Lab 08 Javascript 1

General Information

What you should do

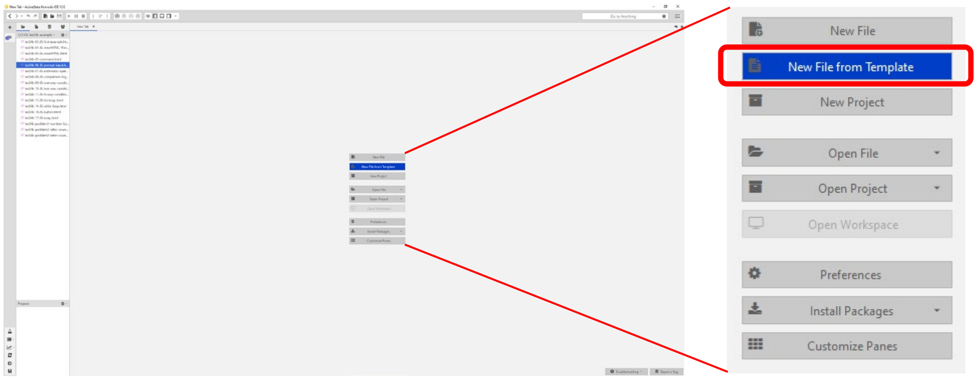
• You should first review Lecture 4 (Part B) up to slide 7 and be familiar with the concepts on Javascript as well as the related code examples so that you can refer to the specific techniques from the corresponding slides when completing the tasks in this lab.
• You should try to come up with the code yourself as much as possible. Do not be afraid of making mistakes, since debugging (finding out where your code goes wrong and fixing it) is part of the learning process.
• We do not give out model programs to the exercises. There can be multiple ways to write the code that solves the same problem. It is important that you build up the program logic yourself instead of merely looking at some code that you do not understand. At any time if you are lost or if you have any questions, feel free to ask the instructor, tutor, or teaching assistant and we will be very happy to help you.

Self-Discovery

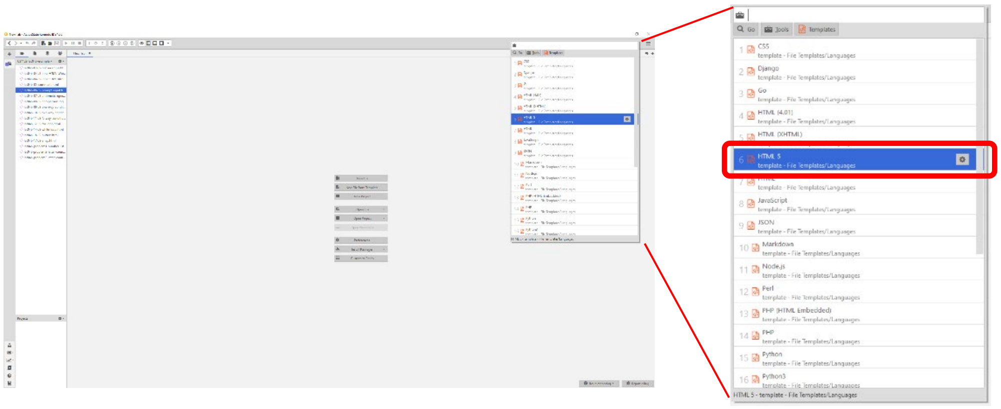
• Most lab tasks are designed to be relatively simple such that you can take the time to think about the related underlying concepts. Besides, we also encourage you to discover things on your own which may not be specified in the tasks.

Task 1.1 Create a New HTML File from Template in Komodo Edit

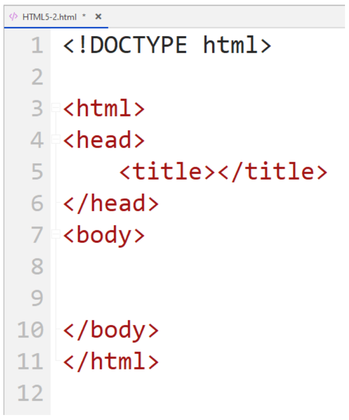
Open Komodo Edit and select "New File from Template" (or press the shortcut keys Ctrl+Shift+N):



A dialog box will pop up on the right. Select "HTML 5" from the list:



You will then see that a new file has been created for you, with some HTML tags that define the basic structure of this HTML webpage.


HTML5-2.html

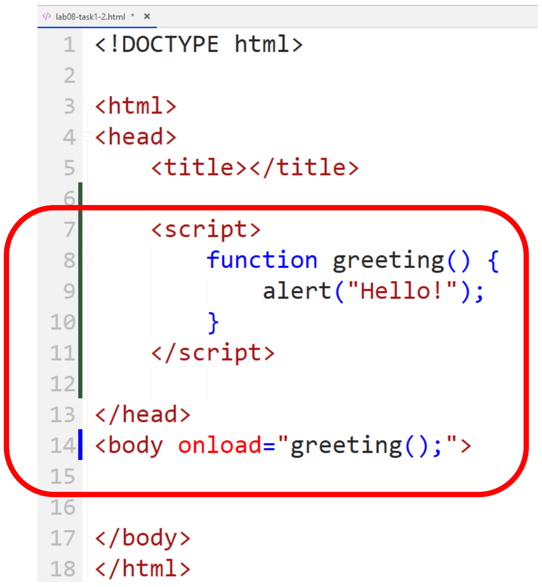
```
<!DOCTYPE html>

<html>
<head>
    <title></title>
</head>
<body>


</body>
</html>

```

You can then add the HTML content and Javascript code on this file. Remember to save the file after you make changes before opening it on a browser to view it.

Note that you should NOT copy and paste any code from this document to your code editor. Type up the code instead, otherwise it may not work properly, especially with symbols such as quotation marks " ".

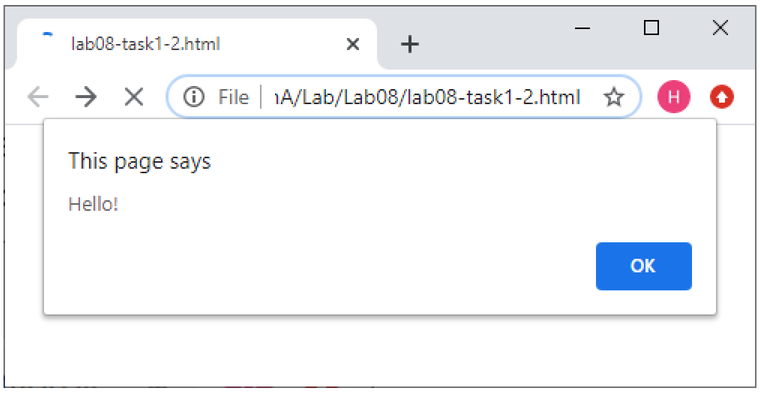
Task 1.2 Output a Message on a Pop-up Window

Save the HTML file created in Task 1.1 as `lab08-task1-2.html`. Add the Javascript code in the head section and the HTML content in the body section as shown in the following screenshot:


lab08-task1-2.html

```
<!DOCTYPE html>

<html>
<head>
    <title></title>

    <script>
        function greeting() {
            alert("Hello!");
        }
    </script>

</head>
<body onload="greeting();">


</body>
</html>
```

You can go to the File Folder where this file is saved and double click on it to open this HTML file on a browser. You will then see that a pop-up window will show up with the text "Hello!":

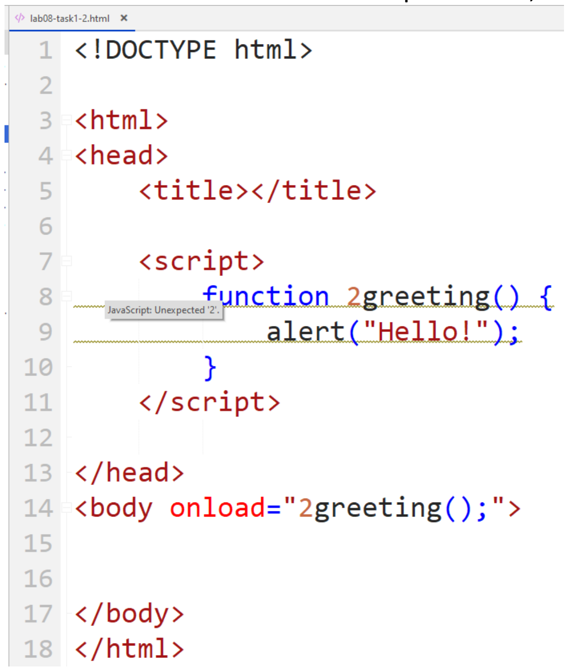


```
This page says
Hello!
```

The onload event handler is invoked when the browser has finished loading the webpage. It will execute the Javascript code defined inside the quotation marks, i.e., `greeting();` which will call the function `greeting()`.
The function `greeting()` is a user-defined function, meaning that it is a function defined by the programmer. The programmer can use a function name (`greeting` in the above example) as long as it satisfies the following requirements:

1) The function name must begin with letters (`A-Z`, `a-z`), underscore _ , or dollar sign `$`, but must NOT begin with digits (`0-9`)
2) The 2nd or any subsequent letter of the function name can be letters (`A-Z`, `a-z`), underscore _ , dollar sign `$`, or digits (`0-9`)
3) The user-defined function name cannot be the same as reserved words, e.g., `var`, `alert`. The list of Javascript reserved words can be found on this webpage: https://www.w3schools.com/js/js_reserved.asp

Also note that the function name is case sensitive, meaning that `greeting()` and `Greeting()` would refer to 2 different functions.
If you change the function name from `greeting()` to `2greeting()` as shown below and save the file, you would find that the webpage would not work properly, i.e., no pop-up window will show up any more because the Javascript code has error. You should also see that in Komodo Edit, the code in line 8 and line 9 are underlined. If you move your mouse over the underline on line 8, then you would see a message "`Javascript: Unexpected '2'.`" which shows an error that the code editor detects in your Javascript code. Since line 8 has error, the code editor would not be able to understand this line, which makes it unable to understand the subsequent code, and it explains why line 9 is also underlined.



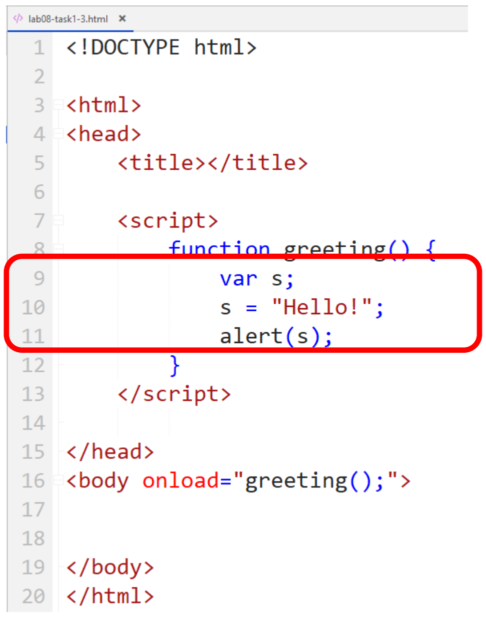
```
1  <!DOCTYPE html>
2
3  <html>
4  <head>
5      <title></title>
6
7      <script>
8          function 2greeting() {
9              alert("Hello!");
10         }
11     </script>
12
13 </head>
14 <body onload="2greeting();">
15
16
17 </body>
18 </html>
```

Exercise 1.2:

a) Change the message from "Hello!" to another greeting message of your choice. Open the webpage again to verify that your new greeting message shows up in the pop-up window.
b) Change the function name to another valid function name of your choice. Carry out the same change for calling this new function name in the onload event handler. Open the webpage again to verify that it still works in the same way as before.

Task 1.3 Variables

Save your HTML file as a new file as `lab08-task1-3.html`. Modify the Javascript code in the function `greeting()` as shown in the following screenshot:



```
 1  <!DOCTYPE html>
 2
 3  <html>
 4  <head>
 5      <title></title>
 6
 7      <script>
 8          function greeting() {
 9              var s;
10              s = "Hello!";
11              alert(s);
12          }
13      </script>
14
15  </head>
16  <body onload="greeting();">
17
18
19  </body>
20  </html>
```

Line 9: `var s;`

It declares a variable `s`. The requirement for a variable name is the same as the requirement for a user-defined function name stated in Task 1.2, i.e.,

1) The variable name must begin with letters (`A-Z`, `a-z`), underscore `_` , or dollar sign `$`, but must NOT begin with digits (`0-9`)
2) The 2nd or any subsequent letter of the variable name can be letters (`A-Z`, `a-z`), underscore `_` , dollar sign `$`, or digits (`0-9`)
3) The variable name cannot be the same as reserved words, e.g., `var`, `alert`. The list of Javascript reserved words can be found on this webpage: https://www.w3schools.com/js/js_reserved.asp

Also note that the variable name is case sensitive, meaning that `a` and `A` would refer to 2 different variables.

Line 10: `s = "Hello!";`

It assigns a string `"Hello!"` as the value of the variable `s`. A string can be considered as text which is composed of a sequence of characters where each character can be an alphabet (`A-Z`, `a-z`), but a character can also be a digit (`0-9`) or a symbol (`$`, `&`, `!`, etc.). To represent a string in Javascript, the text is enclosed by quotation marks (either double or single, i.e., either `"Hello!"` or `'Hello!'`).

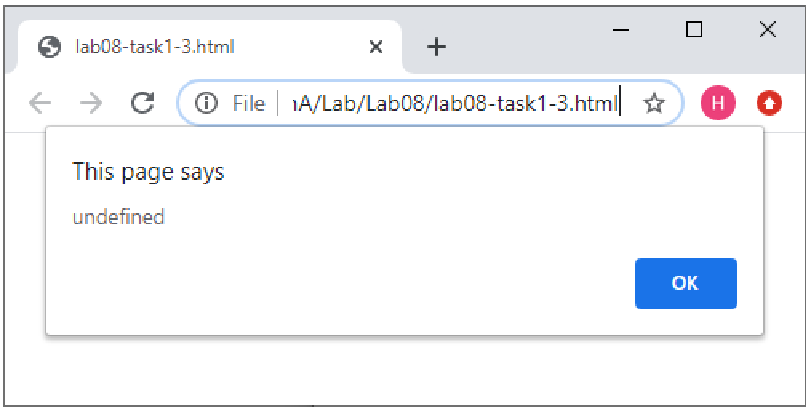
Line 11: `alert(s);`

The Javascript built-in function `alert()` will take the value of the variable `s`, i.e., the string `"Hello!"` that was assigned in line 10 and show it on a pop-up window in the same way as your code in Task 1.2.

What would happen if you have declared a variable but has not assigned any value to it but makes use of this variable? Remove line 10, i.e., the function `greeting()` only has the following 2 lines of code:

```
var s;
alert(s);
```

Open the resulting webpage to see what get displayed on the pop-up window.

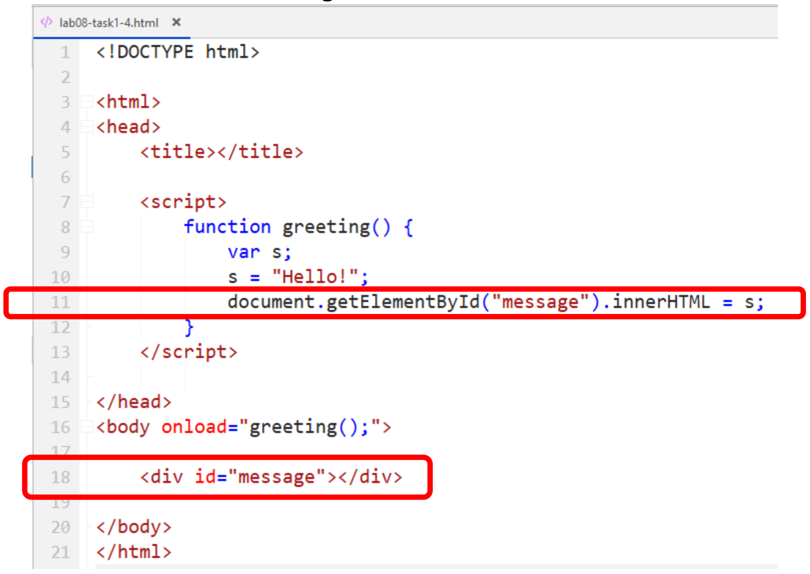


```
This page says
undefined
```

You will see that the pop-up window shows `undefined` which is the value of `s`. It is important to make sure that you have assigned a variable with the desired value before using it for further computation.

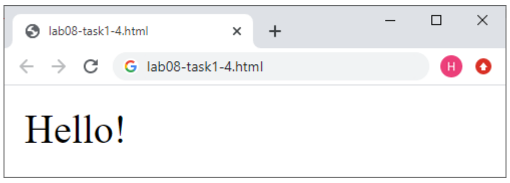
Task 1.4 Change HTML Content

Save your HTML file as a new file as `lab08-task1-4.html`. Modify the Javascript code and HTML code as shown in the following screenshot:



```
1   <!DOCTYPE html>
2
3   <html>
4   <head>
5       <title></title>
6
7       <script>
8           function greeting() {
9               var s;
10              s = "Hello!";
11              document.getElementById("message").innerHTML = s;
12          }
13      </script>
14
15  </head>
16  <body onload="greeting();">
17
18      <div id="message"></div>
19
20  </body>
21  </html>
```

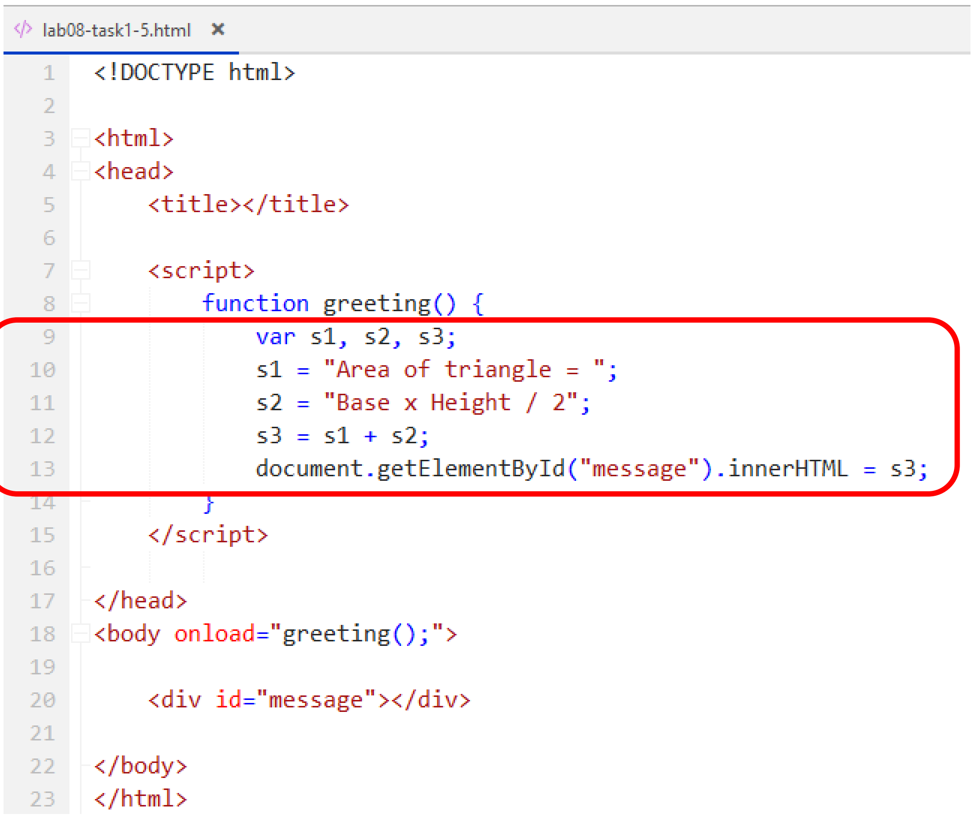
The HTML code `<div id="message"></div>` creates a new div element but it has no content inside these tags. When the Javascript function `greeting()` is called, line 11 `document.getElementById("message").innerHTML = s` will take the value of `s` which is the string `"Hello!"` and use it to replace the current content of that `div` element specified with its ID `"message"`. If you open the HTML file on the browser, the resulting webpage will be shown as follows:



Hello!

Task 1.5 String Concatenation

Save your HTML file as a new file as `lab08-task1-5.html`. Modify the Javascript code as shown in the following screenshot:



```
 1  <!DOCTYPE html>
 2
 3  <html>
 4  <head>
 5      <title></title>
 6
 7      <script>
 8          function greeting() {
 9              var s1, s2, s3;
10              s1 = "Area of triangle = ";
11              s2 = "Base x Height / 2";
12              s3 = s1 + s2;
13              document.getElementById("message").innerHTML = s3;
14          }
15      </script>
16
17  </head>
18  <body onload="greeting();">
19
20      <div id="message"></div>
21
22  </body>
23  </html>
```

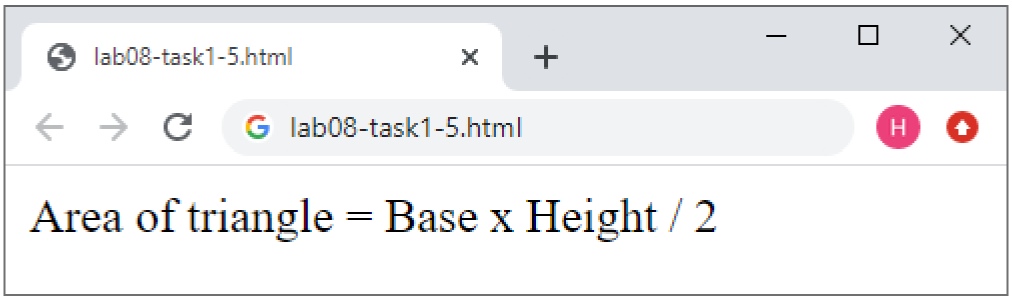
Line 9: `var s1, s2, s3;`

You can declare more than 1 variable with a single statement. The variable names to be declared should be separated using commas (,).

Line 12: `s3 = s1 + s2;`

The value of `s1` is a string defined in line 10 and the value of `s2` is also a string defined in line 11. Since the operands of the `+` operator in line 12, i.e., `s1` and `s2` are both strings, `s1 + s2` will concatenate these strings together.

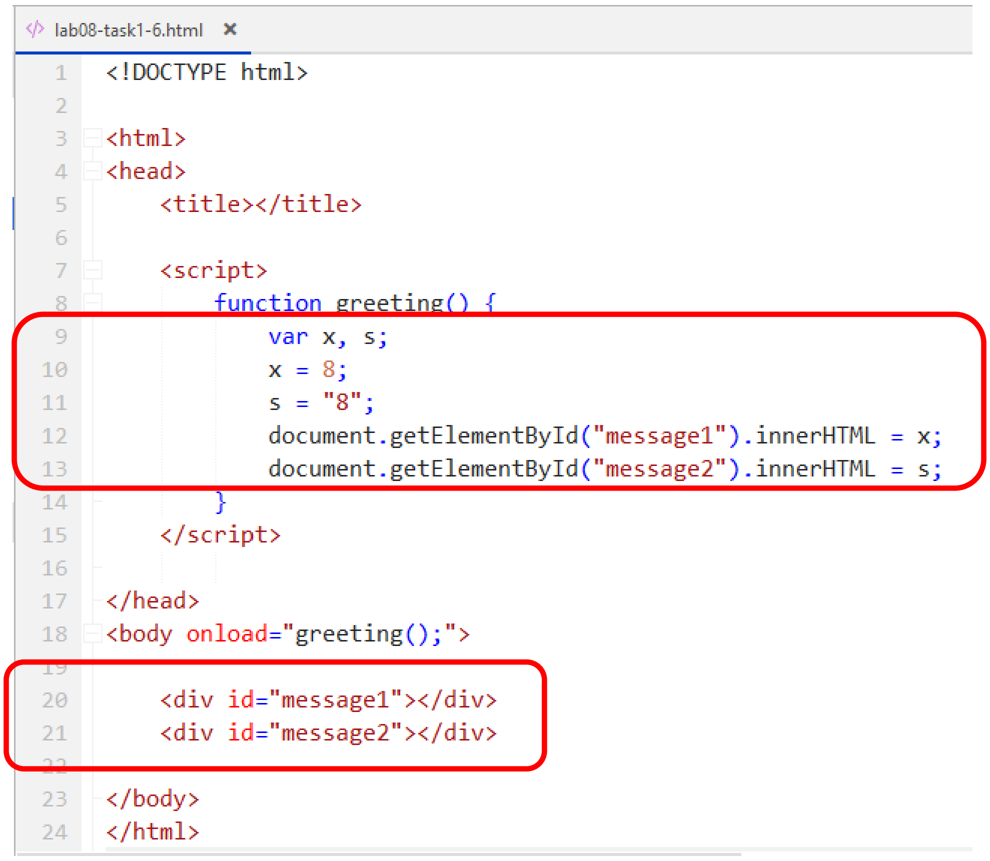
Open the HTML file on a browser and you will see the following result:



```
Area of triangle = Base x Height / 2
```

Task 1.6 String vs Number

Save your HTML file as a new file as `lab08-task1-6.html`. Modify the Javascript code and HTML code as shown in the following screenshot:



lab08-task1-6.html

```
 1  <!DOCTYPE html>
 2
 3  <html>
 4  <head>
 5      <title></title>
 6
 7      <script>
 8          function greeting() {
 9              var x, s;
10              x = 8;
11              s = "8";
12              document.getElementById("message1").innerHTML = x;
13              document.getElementById("message2").innerHTML = s;
14          }
15      </script>
16
17  </head>
18  <body onload="greeting();">
19
20      <div id="message1"></div>
21      <div id="message2"></div>
22
23  </body>
24  </html>
```

Line 10: `x = 8;`

The variable `x` is assigned with the number 8.

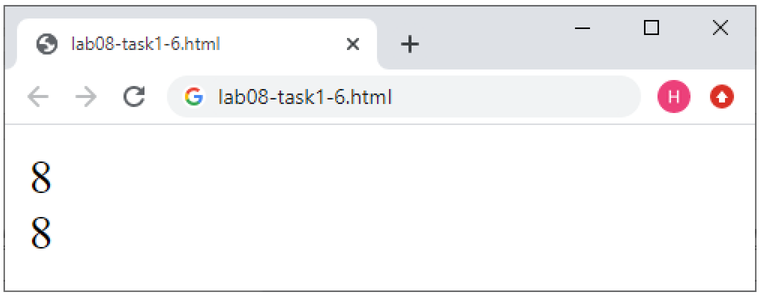
Line 11: `s = “8”;`

The variable `s` is assigned with the string “8”.

Line 12: `document.getElementById(“message1”).innerHTML = x;`

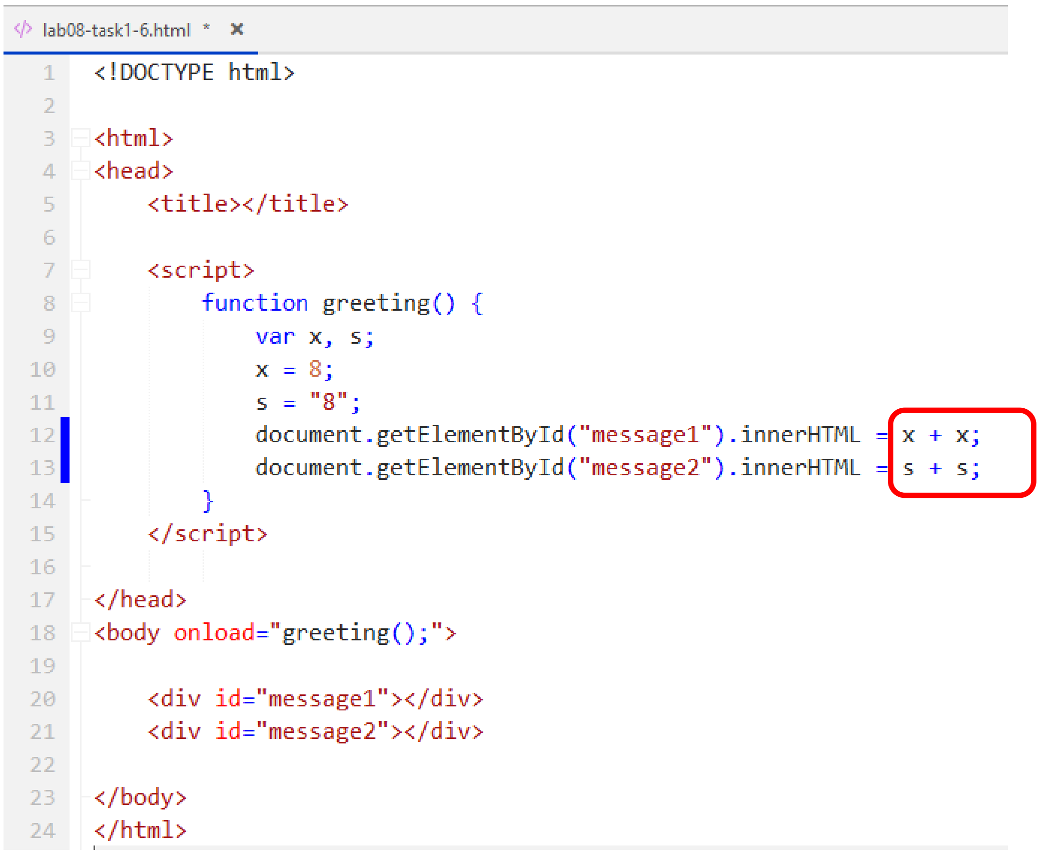
The left hand side `document.getElementById(“message1”).innerHTML` is expecting a string, and the expression on the right hand side will be evaluated and then converted to a string.

Open the HTML file on a browser and you will see the following result:



```
8
8
```

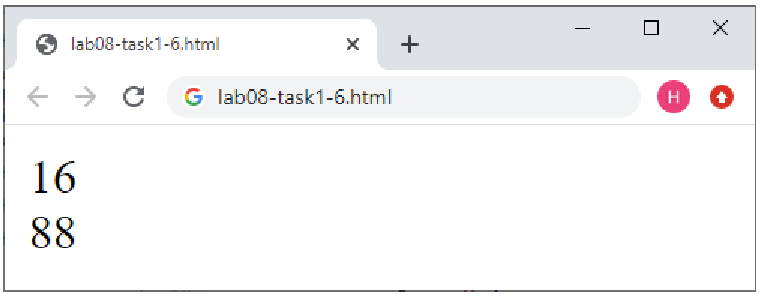
It may appear that there is no difference between a string and a number defined by `s` and `x` respectively. However, the difference becomes clear when you modify the Javascript code as shown in the following screenshot:



lab08-task1-6.html *

```
 1  <!DOCTYPE html>
 2
 3  <html>
 4  <head>
 5      <title></title>
 6
 7      <script>
 8          function greeting() {
 9              var x, s;
10              x = 8;
11              s = "8";
12              document.getElementById("message1").innerHTML = x + x;
13              document.getElementById("message2").innerHTML = s + s;
14          }
15      </script>
16
17  </head>
18  <body onload="greeting();">
19
20      <div id="message1"></div>
21      <div id="message2"></div>
22
23  </body>
24  </html>
```

Open the HTML file on a browser and you will see the following result:



```
16
88
```

Line 12 will first evaluate the expression on the right hand side `x + x`. Since both operands `x` are numbers, the `+` operator will perform arithmetic calculation 8 + 8 and obtain the number 16. Since the left hand side is expecting a string, the number 16 will be converted to a string.

Line 13 will first evaluate the expression on the right hand side `s + s`. Since both operands `s` are strings, the `+` operator will concatenate these strings and obtain the string “88”.

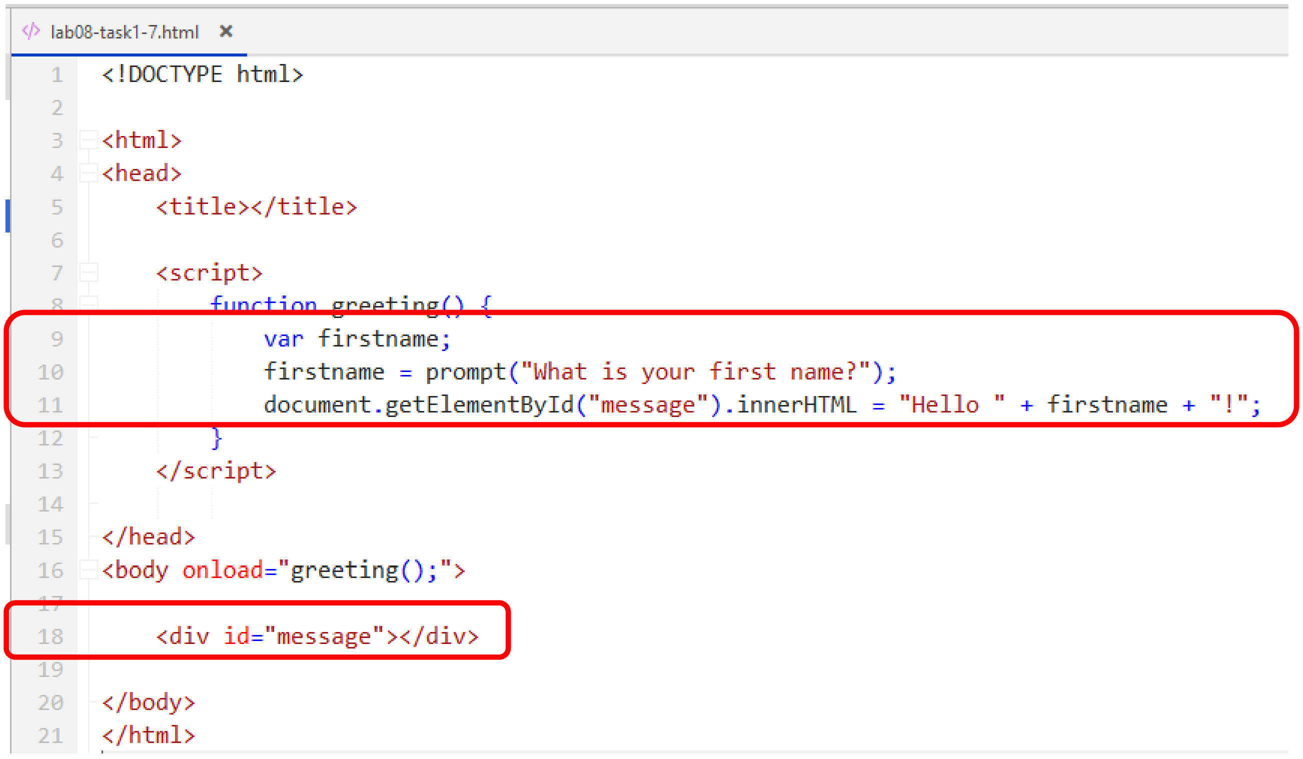
The `+` operator will perform arithmetic addition when both operands are numbers. On the other hand, as long as one of the operands is a string, the `+` operator will treat all operands as strings and perform concatenation.

Exercise 1.6:

a) Replace the right hand side of line 12 with `x + s`. Anticipate what should be displayed in the first line of the webpage and then open the resulting webpage to verify it.
b) Replace the right hand side of line 12 with `s + x`. Anticipate what should be displayed in the first line of the webpage and then open the resulting webpage to verify it.
c) Replace the right hand side of line 12 with `x + x + s`. Anticipate what should be displayed in the first line of the webpage and then open the resulting webpage to verify it.
d) Replace the right hand side of line 12 with `x + (x + s)`. Anticipate what should be displayed in the first line of the webpage and then open the resulting webpage to verify it.
e) Replace the right hand side of line 12 with `x + s + x`. Anticipate what should be displayed in the first line of the webpage and then open the resulting webpage to verify it.

Task 1.7 Getting Input from User

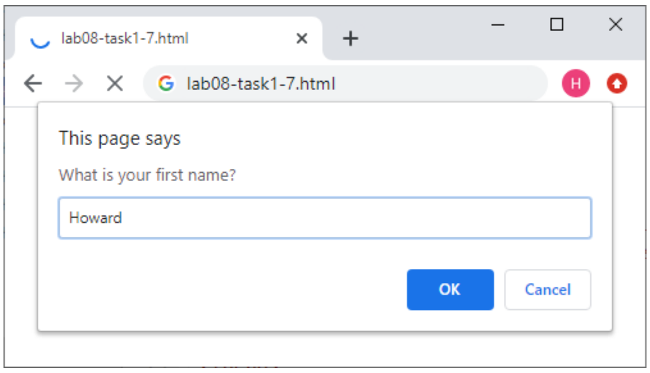
Save your HTML file as a new file as `lab08-task1-7.html`. Modify the Javascript code and HTML code as shown in the following screenshot:



lab08-task1-7.html

```
1   <!DOCTYPE html>
2
3   <html>
4   <head>
5       <title></title>
6
7       <script>
8           function greeting() {
9               var firstname;
10              firstname = prompt("What is your first name?");
11              document.getElementById("message").innerHTML = "Hello " + firstname + "!";
12          }
13      </script>
14
15  </head>
16  <body onload="greeting();">
17
18      <div id="message"></div>
19
20  </body>
21  </html>
```

In line 10, `prompt()` Is a Javascript built-in function. It will show a pop-up window and ask the user to enter something in a text box. The string inside the parentheses, i.e., "`What is the first name?`" will be displayed on the pop-up window to guide the user about what to input.

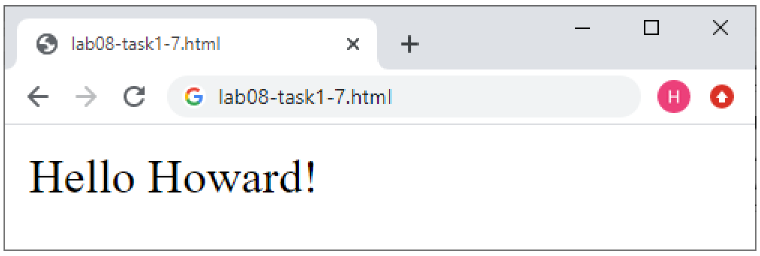


This page says

What is your first name?

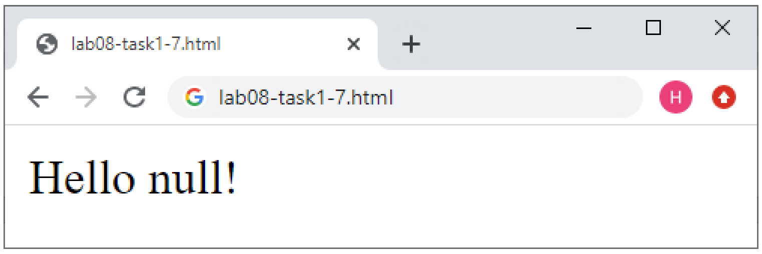
Howard

After the user clicks OK, the following webpage will be displayed on the browser:



Hello Howard!

Now refresh the webpage and enter a name on the text box again. Instead of clicking the OK button, click the Cancel button and you would see the resulting webpage as follows:



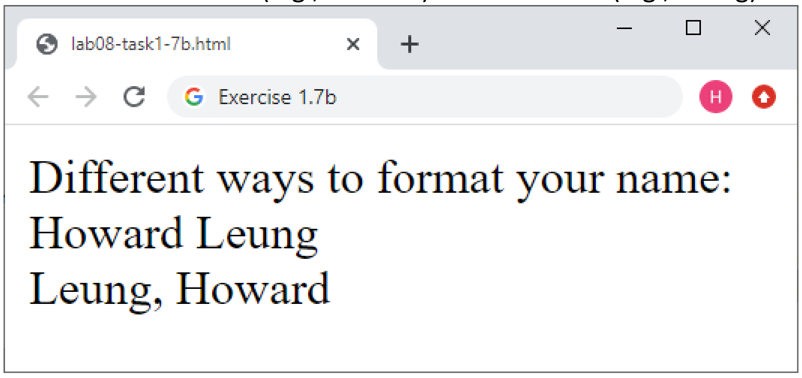
Hello null!

When the cancel button is clicked, `null` will be returned by the function `prompt()` and assigned as the value of the variable `firstname`. The programmer may use this information to check if the user has completed the input process and ask the user to input again (by writing more code).

Exercise 1.7:

a) Modify the Javascript code so that after asking the user to enter the first name, show another pop-up window to ask the user to enter the last name. (Hint: create another variable to store the last name). Show the content "Hello [Last name] [First name]!" on the webpage.

b) Extend your code from a) so that the following content will be shown on the webpage after the user enters the first name (e.g., Howard) and last name (e.g., Leung):

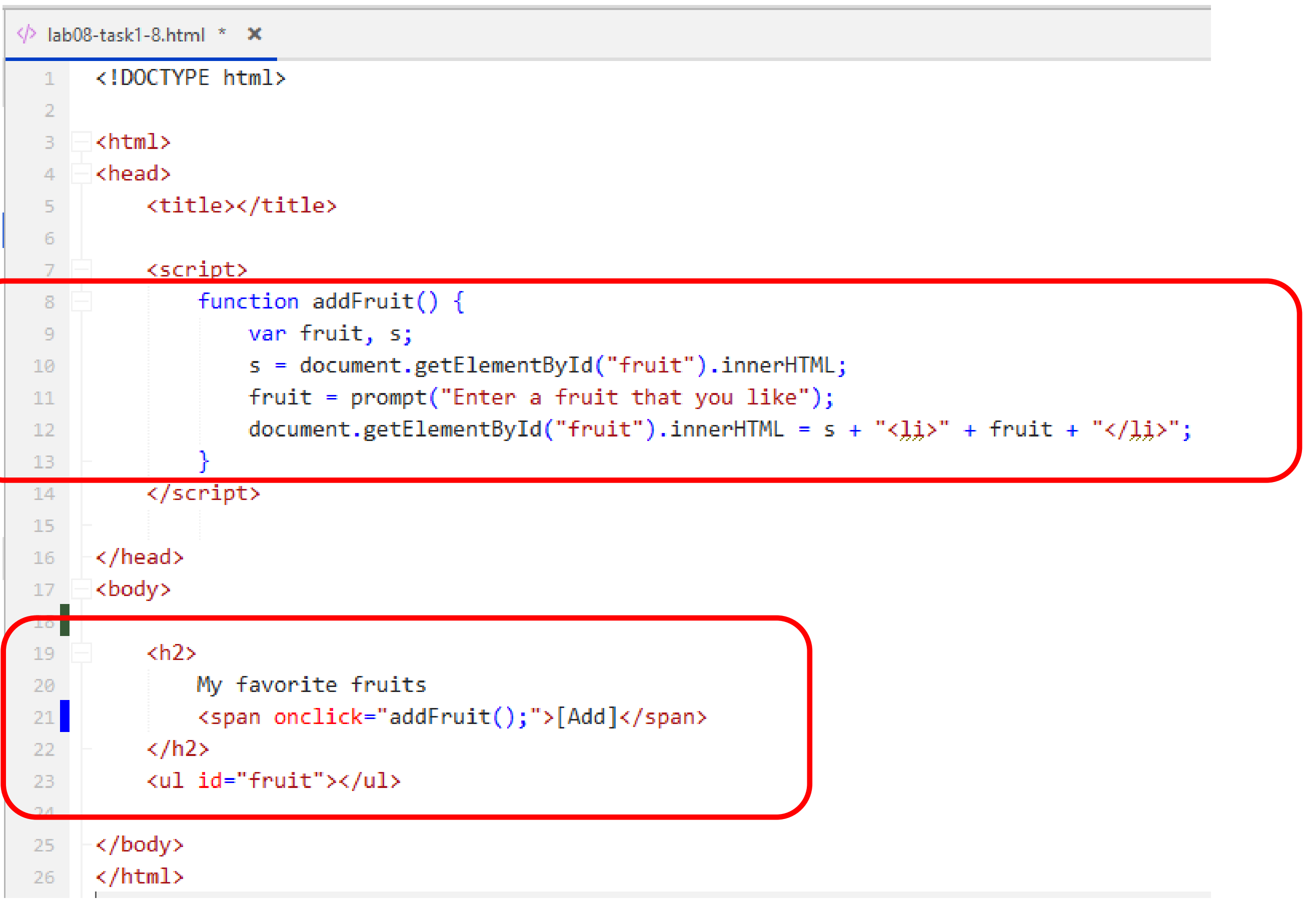


Different ways to format your name:
Howard Leung
Leung, Howard

Hint: on the right hand side of `document.getElementById("message").innerHTML`, add `"<br/>"` to your string where you would like to display with a new line.

Task 1.8 Update Other HTML Element

Save your HTML file as a new file as `lab08-task1-8.html`. Modify the Javascript code and HTML code as shown in the following screenshot:



lab08-task1-8.html *

```
1   <!DOCTYPE html>
2
3   <html>
4   <head>
5       <title></title>
6
7       <script>
8           function addFruit() {
9               var fruit, s;
10              s = document.getElementById("fruit").innerHTML;
11              fruit = prompt("Enter a fruit that you like");
12              document.getElementById("fruit").innerHTML = s + "<li>" + fruit + "</li>";
13          }
14      </script>
15
16  </head>
17  <body>
18
19      <h2>
20          My favorite fruits
21          <span onclick="addFruit();">[Add]</span>
22      </h2>
23      <ul id="fruit"></ul>
24
25  </body>
26  </html>
```

Line 21: `<span onclick="addFruit();">[Add]</span>`

The even handler onclick will be invoked if the user clicks on this `span` element. So if the user clicks on `[Add]`, then the function `addFruit()` will be called.

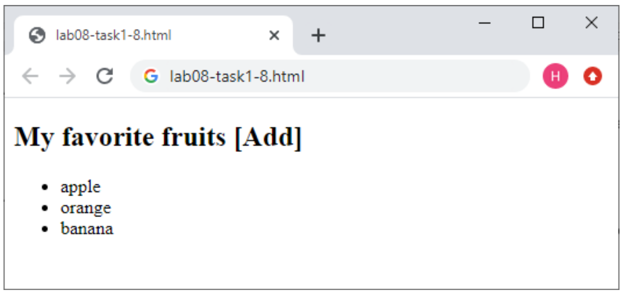
Line 10: `s = document.getElementById("fruit").innerHTML;`

The current HTML content of the element with ID `"fruit"` is extracted and stored with the variable `s`.

Line 12: `document.getElementById("fruit").innerHTML = s + "<li>" + fruit + "</li>";`

The expression `"<li>" + fruit + "</li>"` is the HTML code for creating a new list element with the user input fruit. Line 10 extracts the current HTML of the element with ID `"fruit"` (i.e., all list items in the unordered list) to be stored with the variable `s`. The expression `s + "<li>" + fruit + "</li>"` means adding the newest user input fruit as the last list item, which will then be assigned back to `document.getElementById("fruit").innerHTML` to update the unordered list.

Open the HTML file on a browser. Click [Add] and enter a fruit. Do this a couple of times and you will see that each time your input fruit shows up at the end of the unordered list:

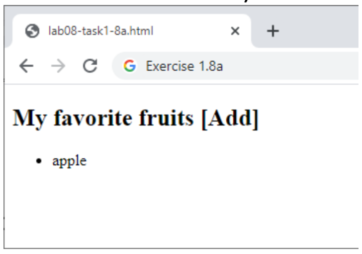


My favorite fruits [Add]

- apple
- orange
- banana

Exercise 1.8:

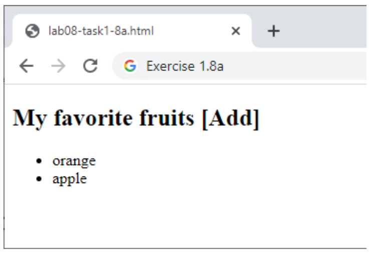
a) Modify the Javascript code so that instead of adding the user input fruit to the end of the unordered list, add it as the first list item. (Hint: change the order of the terms in the expression on the right hand side on line 12).


lab08-task1-8a.html

Exercise 1.8a

My favorite fruits [Add]

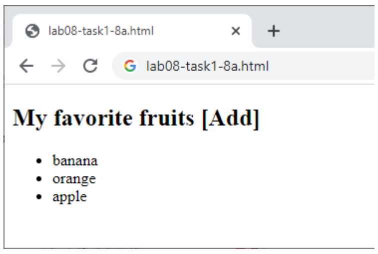
- apple


lab08-task1-8a.html

Exercise 1.8a

My favorite fruits [Add]

- orange
- apple

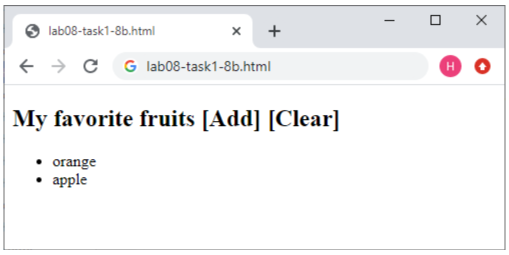

lab08-task1-8a.html

lab08-task1-8a.html

My favorite fruits [Add]

- banana
- orange
- apple

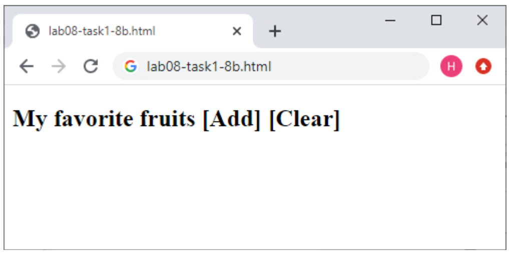
b) Modify the Javascript code so that there is another text [Clear] and when the user clicks on it, it will clear all items in the unordered list.


lab08-task1-8b.html

lab08-task1-8b.html

My favorite fruits [Add] [Clear]

- orange
- apple


lab08-task1-8b.html

lab08-task1-8b.html

My favorite fruits [Add] [Clear]

Hint 1: Add another span element to enclose the text `[Clear]` and add an `onclick` event handler to this span element to call another function defined by you.

Hint 2: Define another function in the `<script>` block to handle the event when `[Clear]` is clicked. In this function, set `document.getElementById("fruit").innerHTML` to the empty string, i.e., "" to clear its content.

```
<script>
      function addFruit(){
            .....................
      }
      function clearFruit(){
            document.getElementById("fruit").innerHTML = "";
      }
</script>
```

Task 2 Review Previous Labs

You can review the previous labs by reading and following the lab instructions, and check that you are able to complete the lab tasks and exercises by yourself. If you have any questions, you can ask the lab tutor or TA during the lab session.

Task 3 Complete the assessment from the Canvas course page

You should complete the Lab 08 Assessment from the Canvas course page before the posted deadline.

Task 4 Challenge your classmates

You can first reflect on what you have learnt in this lab, and then come up with problems to challenge your classmates. You can post your problem on the Canvas course page, under this Discussion page. One should be able to solve your problem by using what he/she learns in Lab 08. You will not get extra marks by posting a challenging problem or solving a challenging problem posted by another student, but you will earn your fame so that you can impress the course leader, the lab tutors, and your classmates.