LEBANON HIGH SCHOOL
COMPUTER TECHNOLOGY

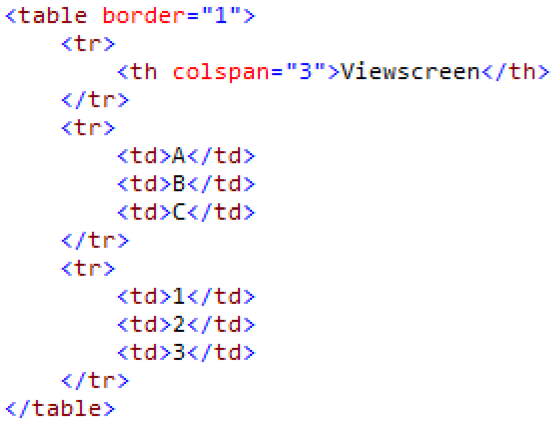
CrazyScript Calculator

Step 1:

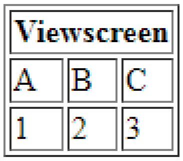
Create a New Site in Expression Web. It will be fastest if you create a Single Page Site.

Step 2:

Create a table to represent the calculator layout. Here is some example code and its visual result:



```
<table border="1">
    <tr>
        <th colspan="3">Viewscreen</th>
    </tr>
    <tr>
        <td>A</td>
        <td>B</td>
        <td>C</td>
    </tr>
    <tr>
        <td>1</td>
        <td>2</td>
        <td>3</td>
    </tr>
</table>
```



Viewscreen		
A	B	C
1	2	3

Step 3:

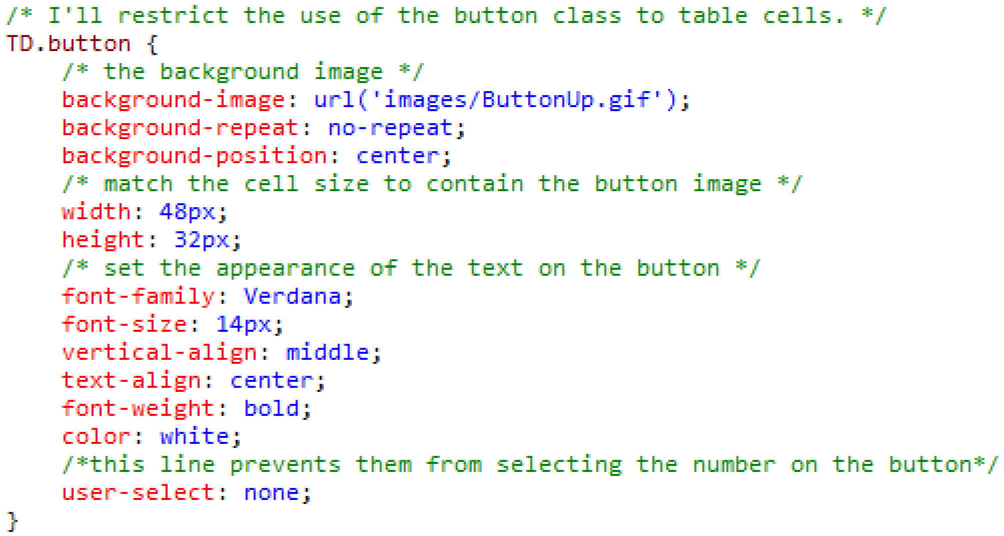
Create an images folder in your site. Find or create a button image to use for each key of your calculator.

You'll need both an **UP** and **DOWN** version of your button image.

NEXT PAGE →

Step 4:

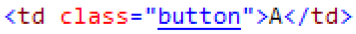
Add a stylesheet (e.g. styles.css) and then add a class called “button”. Since the “button” class will only be used in table cells, I’ll restrict it to the TD tag.



```
/* I'll restrict the use of the button class to table cells. */
TD.button {
    /* the background image */
    background-image: url('images/ButtonUp.gif');
    background-repeat: no-repeat;
    background-position: center;
    /* match the cell size to contain the button image */
    width: 48px;
    height: 32px;
    /* set the appearance of the text on the button */
    font-family: Verdana;
    font-size: 14px;
    vertical-align: middle;
    text-align: center;
    font-weight: bold;
    color: white;
    /*this line prevents them from selecting the number on the button*/
    user-select: none;
}
```

Note: *user-select only works with modern browsers such as Google chrome.*

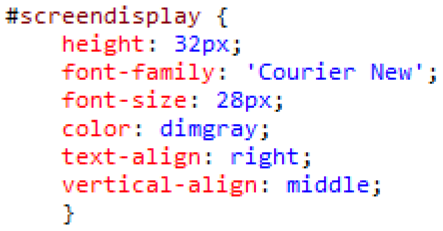
For each cell that will contain a button, assign it the class “button”.



```
<td class="button">A</td>
```

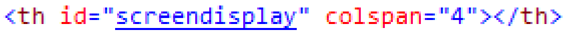
Step 5:

In your stylesheet, add an ID selector called “screendisplay”. Set the properties of your calculator’s display to your liking.



```
#screendisplay {
    height: 32px;
    font-family: 'Courier New';
    font-size: 28px;
    color: dimgray;
    text-align: right;
    vertical-align: middle;
    }
```

Be sure to set the display cell on your calculator to have the matching ID.

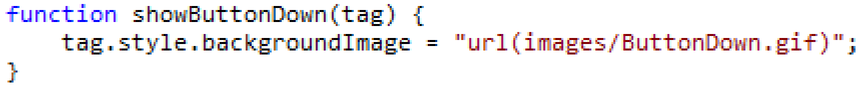


```
<th id="screendisplay" colspan="4"></th>
```

NEXT PAGE →

Step 6:

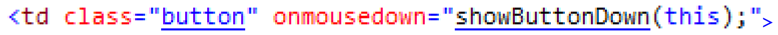
Now it's time to make the button's give visual feedback when pressed. Here is a Java function that would swap the background images:



```
function showButtonDown(tag) {
    tag.style.backgroundImage = "url(images/ButtonDown.gif)";
}
```

You'll want to place this code in the <script> tag inside the <head> of the document.

Be sure to tell each button when to do this:

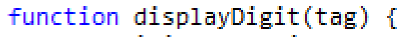


```
<td class="button" onmousedown="showButtonDown(this);">
```

You'll also need to tell the button to switch back when the mouse is released. Add a new function called showButtonUp() and then use the onmouseup event to tell it when to fire.

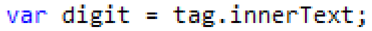
Step 7:

You'll need to write a new function that allows you to send it the current tag.



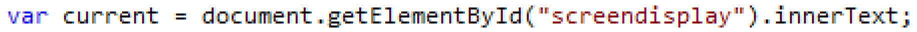
```
function displayDigit(tag) {
```

You can display a digit in the viewscreen by first accessing and storing the *innerText* of the current tag.



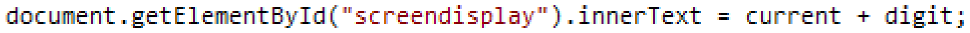
```
var digit = tag.innerText;
```

You'll want to keep track of what is already in the calculator screen.



```
var current = document.getElementById("screendisplay").innerText;
```

Then, you can update the screen.



```
document.getElementById("screendisplay").innerText = current + digit;
```

Don't forget to close your function at the end.

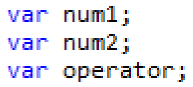
Step 8:

Write a function called clearDisplay() to empty the contents of the calculator screen. You can use the value "" to display empty text. You'll need to use the *getElementById()* function for this.

NEXT PAGE →

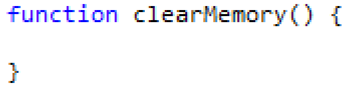
Step 9:

The calculator needs to keep track of two numbers and an operator. Add variables for these at the top of the script.



```
var num1;
var num2;
var operator;
```

You'll want to also create a function called *clearMemory()* that sets all of these equal to null and clears the display.

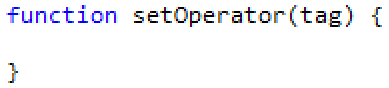


```
function clearMemory() {

}
```

Step 10:

Now it's time to do make the calculator function. You'll need two functions for this. The first is *setOperator()*.

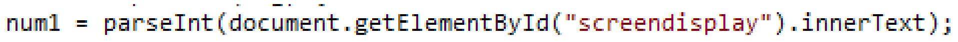


```
function setOperator(tag) {

}
```

This function should:

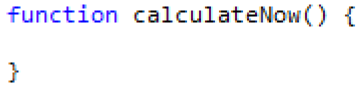
- Save the text in the calculator screen to the *num1* variable:



```
num1 = parseInt(document.getElementById("screendisplay").innerText);
```

- Save the operator that was just clicked to the *operator* variable.
  (**Hint:** use *tag.innerText* to help you do this).
- Clear the display

You'll also need a function to do the calculations. We'll call this *calculateNow()*.

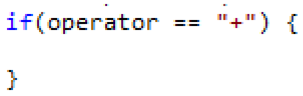


```
function calculateNow() {

}
```

This function should:

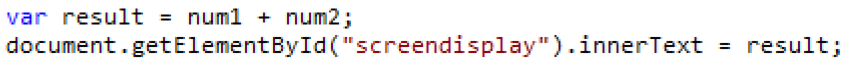
- Save the text in the calculator screen to the *num2* variable:
- See what operator was used:



```
if(operator == "+") {

}
```

- Calculate and display the result. For addition, this would be:



```
var result = num1 + num2;
document.getElementById("screendisplay").innerText = result;
```

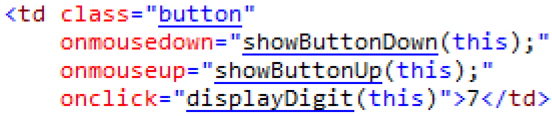
- Set all of the variables back to *null* so you can start over.

NEXT PAGE →

Step 11:

We have some functions now, but we need to tell the page when to execute them!

One of the button cells might look like this when you are finished:



```
<td class="button"
    onmousedown="showButtonDown(this);"
    onmouseup="showButtonUp(this);"
    onclick="displayDigit(this)">7</td>
```

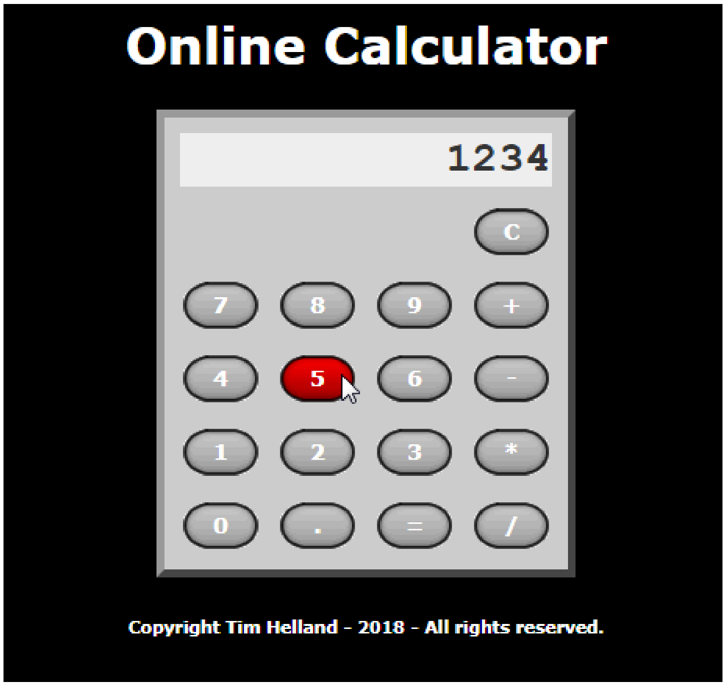
This is formatted to emphasize the parts. Feel free to write it all on one line.

What will be different in the C (clear) button?

When you are all done, you're calculator should allow you to **add** and **multiply**.
*Add in **subtraction**, **divide**, and **square root** to improve your overall score or earn extra credit.*

Step 12:

Make it look good! An example of a possible finished product is shown below.



Online Calculator

Copyright Tim Helland - 2018 - All rights reserved.