append(container);

        /* create the divs */
        for (let x = 0; x < numberOfDivs; x++)
        {
            let angle = (x / numberOfDivs) * 2 *
Math.PI;

            let xPos = centerX + radius *
Math.cos(angle);

            let yPos = centerY + radius *
Math.sin(angle);
```

```
    let div = document.createElement("div");
    div.style.width = "10px";
    div.style.height = "10px";
    div.style.background = "blue";
    div.style.borderRadius = "50%";
    div.style.position = "absolute";
    div.style.left = xPos + "px";
    div.style.top = yPos + "px";
    container.append(div);
}

/* animate the pulsating effect */
let scale = 1;
let growing = true;

function animate()
{
    if (growing)
    {
        scale += 0.01;
```

```
            if (scale >= 2)
            {
                growing = false;
            }
        }
        else
        {
            scale -= 0.01;

            if (scale <= 0.5)
            {
                growing = true;
            }
        }

        container.style.transform =
'scale('+scale+')';
    }
```

```
        setInterval(function()
        {
            animate();
        }, 10);
    }

    circleOfDivsPulsating();

}());
```

```
javascript:(
/* Trigonometry - Draw Sine and Cosine */
function()
{
    function graphSineAndCosine()
    {
        let canvas =
document.createElement("canvas");
        canvas.style.position = "absolute";
        canvas.style.left = 100 + "px";
        canvas.style.top = 50 + "px";
        canvas.width = 700;
        canvas.height = 500;

        /*----*/

        let context = canvas.getContext("2d");

        /* clear the canvas */
        context.clearRect(0, 0, canvas.width,
```

```
canvas.height);

/* draw x and y axes */
context.beginPath();
context.moveTo(0, canvas.height / 2);
context.lineTo(canvas.width,
canvas.height / 2);
context.moveTo(canvas.width / 2, 0);
context.lineTo(canvas.width / 2,
canvas.height);
context.strokeStyle = "black";
context.stroke();
context.closePath();

/* draw sine function */
context.beginPath();
context.moveTo(0, canvas.height / 2);

for (let x = 0; x < canvas.width; x++)
{
```

```
        let radians = (x - canvas.width / 2) *
(Math.PI / 180);

        let y = Math.sin(radians) * (canvas.height /
4) + (canvas.height / 2);

        context.lineTo(x, y);
    }

    context.strokeStyle = "blue";
    context.lineWidth = 4;
    context.stroke();
    context.closePath();

    /* draw cosine function */
    context.beginPath();
    context.moveTo(0, canvas.height / 2);

    for (let x = 0; x < canvas.width; x++)
    {
```

```
            let radians = (x - canvas.width / 2) *
(Math.PI / 180);

            let y = Math.cos(radians) * (canvas.height
/ 4) + (canvas.height / 2);

            context.lineTo(x, y);
        }

        context.strokeStyle = "red";
        context.lineWidth = 4;
        context.stroke();
        context.closePath();

        document.body.append(canvas);
    }

    graphSineAndCosine();

}());
```

```
/*
creates a dynamic trigonometric graph using
canvas to draw the sine and cosine functions
*/
```

```
javascript:(
/* Trigonometry - Sine Wave Motion */
function()
{
    function generateSineWaveWithMotion()
    {
        let amplitude = 50;
        let frequency = 0.1;

        let xOffset = 0;
        let yOffset = 100;

        let canvasWidth = window.innerWidth;
        let canvasHeight = 200;

        /*----*/

        let canvas =
document.createElement('canvas');
        canvas.style.position = 'fixed';
```

```
canvas.style.left = '0';
canvas.style.top = '0';
canvas.width = canvasWidth;
canvas.height = canvasHeight;
canvas.style.zIndex = '10000';
canvas.style.pointerEvents = 'none';
document.body.append(canvas);

/*----*/

let context = canvas.getContext('2d');

/*----*/

function drawSineWave()
{
    context.clearRect(0, 0, canvas.width,
canvas.height);

    context.beginPath();
```

```
        for (let x = 0; x < canvasWidth; x += 1)
        {
            let y = amplitude * Math.sin(frequency *
x + xOffset) + yOffset;

            context.lineTo(x, y);
        }

        context.strokeStyle = 'blue';
        context.lineWidth = 4;
        context.stroke();
    }

    function animate()
    {
        xOffset += 1;
        drawSineWave();
        requestAnimationFrame(animate);
    }
```

```
        animate();
    }

    generateSineWaveWithMotion();

}());
```

```
javascript:(
/* Trigonometry - Cosine Wave Motion */
function()
{
    function generateCosineWaveWithMotion()
    {
        let amplitude = 50;
        let frequency = 0.05;

        let xOffset = 0;
        let yOffset = 100;

        let canvasWidth = window.innerWidth;
        let canvasHeight = 200;

        /*----*/

        let canvas =
document.createElement('canvas');
        canvas.style.position = 'fixed';
```

```
canvas.style.left = '0';
canvas.style.top = '0';
canvas.width = canvasWidth;
canvas.height = canvasHeight;
canvas.style.zIndex = '10000';
canvas.style.pointerEvents = 'none';
document.body.append(canvas);

/*----*/

let context = canvas.getContext('2d');

/*----*/

function drawCosineWaves()
{
    context.clearRect(0, 0, canvas.width,
canvas.height);

    /* draw the cosine wave in red */
```

```
        context.beginPath();
        context.strokeStyle = 'red';
        context.lineWidth = 4;

        for (let x = 0; x < canvasWidth; x += 1)
        {
            let y = amplitude * Math.cos(frequency
* x + xOffset) + yOffset;
            context.lineTo(x, y);
        }
        context.stroke();
    }

    /*----*/

    function animate()
    {
        xOffset += 0.1;
        drawCosineWaves();
        requestAnimationFrame(animate);
```

```
        }

        animate();
    }

    generateCosineWaveWithMotion();

}());
```

```
javascript:(
/* Trigonometry - Tangent Wave Motion */
function()
{
    function generateTangentWaveWithMotion()
    {
        let amplitude = 50;
        let frequency = 0.05;

        let xOffset = 0;
        let yOffset = 100;

        let canvasWidth = window.innerWidth;
        let canvasHeight = 200;

        /*----*/

        let canvas =
document.createElement('canvas');
        canvas.style.position = 'fixed';
```

```
canvas.style.left = '0';
canvas.style.top = '0';
canvas.width = canvasWidth;
canvas.height = canvasHeight;
canvas.style.zIndex = '10000';
canvas.style.pointerEvents = 'none';
document.body.append(canvas);

/*----*/

let context = canvas.getContext('2d');

/*----*/

function drawTangentWave()
{
    context.clearRect(0, 0, canvas.width,
canvas.height);
    context.beginPath();
```

```
        for (let x = 0; x < canvasWidth; x += 1)
        {
            let y = amplitude * Math.tan(frequency *
x + xOffset) + yOffset;
            context.lineTo(x, y);
        }

        context.strokeStyle = 'green';
        context.lineWidth = 2;
        context.stroke();
    }

    function animate()
    {
        xOffset += 0.1;
        drawTangentWave();
        requestAnimationFrame(animate);
    }

    animate();
```

```
    }

    generateTangentWaveWithMotion();

}());
```

```
javascript:(
/* Trigonometry - Sine and Cosine Wave Motion
*/
function()
{
    function
generateSineAndCosineWavesWithMotion()
    {
        let amplitude = 50;
        let frequency = 0.05;

        let xOffset = 0;
        let yOffset = 100;

        let canvasWidth = window.innerWidth;
        let canvasHeight = 200;

        /*----*/
```

```
let canvas =
document.createElement('canvas');
canvas.style.position = 'fixed';
canvas.style.left = '0';
canvas.style.top = '0';
canvas.width = canvasWidth;
canvas.height = canvasHeight;
canvas.style.zIndex = '10000';
canvas.style.pointerEvents = 'none';
document.body.append(canvas);

/*----*/

let context = canvas.getContext('2d');

/*----*/

function drawSineCosineWaves()
{
```

```
        context.clearRect(0, 0, canvas.width,
canvas.height);

        /* draw the sine wave in blue */
        context.beginPath();
        context.strokeStyle = 'blue';
        context.lineWidth = 2;

        for (let x = 0; x < canvasWidth; x += 1)
        {
            let y = amplitude * Math.sin(frequency *
x + xOffset) + yOffset;
            context.lineTo(x, y);
        }

        context.stroke();

        /* draw the cosine wave in red */
        context.beginPath();
        context.strokeStyle = 'red';
```

```
        for (let x = 0; x < canvasWidth; x += 1)
        {
            let y = amplitude * Math.cos(frequency
* x + xOffset) + yOffset;
            context.lineTo(x, y);
        }

        context.stroke();
    }

    /*----*/

    function animate()
    {
        xOffset += 0.1;
        drawSineCosineWaves();
        requestAnimationFrame(animate);
    }
```

```
    animate();
  }

  generateSineAndCosineWavesWithMotion();

}());
```

```
javascript:(
/* Trigonometry - Draw a Circle - Draw a 45
Degree Line - Create a Label */
function()
{
    function drawCircleWith45DegreeLine()
    {
        let canvasWidth = 400;
        let canvasHeight = 400;

        let canvas =
document.createElement('canvas');
        canvas.style.position = 'fixed';
        canvas.style.left = '0';
        canvas.style.top = '0';
        canvas.width = canvasWidth;
        canvas.height = canvasHeight;
        canvas.style.zIndex = '10000';
        canvas.style.pointerEvents = 'none';
        document.body.append(canvas);
```

```
/*----*/

let context = canvas.getContext('2d');

/*----*/
function drawCircle()
{
    let centerX = canvasWidth / 2;
    let centerY = canvasHeight / 2;

    let radius = 150;

    context.strokeStyle = 'aqua';

    context.lineWidth = 4;

    /* draw the circle */
    context.beginPath();
```

```
        context.arc(centerX, centerY, radius, 0, 2
* Math.PI);

        context.stroke();

        /*----*/

        /* draw the radius line at an angle */
        let angleInDegrees = 45;

        let angleInRadians = (angleInDegrees *
Math.PI) / 180;

        let endX = centerX + radius *
Math.cos(angleInRadians);

        let endY = centerY + radius *
Math.sin(angleInRadians);

        context.strokeStyle = 'white';
```

```
            context.lineWidth = 4;
            context.beginPath();
            context.moveTo(centerX, centerY);
            context.lineTo(endX, endY);
            context.stroke();

            /*----*/

            /* label the angle in degrees */
            context.font = '30px arial';
            context.fillStyle = 'white';
            context.fillText(angleInDegrees + '°',
centerX + 20, centerY - 20);
        }
        drawCircle();
    }

    drawCircleWith45DegreeLine();

}());
```

```
/*
creates a circle with a 45 degree line and a label
*/
```

```
javascript:(
/* Trigonometry - Angle between two objects
Constant Updates */
function()
{
    let object1 = document.createElement("div");
    object1.style.position = "absolute";
    object1.style.top = "50px";
    object1.style.left = "50px";
    object1.style.width = "20px";
    object1.style.height = "20px";
    object1.style.background = "red";
    document.body.append(object1);

    /*----*/

    let object2 = document.createElement("div");
    object2.style.position = "absolute";
    object2.style.top = "100px";
    object2.style.left = "200px";
```

```
    object2.style.width = "20px";
    object2.style.height = "20px";
    object2.style.background = "blue";
    document.body.append(object2);

    function calculateAngle()
    {
        let deltaX = object2.offsetLeft -
object1.offsetLeft;

        let deltaY = object2.offsetTop -
object1.offsetTop;

        let angle = Math.atan2(deltaY, deltaX) * (180
/ Math.PI);

        return angle;
    }

    function updateAngleDisplay()
```

```
{
    let angle = calculateAngle();

    console.log("Angle between objects: " +
angle.toFixed(2) + " degrees");
}

function moveObjectDown()
{
    let top = parseInt(object2.style.top) || 0;

    object2.style.top = (top + 1) + "px";

    /* move the blue object down */
    updateAngleDisplay();
}

function animate()
{
    moveObjectDown();
```

```
        requestAnimationFrame(animate);
    }

    animate();

    /* start moving the blue object down and
updating the angle */
    moveObjectDown();

}());
```

Inspector
Console
Debugger
Filter Output
Angle between objects: 47.20 degrees
Angle between objects: 47.38 degrees
Angle between objects: 47.55 degrees
Angle between objects: 47.73 degrees
Angle between objects: 47.90 degrees
Angle between objects: 48.07 degrees

```
javascript:(
/* Make Roads like in Cities Skylines */
function()
{
    function makeRoad()
    {
        let road = document.createElement("div");
        road.style.position = "absolute";
        road.style.backgroundColor = "gray";
        road.style.height = "20px";
        road.style.width = "5px";
        road.style.pointerEvents = "none";
        document.body.append(road);

        let isDrawing = false;

        document.addEventListener("mousedown",
function(e)
        {
            isDrawing = true;
```

```
        road.style.left = e.pageX + "px";
        road.style.top = e.pageY + "px";
    });

    document.addEventListener("mouseup",
function()
    {
        isDrawing = false;
    });

    document.addEventListener("mousemove",
function(e)
    {
        if (isDrawing)
        {
            let currentX = e.pageX;
            let currentY = e.pageY;

            let deltaX = currentX -
parseInt(road.style.left);
```

```
            let deltaY = currentY - parseInt(road.style.top);

            let length = Math.sqrt(deltaX * deltaX + deltaY * deltaY);

            let angle = Math.atan2(deltaY, deltaX);

            road.style.width = length + "px";

            road.style.transform = "rotate(" + angle + "rad)";
        }
    });
}

makeRoad();

}());
```

/*
After we have activated this bookmarklet, we hold left click and drag our mouse to make the road, and then let go of the left click to place the road at that angle.

Now at anytime that we want we can left click to place that angled road piece anywhere on the screen.
*/

```
javascript:(
/* Password Generator - Specify How Many
Characters without Input Validation */
function()
{
    function makePassword()
    {
        function generatePassword(whichAmount)
        {
            const characters =
"abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLM
NOPQRSTUVWXYZ0123456789!@#$%^&*()_-
+=<>?/{}[]";

            let password = "";

            for (let x = 0; x < whichAmount; x++)
            {
                const randomIndex =
Math.floor(Math.random() * characters.length);
```

```
            password +=
characters.charAt(randomIndex);
        }

        return password;
    }

    const passwordLength = prompt("Enter
how many characters");

    if (passwordLength !== null)
    {
        const parsedLength =
parseInt(passwordLength);

        const generatedPassword =
generatePassword(parsedLength);
```

```
            console.log("Password is " +
generatedPassword);

            alert("Password is " +
generatedPassword);
        }
    }

    makePassword();

}());
```

```
javascript:(
/* Password Generator - Specify How Many
Characters with Input Validation */
function()
{
    function makePassword()
    {
        function generatePassword(whichAmount)
        {
            const characters =
"abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLM
NOPQRSTUVWXYZ0123456789!@#$%^&*()_-
+=<>?/{}[]";

            let password = "";

            for (let x = 0; x < whichAmount; x++)
            {
                const randomIndex =
Math.floor(Math.random() * characters.length);
```

```
            password +=
characters.charAt(randomIndex);
        }

        return password;
    }

    const passwordLength = prompt("Enter
how many characters");

    if (passwordLength !== null)
    {
        const parsedLength =
parseInt(passwordLength);

        if (!isNaN(parsedLength) &&
parsedLength > 0)
        {
```

```
            const generatedPassword =
generatePassword(parsedLength);

            console.log("Password is " +
generatedPassword);

            alert("Password is " +
generatedPassword);
        }
        else
        {
            alert("You didn't enter how many
characters you want the password to be.");
        }
    }
  }

  makePassword();

}());
```

```
javascript:(
/* Robot AI - Array of Objects - Responds once to
EXACT word(s) */
function()
{
    const responses =
    [
        {
            keyword: "hi",
            response: "Howdy! Good to see you bud"
        },

        {
            keyword: "how is the weather today",
            response: "Very Nice!"
        },

        {
            keyword: "how are you",
            response: "I'm doing good, you?"
```

```
        },

        {
            keyword: "bye",
            response: "Bye. Talk to you soon"
        }
    ];

    function createRobot()
    {
        const userInput = prompt("Ask me a
question:");

        if (userInput)
        {
            const responseObj =
responses.find(function (obj)
            {
                return obj.keyword.toLowerCase() ===
userInput.toLowerCase();
```

```
            });

            if (responseObj)
            {
                alert(responseObj.response);
            }
            else
            {
                alert("Rephrase the question");
            }
        }
    }

    createRobot();

}());
```

```
javascript:(
/* Robot AI - Array of Objects - Responds to
EXACT word(s) Continuously */
function()
{
    const responses =
    [
        {
            keyword: "hi",
            response: "Howdy! Good to see you,
bud"
        },
        {
            keyword: "how is the weather",
            response: "Very nice!"
        },
        {
            keyword: "how are you",
            response: "I'm doing good, you?"
        },
```

```
        {
            keyword: "bye",
            response: "Bye. Talk to you soon"
        }
    ];

    function createRobot()
    {
        while (true)
        {
            const userInput = prompt("Ask me a
question:");

            if (!userInput)
            {
                /* exit the loop if the user cancels or
enters nothing */
                break;
            }
```

```
        const responseObj =
responses.find(function (obj)
        {
            return obj.keyword.toLowerCase() ===
userInput.toLowerCase();
        });

        if (responseObj)
        {
            alert(responseObj.response);
        }
        else
        {
            alert("Rephrase the question");
        }
    }
  }
  createRobot();
}());
```

```
javascript:(
/* Robot AI - Array Of Objects - Responds to
word(s) continuously with language
comprehension (keyword-based) */
function()
{
    const responses =
    [
        {
            keyword: "hi",
            response: "Howdy! Good to see you,
bud"
        },

        {
            keyword: "weather",
            response: "Very nice!"
        },

        {
```

```
            keyword: "how are you",
            response: "I'm doing good, you?"
        },

        {
            keyword: "bye",
            response: "Bye. Talk to you soon"
        }
    ];

    function createRobot()
    {
        while (true)
        {
            const userInput = prompt("Ask me a
question:");

            if (!userInput)
            {
```

```
            /* exit the loop if the user cancels or
enters nothing */
            break;
        }

        /* check for keywords in the user's input */
        const responseObj =
responses.find(function(obj)
        {
            return
userInput.toLowerCase().includes(obj.keyword);
        });

        if (responseObj)
        {
            alert(responseObj.response);

            if (responseObj.keyword === "bye")
            {
                /* exit the loop on "bye" */
```

```
                break;
            }
        }
        else
        {
            alert("Rephrase the question");
        }
    }
}

createRobot();

}());
```

```
javascript:(
/* Robot AI - Array Of Objects - Responds to
word(s) continuously with Language
Comprehension (keyword-based) and
Contextual Responses */
function()
{
    function createRobot()
    {
        const responses =
        [
            {
                keyword: "hi",
                response: "Howdy bud"
            },
            {
                keyword: "weather",
                response: "Very nice!"
            },
            {
```

```
            keyword: "how are you",
            response: "I'm doing good, you?"
        },
        {
            keyword: "bye",
            response: "Bye. Talk to you soon"
        }
    ];

    /* initialize context variable */
    let context = null;

    while (true)
    {
        const userInput = prompt("Ask me a
question:");

        if (!userInput)
        {
```

```
            /* exit the loop if the user cancels or
enters nothing */
            break;
        }

        /* check for keywords in the user's input */
        const responseObj =
responses.find(function(obj)
        {
            return
userInput.toLowerCase().includes(obj.keyword);
        });

        if (responseObj)
        {
            /* check for context and provide
contextual response */
            if (context === responseObj.keyword)
            {
                if (context === "weather")
```

```
        {
            alert("Weather is still nice.");
        }
        else if (context === "how are you")
        {
            alert("Always Excellent");
        }
        else if (context === "hi")
        {
            alert("Hi again.");
        }
    }
    else
    {
        alert(responseObj.response);
    }

    if (responseObj.keyword === "bye")
    {
        /* exit the loop on "bye" */
```

```
                break;
            }

            /* update context based on user input */
            context = responseObj.keyword;
        }
        else
        {
            alert("Rephrase the question");
        }
    }
}

createRobot();

}());
```

```
javascript:(
/* Robot AI - Array Of Objects - Language Comprehension (keyword-based), Contextual Responses and Random Responses for when no keywords are found */
function()
{
    const responses =
    [
        {
            keyword: "hi",
            response: "Howdy bud"
        },
        {
            keyword: "weather",
            response: "Very nice!"
        },
        {
            keyword: "how are you",
            response: "I'm doing good, you?"
```

```
    },
    {
        keyword: "bye",
        response: "Bye. Talk to you soon"
    }
];

const randomResponses =
[
    "That's interesting!",
    "Would you tell me more?",
    "Hmm, tell me more about that.",
    "Would you elaborate?",
    "Interesting, please go on.",
    "Fascinating! Tell me more.",
];

function createRobot()
{
    /* initialize context variable */
```

```
let context = null;

function getRandomResponse()
{
    const randomIndex =
Math.floor(Math.random() *
randomResponses.length);

    return randomResponses[randomIndex];
}

while (true)
{
    const userInput = prompt("Ask me a
question:");

    if (!userInput)
    {
        /* exit the loop if the user cancels or
enters nothing */
```

```
            break;
        }

        /* check for keywords in the user's input */
        const responseObj =
responses.find(function(obj)
        {
            return
userInput.toLowerCase().includes(obj.keyword);
        });

        if (responseObj)
        {
            /* check for context and provide
contextual response */
            if (context === responseObj.keyword)
            {
                if (context === "weather")
                {
                    alert("Weather is still nice.");
```

```
        }
        else if (context === "how are you")
        {
            alert("Always Excellent");
        }
        else if (context === "hi")
        {
            alert("Hi again.");
        }
    }
    else
    {
        alert(responseObj.response);
    }

    if (responseObj.keyword === "bye")
    {
        /* exit the loop on "bye" */
        break;
    }
```

```
            /* update context based on user input */
            context = responseObj.keyword;
        }
        else
        {
            /* generate a random response for
unrecognized input */
            alert(getRandomResponse());
        }
    }
  }

  createRobot();

}());
```

```
javascript:(
/* Robot AI - Array Of Objects - Language Comprehension (keyword-based), Contextual Responses, Random Responses and Calculations */
function()
{
    const responses =
    [
        {
            keyword: "hi",
            response: "Howdy bud"
        },
        {
            keyword: "weather",
            response: "It's a beautiful day outside!"
        },
        {
            keyword: "how are you",
            response: "I'm doing good, you?"
```

```
    },
    {
        keyword: "bye",
        response: "Bye. Talk to you soon."
    },
    {
        keyword: "interests",
        response: "Computer Science is fun."
    }
];

const randomResponses =
[
    "That's interesting!",
    "Would you tell me more?",
    "Hmm, tell me more about that.",
    "Would you elaborate?",
    "Interesting, please go on.",
    "Fascinating! Tell me more.",
];
```

```
function createRobot()
{
    /* initialize context variable */
    let context = null;

    function getRandomResponse()
    {
        const randomIndex =
Math.floor(Math.random() *
randomResponses.length);

        return randomResponses[randomIndex];
    }

    while (true)
    {
        const userInput = prompt("Ask me a
question:");
```

```
        if (!userInput)
        {
            /* exit the loop if the user cancels or
enters nothing */
            break;
        }

        /* check for keywords in the user's input */
        const responseObj =
responses.find(function(obj)
        {
            return
userInput.toLowerCase().includes(obj.keyword);
        });

        if (responseObj)
        {
            /* check for context and provide
contextual response */
            if (context === responseObj.keyword)
```

```
{
    if (context === "weather")
    {
        alert("Weather is still nice.");
    }
    else if (context === "how are you")
    {
        alert("Always excellent.");
    }
    else if (context === "hi")
    {
        alert("Hi again.");
    }
}
else
{
    alert(responseObj.response);
}

if (responseObj.keyword === "bye")
```

```
            {
                /* exit the loop on "bye" */
                break;
            }

            /*----*/

            /* update context based on user input */
            context = responseObj.keyword;
        }

        else
        {
            /* check if the user input is a
calculation */
            const result = calculate(userInput);

            if (result !== null)
            {
                alert("The result is: " + result);
```

```
            }
            else
            {
                /* generate a random response for
unrecognized input */
                alert(getRandomResponse());
            }
        }
    }
}

function calculate(input)
{
    try
    {
        return eval(input);
    }
    catch(error)
    {
        /* invalid calculation */
```

```
            return null;
        }
    }

    createRobot();

}());
```

/

If the user opens a new tab in their browser and activates this script and then attempts to use the new calculation feature, then the browser security will not allow it.

Content-Security-Policy: The page's settings blocked the loading of a resource at eval ("script-src").

To fix this issue and be able to use the calculation feature, we must first go to any

website, such as google, and then activate our script.

When we are on any website, we are allowed to use the calculation eval feature.

Type in 2+2, then press OK.

*/

```
javascript:(
/* Robot AI - Random Responses every time to
keywords found. Other random response when
no keywords are found - and Calculations */
function()
{
    /* if keyword is found, use these responses */
    const responses =
    [
        {
            keyword: "hi",
            responses: ["Howdy", "Hi there", "Hi"]
        },
        {
            keyword: "weather",
            responses: ["It's a beautiful day.", "The
weather is very nice.", "It's sunny and warm."]
        },
        {
            keyword: "how are you",
```

```
            responses: ["I'm doing good, you?",
"Having fun and you?", "I'm good, how about
you?"]
        },
        {
            keyword: "bye",
            responses: ["Bye. Talk to you soon.",
"Goodbye!", "Have fun."]
        },
        {
keyword: "interests",
responses: ["Computer Science is fun.",
"Programming is lots of fun.", "I like
programming a lot."]
        }
    ];

    /* if no keywords are found, use these
reponses */
    const randomResponses =
```

```
    [
        "That's interesting!",
        "Would you tell me more?",
        "Hmm, tell me more about that.",
        "Would you elaborate?",
        "Interesting, please go on.",
        "Fascinating! Tell me more.",
    ];

    function createRobot()
    {
        let context = null;

        while (true)
        {
            const userInput = prompt("Ask me a
question:");

            if (!userInput)
            {
```

```
            break;
        }

        let responseObj =
responses.find(function(obj)
        {
            return
userInput.toLowerCase().includes(obj.keyword);
        });

        /* if keyword is found */
        if (responseObj)
        {
            const randomIndex =
Math.floor(Math.random() *
responseObj.responses.length);

  alert(responseObj.responses[randomIndex]);

            if (responseObj.keyword === "bye")
```

```
            {
                break;
            }

            context = responseObj.keyword;
        }
        else
        {
            const result = calculate(userInput);

            if (result !== null)
            {
                alert("The result is: " + result);
            }
            /* if no keyword is found */
            else
            {
                const randomIndex =
Math.floor(Math.random() *
randomResponses.length);
```

```
            alert(randomResponses[randomIndex]);
            }
        }
    }
}

function calculate(input)
{
    try
    {
        return eval(input);
    }
    catch (error)
    {
        return null;
    }
}
createRobot();
}());
```

```
javascript:(
/* Robot AI - Random Responses every time to
keywords and phrases and variations of
keywords and phrases found. Other random
response when no keywords are found - and
Calculations */
function()
{
    /* if keyword is found, use these responses */
    let responses =
    [
        {
            keywords: ["hi", "howdy", "hey"],
            responses: ["Howdy", "Hi there", "Hi"]
        },
        {
            keywords: ["weather", "forecast",
"sunny", "cloudy"],
            responses: ["It's a beautiful day.", "The
weather is very nice.", "It's sunny and warm."]
```

```
        },
        {
            keywords: ["how are you", "what's up",
"what are you up to?"],
            responses: ["I'm doing good, you?",
"Having fun and you?", "I'm good, how about
you?"]
        },
        {
            keywords: ["bye", "goodbye", "take
care"],
            responses: ["Bye. Talk to you soon.",
"Goodbye!", "Have fun."]
        },
        {
            keywords: ["interests", "career"],
            responses: ["Computer Science is fun.",
"Programming is lots of fun.", "I like
programming a lot."]
        }
```

```
    ];

    /* if no keywords are found, use these
responses */
    let randomResponses =
    [
        "That's interesting!",
        "Would you tell me more?",
        "Hmm, tell me more about that.",
        "Would you elaborate?",
        "Interesting, please go on.",
        "Fascinating! Tell me more.",
    ];

    function createRobot()
    {
        let context = null;

        while (true)
        {
```

```
        let userInput = prompt("Ask me a
question:");

        if (!userInput)
        {
            break;
        }

        let responseObj =
responses.find(function(obj)
        {
            return
obj.keywords.some(function(keyword) {
                return
userInput.toLowerCase().indexOf(keyword) !== -
1;
            });
        });

        /* if keyword is found */
```

```
        if (responseObj)
        {
            let randomIndex =
Math.floor(Math.random() *
responseObj.responses.length);

    alert(responseObj.responses[randomIndex]);

            if
(responseObj.keywords.indexOf("bye") !== -1)
            {
                break;
            }

            context = responseObj.keywords;
        }
        else
        {
            let result = calculate(userInput);
```

```
            if (result !== null)
            {
                alert("The result is: " + result);
            }
            else
            {
                let randomIndex =
Math.floor(Math.random() *
randomResponses.length);

            alert(randomResponses[randomIndex]);
            }
        }
    }
}

function calculate(input)
{
    try
    {
```

```
            return eval(input);
        }
        catch (error)
        {
            return null;
        }
    }

    createRobot();

}());
```

```
javascript:(
/* Robot AI - Textbox for Input with Send Button - Random Responses every time to keywords and phrases and variations of keywords and phrases found. Other random response when no keywords are found - and Calculations*/
function()
{
    let mainDiv = document.createElement("div");
    mainDiv.style.position = "fixed";
    mainDiv.style.top = "0";
    mainDiv.style.right = "0";
    mainDiv.style.padding = "5px";
    mainDiv.style.backgroundColor = "rgb(0, 0, 0)";
    mainDiv.style.color = "rgb(255, 255, 255)";
    mainDiv.style.zIndex = "10000";
    document.body.append(mainDiv);
```

```
    let userInput =
document.createElement("input");
    userInput.type = "text";
    userInput.placeholder = "Type Words Here";
    userInput.style.width = "200px";
    userInput.style.paddingLeft = "10px";
    userInput.style.paddingRight = "10px";
    userInput.style.paddingTop = "4px";
    userInput.style.paddingBottom = "4px";
    mainDiv.append(userInput);

    let sendButton =
document.createElement("button");
    sendButton.textContent = "Send";
    mainDiv.append(sendButton);

    /* if keyword is found, use these responses */
    let responses =
    [
        {
```

```
        keywords: ["hi", "howdy", "hey"],
        responses: ["Howdy", "Hi there", "Hi"]
    },
    {
        keywords: ["weather", "forecast",
"sunny", "cloudy"],
        responses: ["It's a beautiful day.", "The
weather is very nice.", "It's sunny and warm."]
    },
    {
        keywords: ["how are you", "what's up",
"what are you up to?"],
        responses: ["I'm doing good, you?",
"Having fun and you?", "I'm good, how about
you?"]
    },
    {
        keywords: ["bye", "goodbye", "take
care"],
```

```
            responses: ["Bye. Talk to you soon.",
"Goodbye!", "Have fun."]
        },
        {
            keywords: ["interests", "career"],
            responses: ["Computer Science is fun.",
"Programming is lots of fun.", "I like
programming a lot."]
        }
    ];

    /* if no keywords are found, use these
responses */
    let randomResponses =
    [
        "That's interesting!",
        "Would you tell me more?",
        "Hmm, tell me more about that.",
        "Would you elaborate?",
        "Interesting, please go on.",
```

```
        "Fascinating! Tell me more.",
    ];

    let context = null;

    /* we make it so that the Enter key on the
keyboard will trigger a click on the send button
*/
    userInput.onkeydown = function(event)
    {
        if (event.key === "Enter")
        {
            /* prevent default Enter key behavior */
            event.preventDefault();

            /* trigger a click on the send button */
            sendButton.click();
        }
    };
```

```
    sendButton.onclick = function()
    {
        const userQuestion =
userInput.value.trim();

        if (userQuestion)
        {
            let responseObj =
responses.find(function(obj)
            {
                return obj.keywords.some(function
(keyword)
                {
                    return
userQuestion.toLowerCase().indexOf(keyword) !
== -1;
                });
            });

            if (responseObj)
```

```
        {
            let randomIndex =
Math.floor(Math.random() *
responseObj.responses.length);

    alert(responseObj.responses[randomIndex]);

            if
(responseObj.keywords.indexOf("bye") !== -1)
            {
                context = null;
            }
            else
            {
                context = responseObj.keywords;
            }
        }
        else
        {
            let result = calculate(userQuestion);
```

```
            if (result !== null)
            {
                alert("The result is: " + result);
            }
            else
            {
                let randomIndex =
Math.floor(Math.random() *
randomResponses.length);

            alert(randomResponses[randomIndex]);
            }
        }

        userInput.value = "";
    }
};

function calculate(input)
```

```
    {
        try
        {
            return eval(input);
        }
        catch (error)
        {
            return null;
        }
    }

}());
```

Type Words Here | Send

/* and diagram */

/* or diagram */

Christopher Topalian

/* and or diagram */

Christopher Topalian

True Artificial Intelligence System

16-Gon

For More Tutorials:

CollegeOfScripting.weebly.com

CollegeOfScripting.wordpress.com

GitHub.com/ChristopherTopalian

Youtube.com/ScriptingCollege

Twitter.com/CollegeOfScript

sites.google.com/view/CollegeOfScripting

Dedicated to God the Father

This book is created by the
College of Scripting Music & Science
Always remember, that each time you write a script with a pencil and paper, it becomes imprinted so deeply in memory that the material and methods are learned extremely well. When you Type the scripts, the same is true. The more you type and write out the scripts by keyboard or pencil and paper, the more you will learn programming!
Write and Type EVERY example that you find.
Keep all of your scripts organized.
Every script that you create increases your programming abilities.
SEEING CODE, is one thing,
but WRITING CODE is another.
Write it, Type it, Speak It, See It, Dream It.
www.CollegeOfScripting.weebly.com