CSIT884:
Web Development

JavaScript events & Advanced functions

UNIVERSITY
OF WOLLONGONG
AUSTRALIA

School of Computing and Information Technology
University of Wollongong

Event

In JavaScript, we use events (such as clicking a button, pressing a mouse, …) to control user interaction with the web page.

```
<button onClick="handlerFunction1()">
  Click me
</button>

<span onMouseOver="handlerFunction2()">
 some text
</span>
```

JavaScript event handler is often associated with a HTML element (such as a button, an image, a text span,…), by specifying an attribute such as `onClick`, `onMouseOver`, `onChange`, ... We often write a function so that when the event happens the function will get executed.

onLoad

The **`onLoad`** event is activated after the element’s resource is completely loaded, for example,

- On `<body>` element, the onLoad event is activated after all the contents of the web page have been loaded (including images, JavaScript files, CSS files, etc.)
- On `<img>` element, the onLoad event is activated after the image file has been retrieved from the server.

```
<script>

function greeting(){
  alert("Welcome to my website!");
}

</script>


<body onLoad="greeting()">
```

onClick

The **onClick** event is activated when the user clicks on the element such as a button, an image, a check box, a radio button, ...

```
<script>

function cat(){
  alert("Don't click me, click the dog!");
}

function dog(){
  alert("I don't mind being clicked!");
}

</script>

<img onClick="cat()" src="cat.png" /> . . . .(1)

<img onClick="dog()" src="dog.png" /> . . . .(2)
```

onFocus

The **onFocus** event is activated when the element is in focus, such as,

- when the user enters into a text/password field to type,
- when the user enters into a textarea to type,
- when the user enters into a drop-down list to make a selection.

```
<input type="text" onFocus="..." />

<textarea onFocus="...">
</textarea>

<select onFocus="...">
  <option value="...">..</option>
  <option value="...">...</option>
</select>
```

onBlur

The **`onBlur`** event is activated when the element loses focus, such as,

- when the user moves out of a text/password field,
- when the user moves out of a textarea,
- when the user moves out of a drop-down list.

```
<input type="text" onBlur="..." />

<textarea onBlur="...">
</textarea>

<select onBlur="...">
  <option value="...">..</option>
  <option value="...">...</option>
</select>
```

onChange

The **`onChange`** event is activated when the element loses the focus **and** its value has been changed, such as,

- when the user enters a text/password field, types some text, then moves out of the text field,
- when the user enters a textarea, types some text, then moves out of the textarea,
- when the user makes a new selection in a drop-down list.

Note that if the user goes into a text field but does not make changes to the text, then when the user moves out, this event will not be fired. Similarly, if the user selects the same option in a drop-down list, then this event will not be fired.

onChange

Example: When the user enters the discount code, leaves the text field, a function is triggered which transforms the discount code to uppercase:

```
Enter discount code:
<input type="text" id="discountCode" onChange="uppercase()">


function uppercase(){
  var discountField = document.getElementById("discountCode");
  discountField.value = discountField.value.toUpperCase();
}
```

onSelect

The **`onSelect`** event is activated when the user selects (highlights) some text in a text/password field or a textarea.

```
<input type="text" onSelect="..." />

<textarea onSelect="...">
</textarea>
```

Mouse Event

`onMouseDown:` the event is activated when the user presses a mouse button over the element.

`onMouseUp:` the event is activated when the user releases a mouse button over the element.

`onMouseOver:` the event is activated when the user moves the mouse over the element.

`onMouseOut:` the event is activated when the user moves the mouse out of the element.

Mouse Event

```
<span id="demo" onMouseDown="mouseDown()" onMouseUp="mouseUp()">
Click Me
</span>



function mouseDown() {
  var demoSpan = document.getElementById("demo");
  demoSpan.innerHTML = "Release Me";
}

function mouseUp() {
  var demoSpan = document.getElementById("demo");
  demoSpan.innerHTML = "Thank You";
}
```

Mouse Event

```
<span id="demo" onMouseOver="mouseOver()" onMouseOut="mouseOut()">
Mouse Over Me
</span>
```

```
function mouseOver() {
  var demoSpan = document.getElementById("demo");
  demoSpan.innerHTML = "Mouse Over Me";
}

function mouseOut() {
  var demoSpan = document.getElementById("demo");
  demoSpan.innerHTML = " Thank You";
}
```

onSubmit

The **onSubmit** event is activated when the user submits a form.

We often use this to validate the form

- we can use function that checks the user input,
- if there the user input is not valid, the function should return false, and that will stop the form from submission to the server
- If everything is valid, the function should return true, in this case, the form will be submitted to the server

```
<script>
function validateForm(){
  if (... something wrong ...) {
    return false;
  }
  return true;
}
</script>

<form onSubmit="return validateForm()" action="..."
method="..." >
 ...
</form>
```

onSubmit

```
<script>
function validateForm(){
  var fname = document.getElementById("fname").value;

  if (fname == null || fname.trim() == "") {
    alert("You must enter first name");
    return false;
  }

  //check last name and others input...

  return true;
}
</script>

<form onSubmit="return validateForm()" action="..." method="..." >
  First name: <input id="fname" type="text" name="fname">
  Last name: <input id="lname" type="text" name="lname">
  ...
</form>
```

String functions

```
var text = "One Fish, Two Fish, Red Fish, Blue Fish";

var textLength = text.length;
                 → 39

var upper = text.toUpperCase();
                 → ONE FISH, TWO FISH, RED FISH, BLUE FISH

var lower = text.toLowerCase();
                 → one fish, two fish, red fish, blue fish

var fishIndex = text.indexOf("Fish");        → 4
var catIndex = text.indexOf("cat");          → -1

var redFound = text.includes("Red");         → true
var greenFound = text.includes("Green");     → false
```

String functions

```
var text = "One Fish, Two Fish, Red Fish, Blue Fish";

var s1 = text.slice(10, 12);    → Tw

var s2 = text.slice(10);        → Two Fish, Red Fish, Blue Fish

var s3 = text.slice(-9, -6);    → Blu

var s4 = text.slice(-9);        → Blue Fish
```

Date functions

There are several ways to create a Date object.

```
var d = new Date(); //current date & time . . . (1)
var d = new Date(millisec); . . . (2)
var d = new Date(dateString); . . . (3)
var d = new Date(year, month, day, hour, min, sec, millisec);
                                                          . . . (4)
```

Date functions

```
var d = new Date(millisec);
```

Dates are calculated in milliseconds from 01st of January, 1970 00:00:00 Universal Time (UTC). One day contains 86,400,000 millisecond.

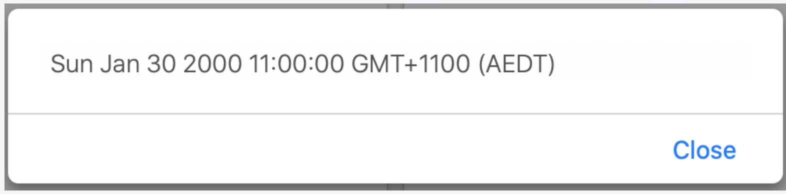
```
var d = new Date(86400000);
alert(d);     //02 Jan 1970 00:00:00 UTC
```

Date functions

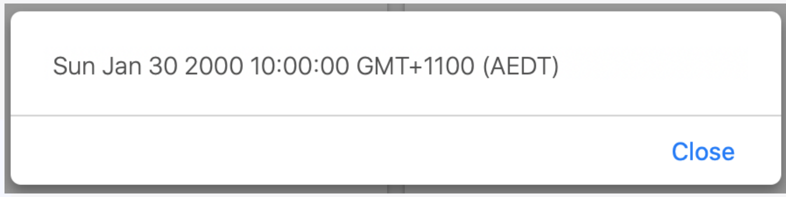
```
var d = new Date(dateString);

//using YYYY-MM-DD format
var d = new Date("2000-01-30");
alert(d);

//using YYYY-MM-DDTHH:MI:SS
var d = new Date("2000-01-30T10:00:00");
alert(d);
```



```
Sun Jan 30 2000 11:00:00 GMT+1100 (AEDT)
```



```
Sun Jan 30 2000 10:00:00 GMT+1100 (AEDT)
```

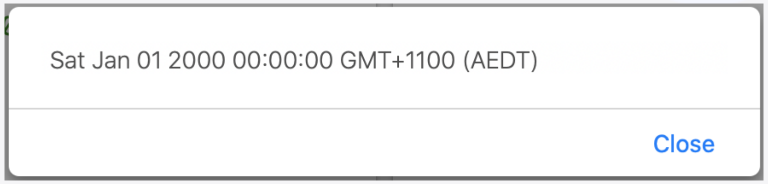
Date functions

```
var d = new Date(year, month, day, hour, min, sec, millisec);
```

The last 4 arguments can be omitted.

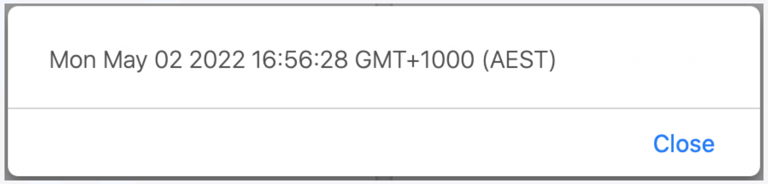
Months count from 0 to 11. January is 0. December is 11.

```
var d = new Date(2000, 0, 1);
alert(d);
```



```
Sat Jan 01 2000 00:00:00 GMT+1100 (AEDT)
```

```
var d = new Date(2020, 28, 2, 16, 56, 28);
alert(d);
```



```
Mon May 02 2022 16:56:28 GMT+1000 (AEST)
```

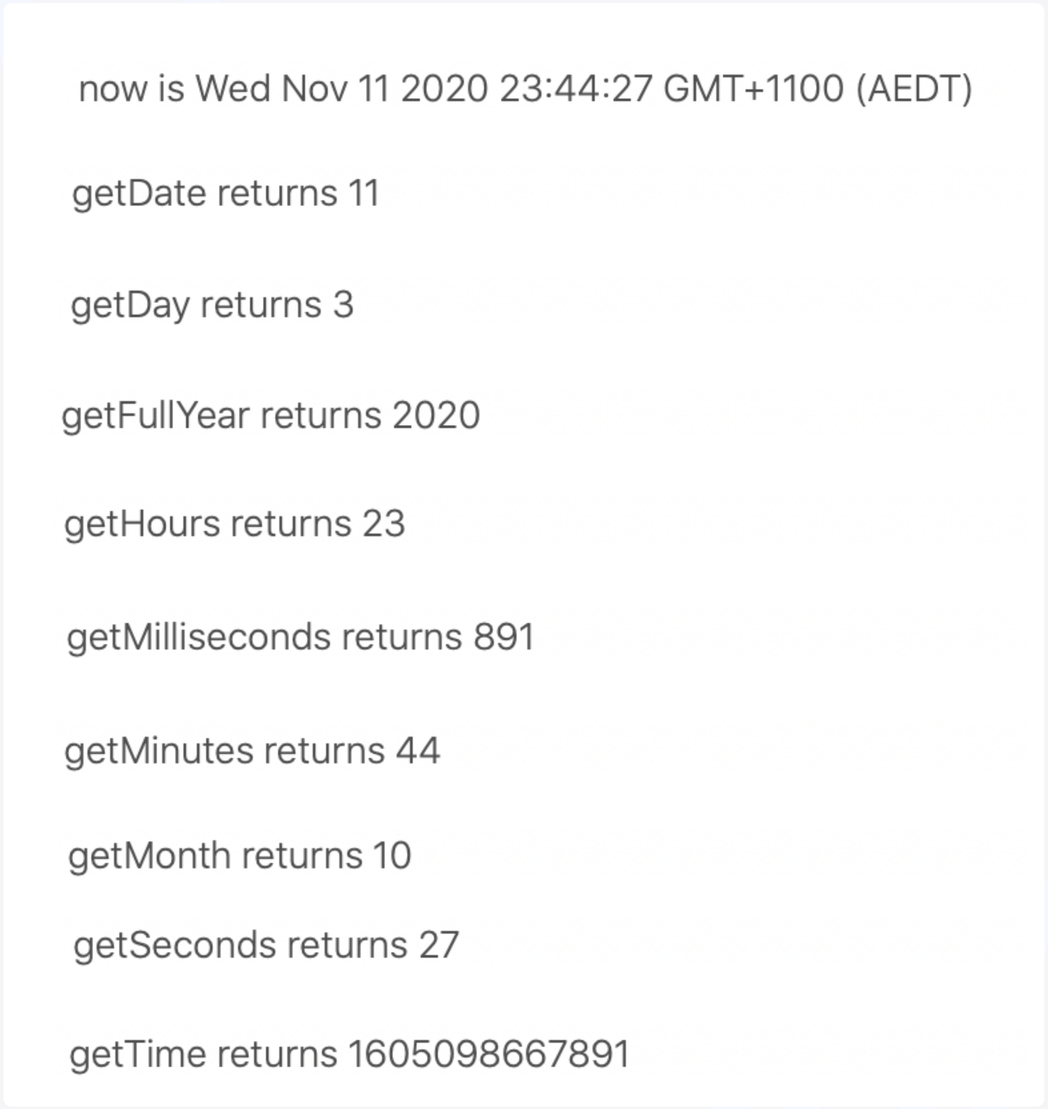
Date functions

`getDate()`	Get the day as a number (1-31)
`getDay()`	Get the weekday as a number (**0-6**) *Sunday is 0, Saturday is 6*
`getFullYear()`	Get the four digit year (yyyy)
`getHours()`	Get the hour (0-23)
`getMilliseconds()`	Get the milliseconds (0-999)
`getMinutes()`	Get the minutes (0-59)
`getMonth()`	Get the month (**0-11**) *January is 0, December is 11*
`getSeconds()`	Get the seconds (0-59)
`getTime()`	Get the milliseconds since 01/Jan/1970

Date functions

```
var now = new Date();
alert("now is " + now);
alert("getDate returns " + now.getDate());
alert("getDay returns " + now.getDay());
alert("getFullYear returns " + now.getFullYear());
alert("getHours returns " + now.getHours());
alert("getMilliseconds returns " + now.getMilliseconds());
alert("getMinutes returns " + now.getMinutes());
alert("getMonth returns " + now.getMonth());
alert("getSeconds returns " + now.getSeconds());
alert("getTime returns " + now.getTime());
```

Date functions



```
now is Wed Nov 11 2020 23:44:27 GMT+1100 (AEDT)

getDate returns 11

getDay returns 3

getFullYear returns 2020

getHours returns 23

getMilliseconds returns 891

getMinutes returns 44

getMonth returns 10

getSeconds returns 27

getTime returns 1605098667891
```

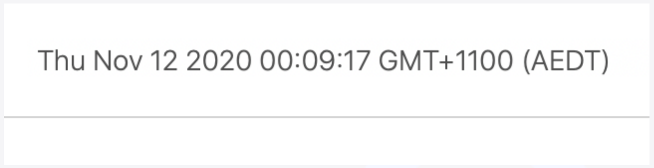
Date functions

`setDate()`	Set the day as a number (1-31)
`setTime()`	Set the milliseconds since 01/Jan/1970
`setFullYear()`	Set the year (optionally month and day)
`setMonth()`	Set the month (0-11)
`setHours()`	Set the hour (0-23)
`setMinutes()`	Set the minutes (0-59)
`setSeconds()`	Set the seconds (0-59)
`setMilliseconds()`	Set the milliseconds (0-999)

Date functions

(1)

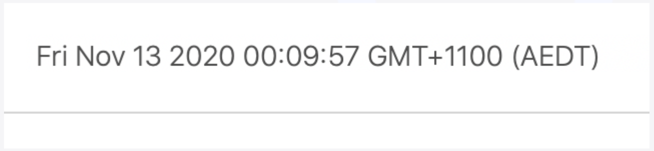
```
var now = new Date();
alert(now);
```



```
Thu Nov 12 2020 00:09:17 GMT+1100 (AEDT)
```

(2)

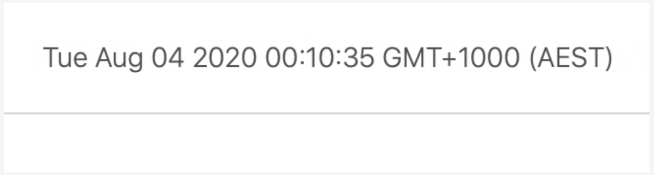
```
var tomorrow = new Date();
tomorrow.setDate(now.getDate() + 1);
alert(tomorrow);
```



```
Fri Nov 13 2020 00:09:57 GMT+1100 (AEDT)
```

(3)

```
var hundredDayAgo = new Date();
hundredDayAgo.setDate(now.getDate() - 100);
alert(hundredDayAgo);
```



```
Tue Aug 04 2020 00:10:35 GMT+1000 (AEST)
```

Confirm box

A confirm box is often used if you want the user to verify or accept something.

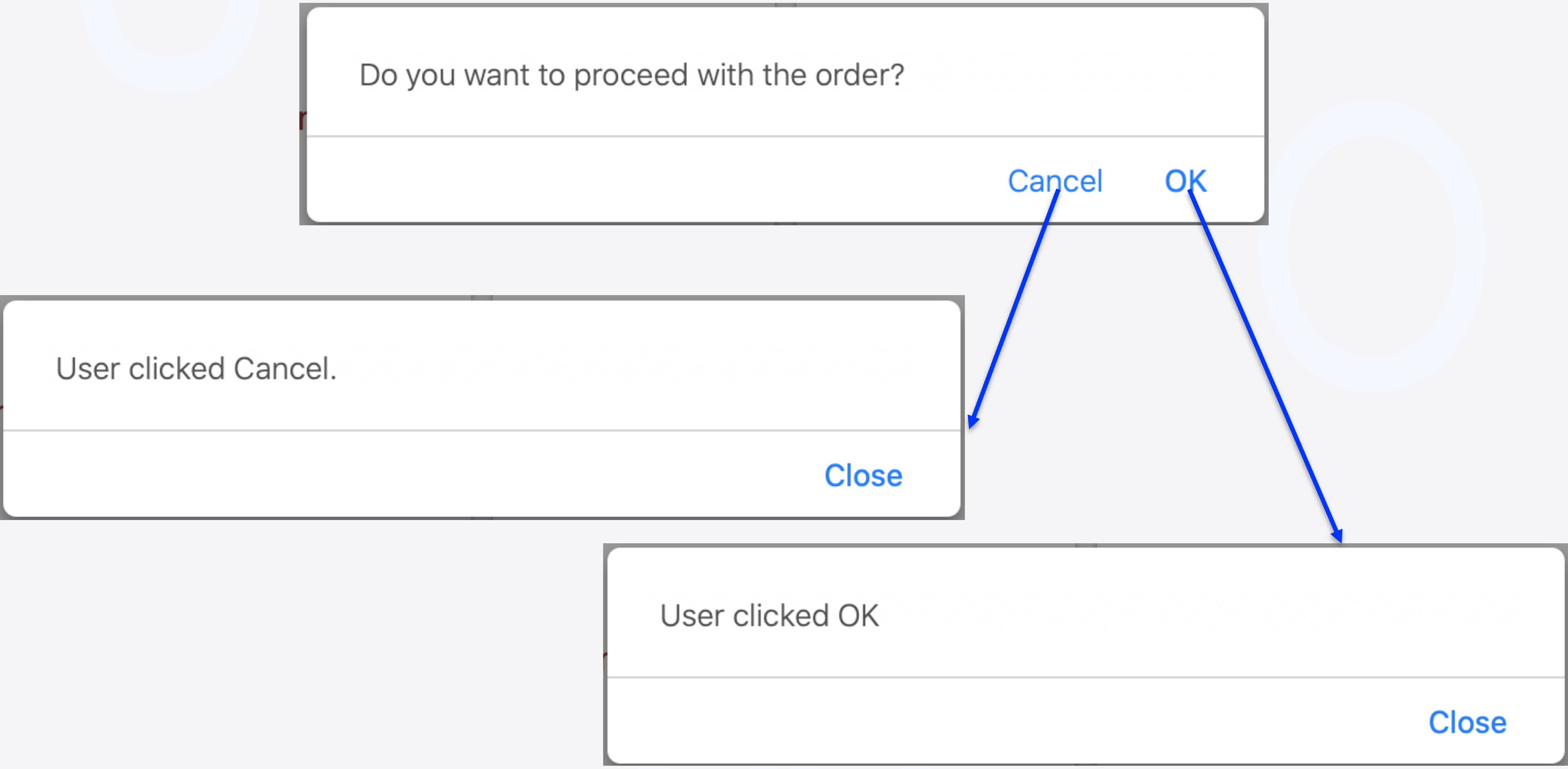
When a confirm box pops up, the user will have to click either "OK" or "Cancel".

If the user clicks "OK", the box returns true.

If the user clicks "Cancel", the box returns false.

```
var ok = confirm("Do you want to proceed with the order?");
if(ok){
  alert("User clicked OK");
}else{
  alert("User clicked Cancel.");
}
```

Confirm box


Do you want to proceed with the order?

User clicked Cancel.

User clicked OK

Prompt box

When a prompt box pops up, the user will have to click either "OK" or "Cancel".

If the user clicks "OK" the box returns the input value.

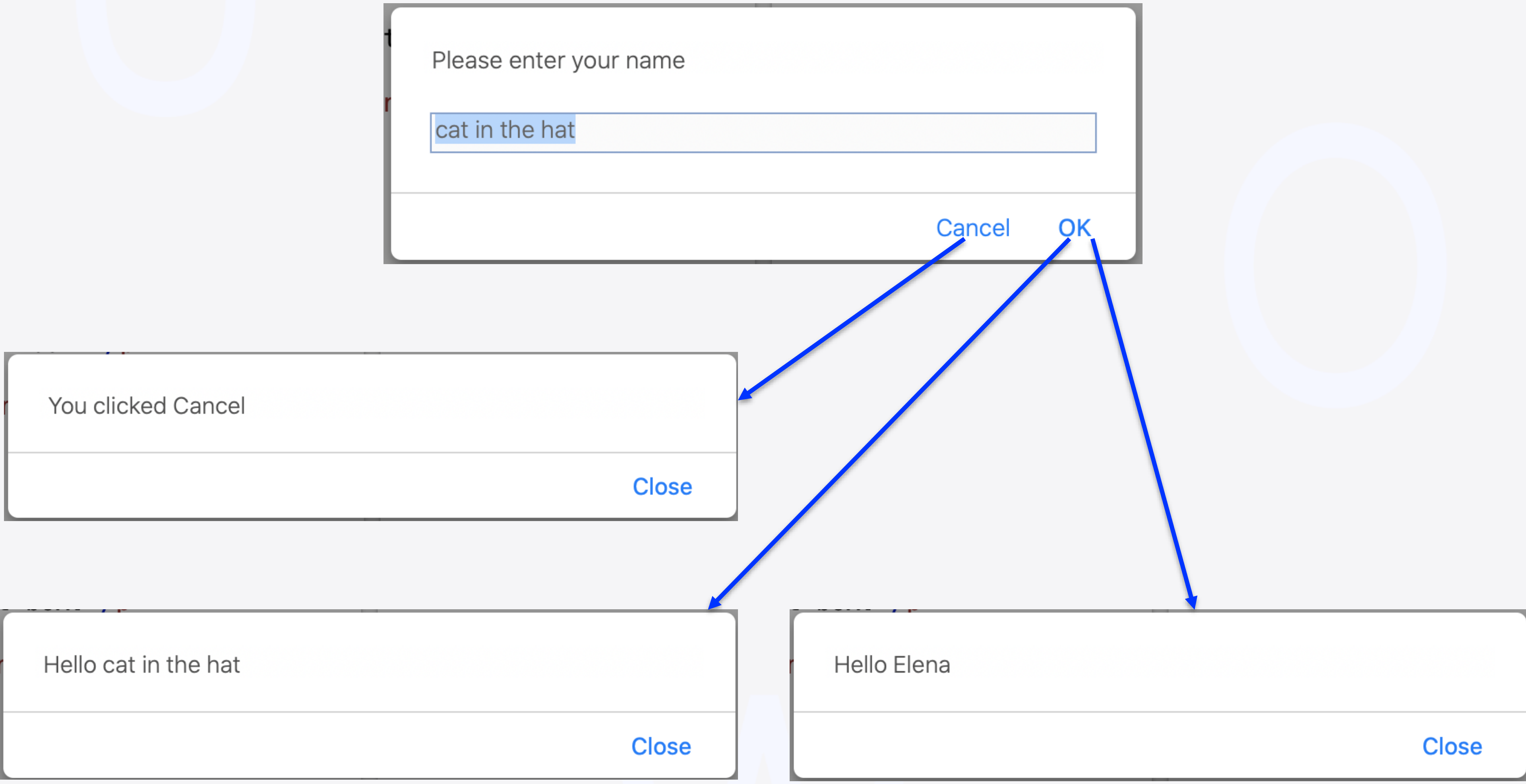
If the user clicks "Cancel" the box returns null.

We can also specify the default text in the input box:

```
prompt("sometext","defaultText");
```

```
var name = prompt("Please enter your name", "cat in the hat");
if(name != null){
  alert("Hello " + name);
}else{
  alert(”You clicked Cancel”);
}
```

Prompt box



Please enter your name

cat in the hat

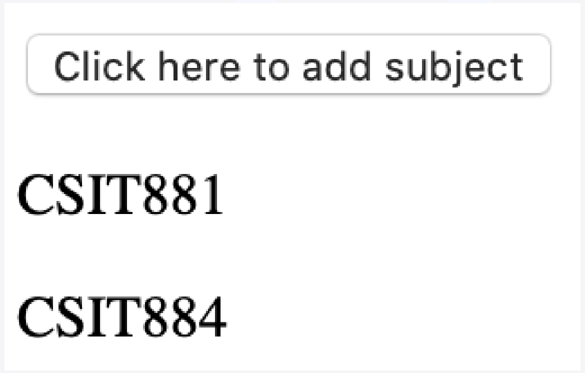
You clicked Cancel

Hello cat in the hat

Hello Elena

Add subject 1

When the button is clicked, a prompt box appears asking the user to enter a subject code. Then the subject is added to the page.



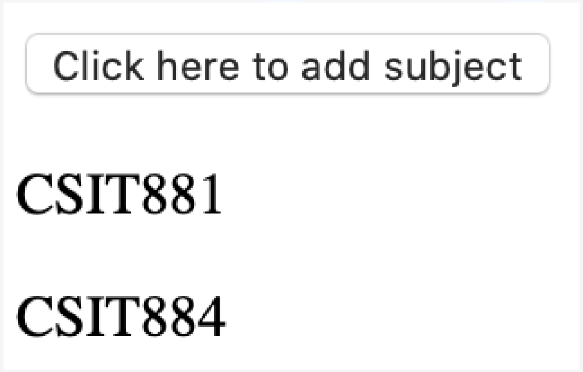
CSIT881

CSIT884

Add subject 1

```
<button onClick="addSubject()">
Click here to add subject
</button>

<div id="subjectList">
</div>
```



CSIT881

CSIT884

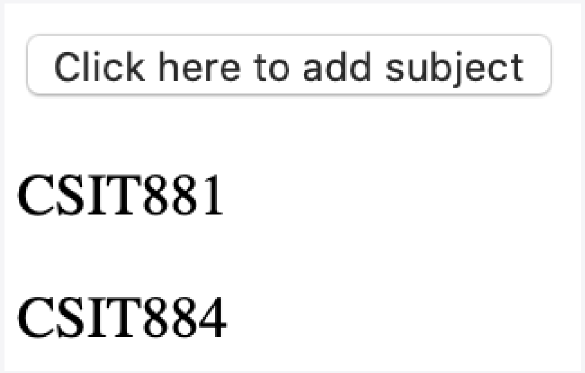
Add subject 1

```
function addSubject(){
  // ask user for a subject code
  var subject = prompt("Enter subject code");

  if(subject != null){
    // create a new paragraph holding the subject code
    var para = document.createElement("p"); //Create a <p> element
    var subjectText = document.createTextNode(subject);
                                                  //Create a text node
    para.appendChild(subjectText); //Append the text to <p>

    // add the new paragraph element to the div element
    var subjectDiv = document.getElementById("subjectList");
    subjectDiv.appendChild(para);
  }
}

<div id="subjectList">
  <p>CSIT881</p>
</div>
```

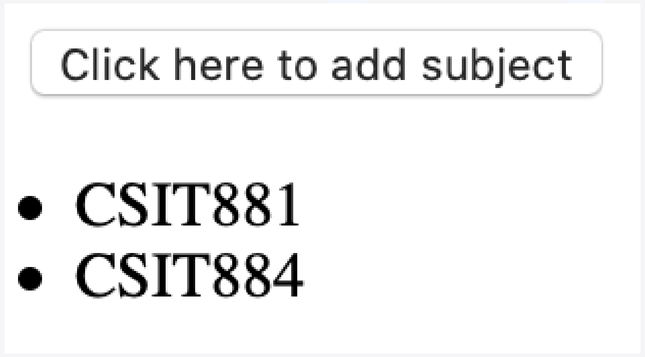


CSIT881

CSIT884

Add subject 2

When the button is clicked, a prompt box appears asking the user to enter a subject code. Then the subject is added to the page in an **unordered list**.

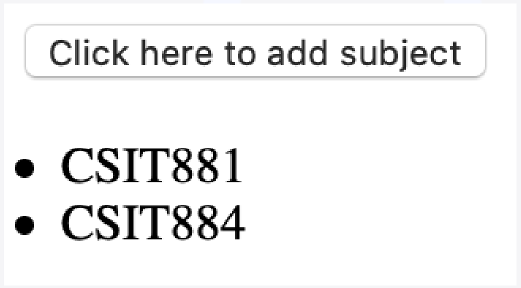


- CSIT881
- CSIT884

Add subject 2

```
<button onClick="addSubject()">
Click here to add subject
</button>

<ul id="subjectList">
</ul>
```



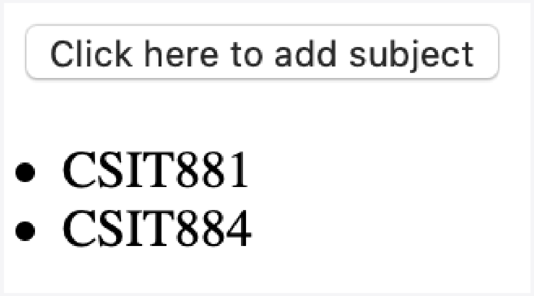
- CSIT881
- CSIT884

Add subject 2

```
function addSubject(){
  // ask user for a subject code
  var subject = prompt("Enter subject code");

  if(subject != null){
    // create a new list item holding the subject code
    var li = document.createElement("li");
    var subjectText = document.createTextNode(subject);
    li.appendChild(subjectText);

    // add the new list item element to the unordered list
    var subjectUL = document.getElementById("subjectList");
    subjectUL.appendChild(li);
  }
}
```



- CSIT881
- CSIT884