

Functions

Readability, Reusability & Events



INPUT



BLACK BOX

```
console.log(...)
document.write(...)
document.createElement(...)
element.appendChild(...)
```

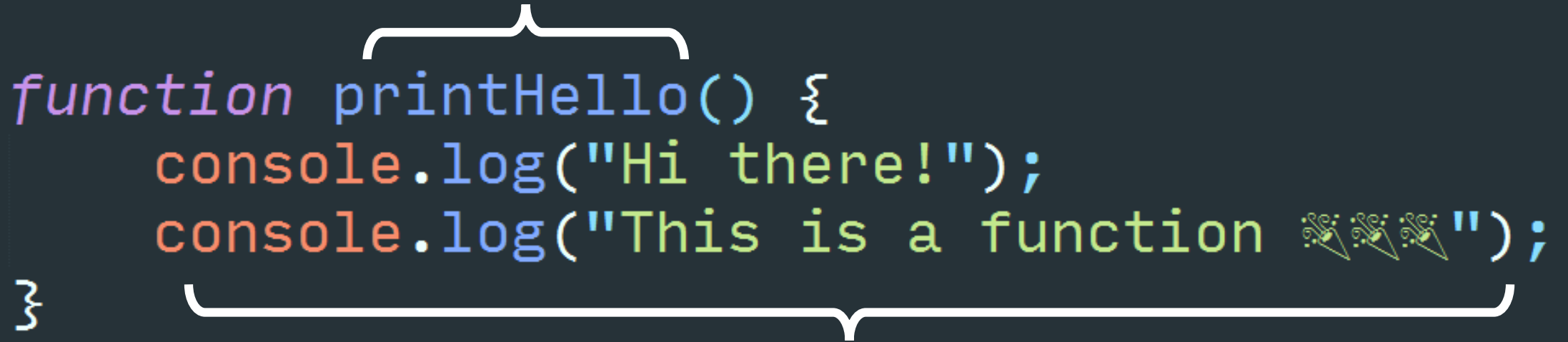


OUTPUT

Basic Functions

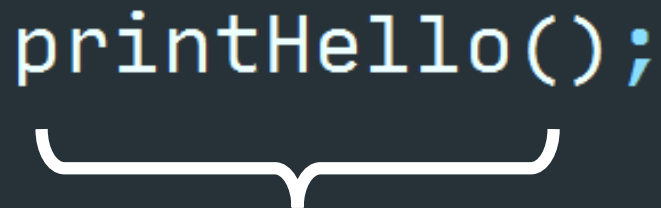
FUNCTION NAME

```
function printHello() {  
    console.log("Hi there!");  
    console.log("This is a function 🐞🐞🐞");  
}
```

A diagram illustrating the components of a JavaScript function definition. A white curly brace is positioned above the function name 'printHello()' in the first line of the code. Another white curly brace is positioned below the function body (the lines between the opening and closing curly braces) in the same code block.

FUNCTION CODE

```
printHello();
```

A diagram illustrating the invocation of a function. A white curly brace is positioned below the function call 'printHello()' in the code.

INVOKING FUNCTION

Functions with Parameters

NAME PARAMETER



A diagram with two white curly braces above the function signature. The first brace is positioned over the word 'welcomeUser' and the second brace is positioned over the parameter '(username)'.

```
function welcomeUser(username) {  
    var h1 = document.createElement("h1");  
    h1.textContent = "Welcome, " + username;  
    var outputDiv = document.querySelector("#output");  
    outputDiv.appendChild(h1);  
}
```

```
welcomeUser("Mike");
```



A diagram with a white curly brace positioned below the argument '"Mike"' in the function call.

ARGUMENT

Functions that Return Something

NAME

PARAMETERS

```
function getAverage(num1, num2, num3) {  
    var sum = num1 + num2 + num3;  
    var average = sum / 3;  
    return average;  
}
```

RETURN STATEMENT

```
var average1 = getAverage(10, 20, 30);  
var average2 = getAverage(60, 20, 100);
```




Function Tracing

1

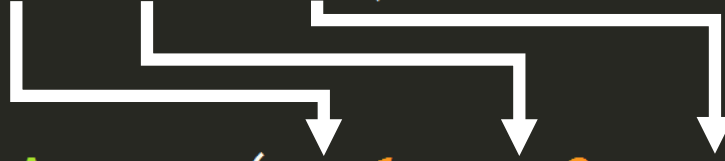
```
printAverage(25, 175, 100);
```

```
function printAverage(num1, num2, num3) {  
    var average = getAverage(num1, num2, num3);  
    printMessage("The average is: " + average + ".");  
}
```

```
function getAverage(n1, n2, n3) {  
    var sum = n1 + n2 + n3;  
    var ave = sum / 3;  
    return ave;  
}
```

```
function printMessage(message) {  
    document.write("<h1>" + message + "</h1>");  
}
```

```
printAverage(25, 175, 100);
```



2

```
function printAverage(num1, num2, num3) {  
    var average = getAverage(num1, num2, num3);  
    printMessage("The average is: " + average + ".");  
}
```

```
function getAverage(n1, n2, n3) {  
    var sum = n1 + n2 + n3;  
    var ave = sum / 3;  
    return ave;  
}
```

```
function printMessage(message) {  
    document.write("<h1>" + message + "</h1>");  
}
```

```
printAverage(25, 175, 100);
```

3

```
function printAverage(num1, num2, num3) {  
    var average = getAverage(num1, num2, num3);  
    printMessage("The average is: " + average + ".");  
}
```

```
function getAverage(n1, n2, n3) {  
    var sum = n1 + n2 + n3;  
    var ave = sum / 3;  
    return ave;  
}
```

```
function printMessage(message) {  
    document.write("<h1>" + message + "</h1>");  
}
```

```
printAverage(25, 175, 100);
```

```
function printAverage(num1, num2, num3) {  
    var average = getAverage(num1, num2, num3);  
    printMessage("The average is: " + average + ".");  
}
```

4

```
function getAverage(n1, n2, n3) {  
    var sum = n1 + n2 + n3;  
    var ave = sum / 3;  
    return ave;  
}
```

```
function printMessage(message) {  
    document.write("<h1>" + message + "</h1>");  
}
```

```
printAverage(25, 175, 100);
```

```
function printAverage(num1, num2, num3) {  
    var average = getAverage(num1, num2, num3);  
    printMessage("The average is: " + average + ".");  
}
```

5

```
function getAverage(n1, n2, n3) {  
    var sum = n1 + n2 + n3;  
    var ave = sum / 3;  
    return ave;  
}
```

```
function printMessage(message) {  
    document.write("<h1>" + message + "</h1>");  
}
```

```
printAverage(25, 175, 100);
```

```
function printAverage(num1, num2, num3) {  
    var average = getAverage(num1, num2, num3);  
    printMessage("The average is: " + average + ".");  
}
```

6

```
function getAverage(n1, n2, n3) {  
    var sum = n1 + n2 + n3;  
    var ave = sum / 3;  
    return ave;  
}
```

```
function printMessage(message) {  
    document.write("<h1>" + message + "</h1>");  
}
```

```
printAverage(25, 175, 100);
```

```
function printAverage(num1, num2, num3) {  
    var average = getAverage(num1, num2, num3);  
    printMessage("The average is: " + average + ".");  
}
```

```
function getAverage(n1, n2, n3) {  
    var sum = n1 + n2 + n3;  
    var ave = sum / 3;  
    return ave;  
}
```

7

```
function printMessage(message) {  
    document.write("<h1>" + message + "</h1>");  
}
```



```
printAverage(25, 175, 100);
```

8

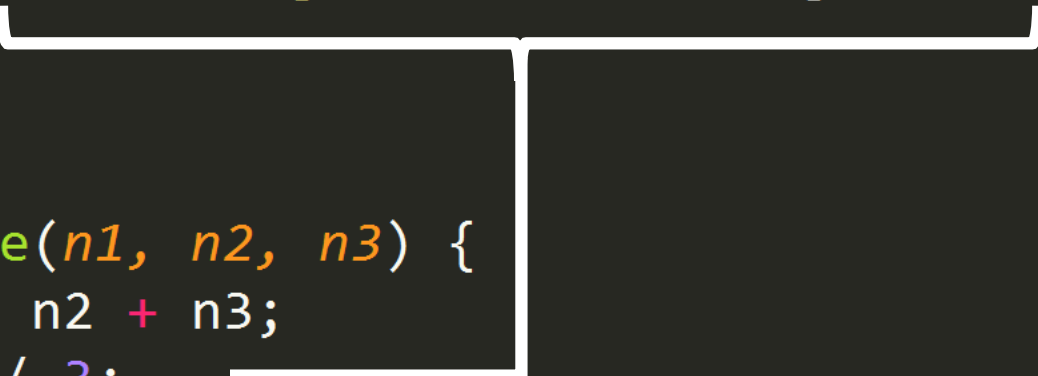
```
function printAverage(num1, num2, num3) {  
    var average = getAverage(num1, num2, num3);  
    printMessage("The average is: " + average + ".");  
}
```

```
function getAverage(n1, n2, n3) {  
    var sum = n1 + n2 + n3;  
    var ave = sum / 3;  
    return ave;  
}
```

```
function printMessage(message) {  
    document.write("<h1>" + message + "</h1>");  
}
```

```
printAverage(25, 175, 100);
```


```
function printAverage(num1, num2, num3) {  
    var average = getAverage(num1, num2, num3);  
    printMessage("The average is: " + average + ".");  
}
```



```
function getAverage(n1, n2, n3) {  
    var sum = n1 + n2 + n3;  
    var ave = sum / 3;  
    return ave;  
}
```

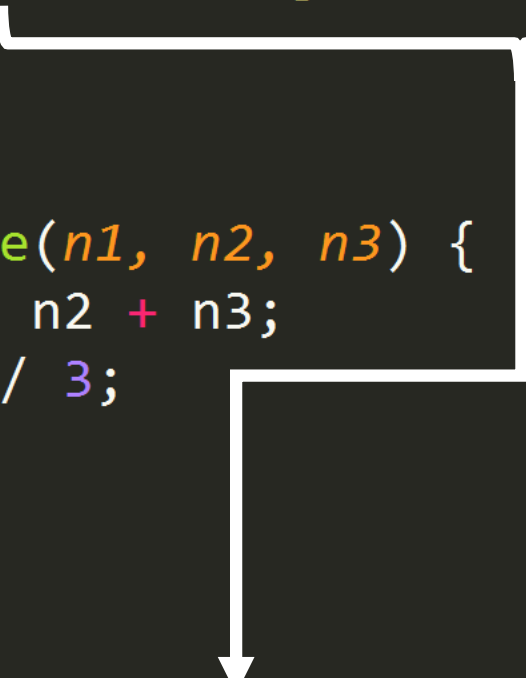
9

```
function printMessage(message) {  
    document.write("<h1>" + message + "</h1>");  
}
```



```
printAverage(25, 175, 100);
```

```
function printAverage(num1, num2, num3) {  
    var average = getAverage(num1, num2, num3);  
    printMessage("The average is: " + average + ".");  
}
```



```
function getAverage(n1, n2, n3) {  
    var sum = n1 + n2 + n3;  
    var ave = sum / 3;  
    return ave;  
}
```

```
function printMessage(message) {  
    document.write("<h1>" + message + "</h1>");  
}
```

1 printAverage(25, 175, 100);

```
2 function printAverage(num1, num2, num3) {  
3     var average = getAverage(num1, num2, num3);  
8     printMessage("The average is: " + average + ".");  
}
```

```
4 function getAverage(n1, n2, n3) {  
5     var sum = n1 + n2 + n3;  
6     var ave = sum / 3;  
7     return ave;  
}
```

```
9 function printMessage(message) {  
10    document.write("<h1>" + message + "</h1>");  
}
```



Invocation vs Reference

```
function welcomeUser() {  
    var message = "Welcome to the site!";  
    alert(message);  
}
```

welcomeUser()



INVOCATION

(runs the function)

welcomeUser

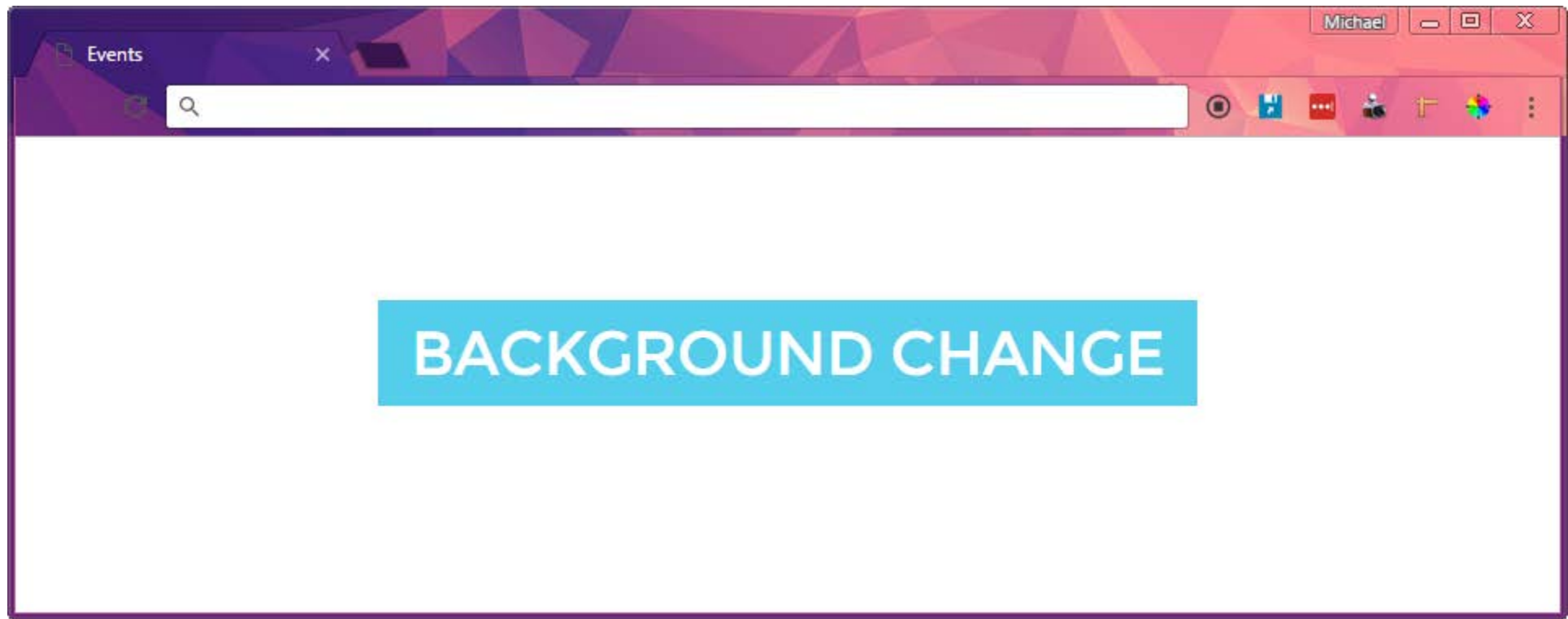


REFERENCE

(does NOT run the function)

```
function welcomeUser() {  
    var message = "Welcome to the site!";  
    alert(message);  
}  
var aliasFunction = welcomeUser;  
aliasFunction();
```

Events!



```
<body>  
  <button id="background-button">Background Change</button>  
  <script src="main.js" type="text/javascript"></script>  
</body>
```

Button Styling

- [Stylish CSS Buttons](#) from callmenick.com
- [css3buttongenerator.com](#)
- [w3schools](#) on button styling

```
// Get an element
```


```
var button = document.querySelector("#background-button");
```

button

```
// Get an element  
var button = document.querySelector("#background-button");  
  
// Define the event handler function  
function changeBackground() {  
    button.style.backgroundColor = "red";  
}
```

button

changeBackground

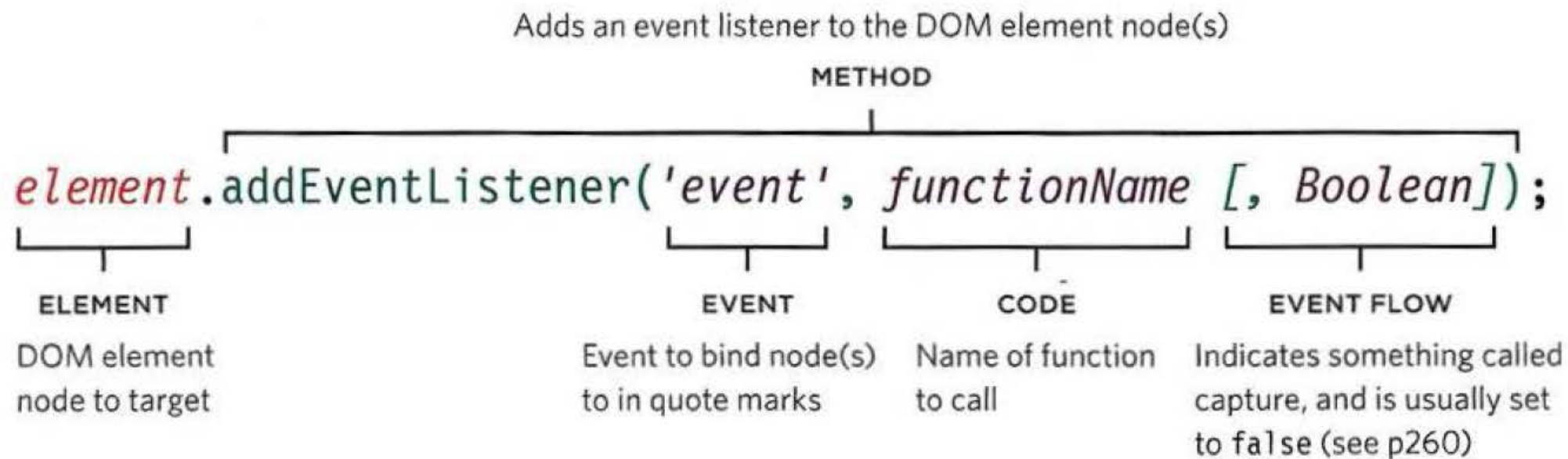


```
// Get an element
var button = document.querySelector("#background-button");

// Define the event handler function
function changeBackground() {
    button.style.backgroundColor = "red";
}

// Bind the event handler to the event
button.addEventListener("click", changeBackground);
```





Example events:

- click
- dblclick
- keydown

Browser Events

- [Megalist of events](#)
- Mouse events
 - click, mousedown, mouseup, mousemove, mouseenter, ...
- Keyboard events
 - keydown, keypress, keyup
- View events
 - resize, scroll
- Sensor events
 - orientationchange, devicemotion, etc.
- Etc.

Function Scope

LOCAL VARIABLE SCOPE



```
function greetPerson(greeting, person) {  
    var message = greeting + ", " + person + ".";  
    document.write("<h1>" + message + "</h1>");  
}
```

```
greetPerson("Good morning", "Mr. President");  
console.log(message); // ERROR  
console.log(greeting); // ERROR
```

GLOBAL SCOPE

```
var greetingHeader = document.getElementById("greeting");
```

```
function greetPerson(greeting, person) {  
    var message = greeting + ", " + person + ".";  
    greetingHeader.textContent = message;  
}
```

```
greetPerson("Good morning", "Mr. President");  
console.log(greetingHeader); // NO ERROR  
console.log(message); // ERROR  
console.log(greeting); // ERROR
```

Event Accumulation Pattern

```
// Global variables
var numClicks = 0;
var clickButton = document.querySelector("#click-button");
var clickCounterDiv = document.querySelector("#click-counter");

// Event handler
function countClicks() {
    numClicks += 1;
    clickCounterDiv.textContent = "Clicks: " + numClicks;
}

// Bind the event handler to the "click" event on the button
clickButton.addEventListener("click", countClicks);
```

Order of Execution

```
// Global variables
var numClicks = 0;
var clickButton = document.querySelector("#click-button");
var clickCounterDiv = document.querySelector("#click-counter");

// Event handler
function countClicks() {
    numClicks += 1;
    clickCounterDiv.textContent = "Clicks: " + numClicks;
}

// Bind the event handler to the "click" event on the button
clickButton.addEventListener("click", countClicks);
```

Order of Execution

(when page loads)

```
// Global variables  
① var numClicks = 0;  
② var clickButton = document.querySelector("#click-button");  
③ var clickCounterDiv = document.querySelector("#click-counter");  
  
// Event handler  
function countClicks() {  
    numClicks += 1;  
    clickCounterDiv.textContent = "Clicks: " + numClicks;  
}  
  
// Bind the event handler to the "click" event on the button  
④ clickButton.addEventListener("click", countClicks);
```

*countClicks is not called UNTIL someone clicks on the button!