JAVASCRIPT & JQUERY

interactive front-end web development

CHAPTER 6

EVENTS

WHAT IS AN EVENT?

Events are the browser's way of saying, "Hey, this just happened."

When an event **fires**, your script can then react by running code (e.g. a function).

By running code when an event fires, your website responds to the user's actions.

It becomes **interactive**.

DIFFERENT EVENT TYPES

USER INTERFACE EVENTS

```
load
unload
error
resize
scroll
```

KEYBOARD EVENTS

```
keydown
keyup
keypress
```

MOUSE EVENTS

```
click
dblclick
mousedown
mouseup
mouseover
mouseout
```

FOCUS EVENTS

```
focus / focusin
blur  / focusout
```

FORM EVENTS

```
input
change
submit
reset
cut
copy
paste
select
```

HOW EVENTS TRIGGER JAVASCRIPT CODE

1

1

Select the **element** node(s) the script should respond to

1

Select the **element** node(s) the script should respond to

2

1

Select the **element** node(s) the script should respond to

2

Indicate the **event** on the selected node(s) that will trigger a response

1

Select the **element** node(s) the script should respond to

2

Indicate the **event** on the selected node(s) that will trigger a response

3

1

Select the **element** node(s) the script should respond to

2

Indicate the **event** on the selected node(s) that will trigger a response

3

State the code you want to run when the event occurs

BINDING AN EVENT TO AN ELEMENT

There are three ways to bind an event to an element:

HTML event handler attributes
Traditional DOM event handlers
DOM Level 2 event listeners

The following examples show a **blur** event on an element stored in a variable called `el` that triggers a function called `checkUsername()`.

HTML EVENT HANDLER ATTRIBUTES (DO NOT USE)

```
<input type="text" id="username"
      onblur="checkUsername()">
```

HTML EVENT HANDLER ATTRIBUTES (DO NOT USE)

ELEMENT

```
<input type="text" id="username"
      onblur="checkUsername()">
```

HTML EVENT HANDLER ATTRIBUTES (DO NOT USE)

```
<input type="text" id="username"
       onblur="checkUsername()">
```

EVENT

HTML EVENT HANDLER ATTRIBUTES (DO NOT USE)

```
<input type="text" id="username"
       onblur="checkUsername()">
```

FUNCTION

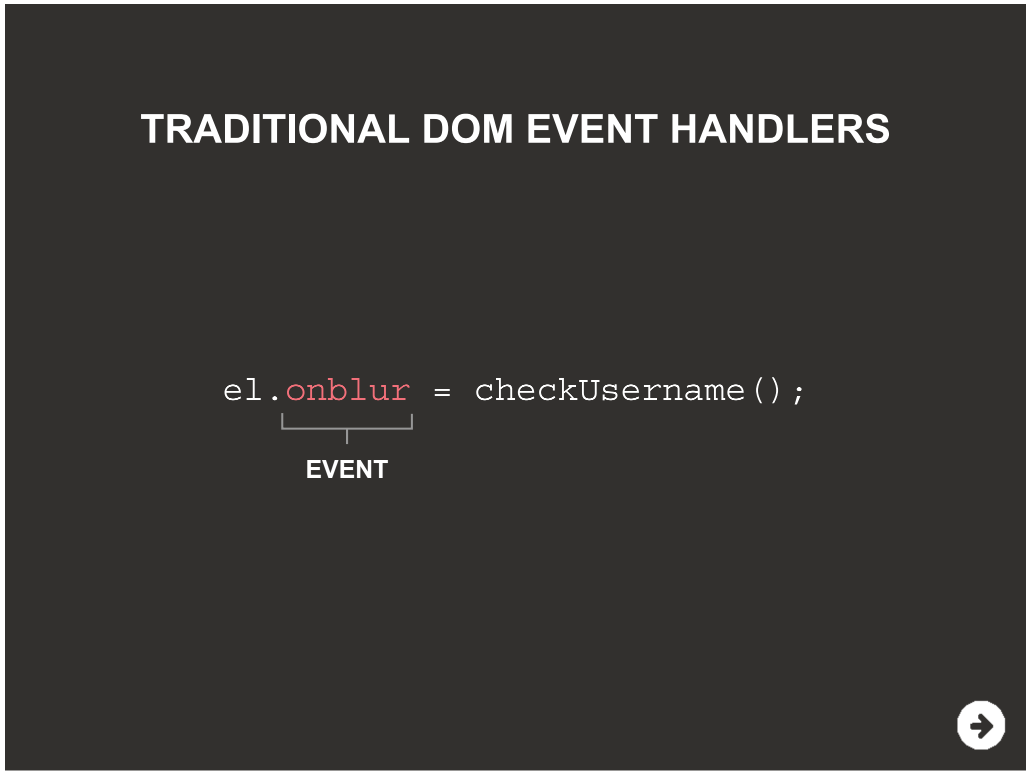
TRADITIONAL DOM EVENT HANDLERS

```
el.onblur = checkUsername();
```

TRADITIONAL DOM EVENT HANDLERS

```
el.onblur = checkUsername();
```

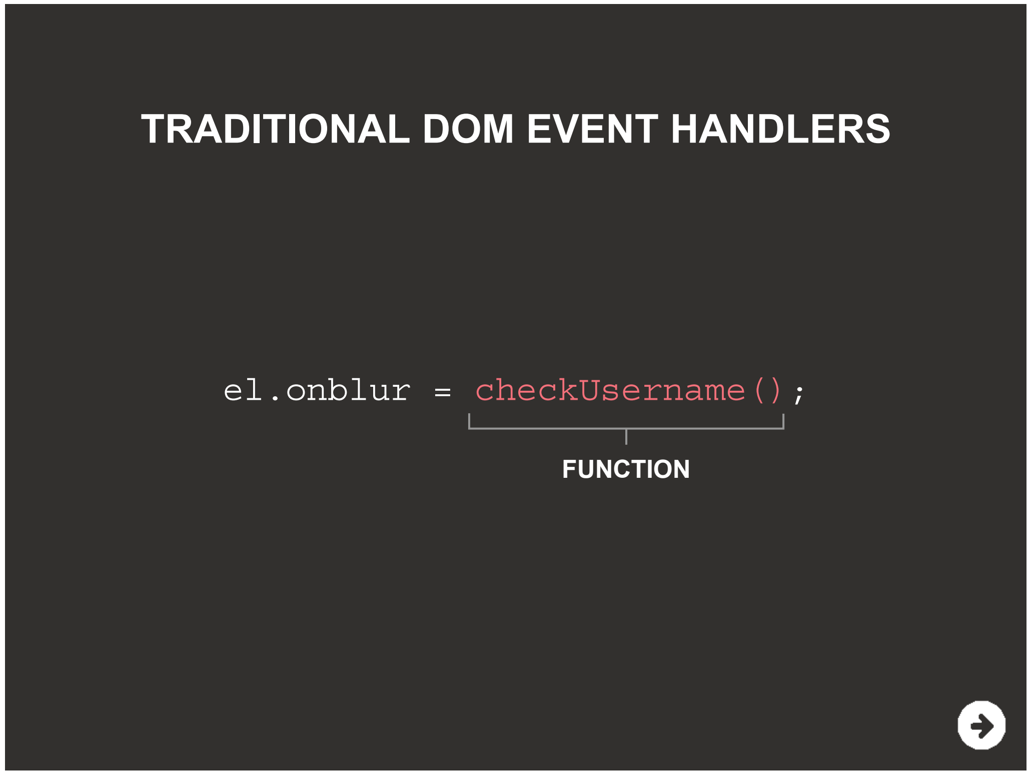
ELEMENT



TRADITIONAL DOM EVENT HANDLERS

```
el.onblur = checkUsername();
```

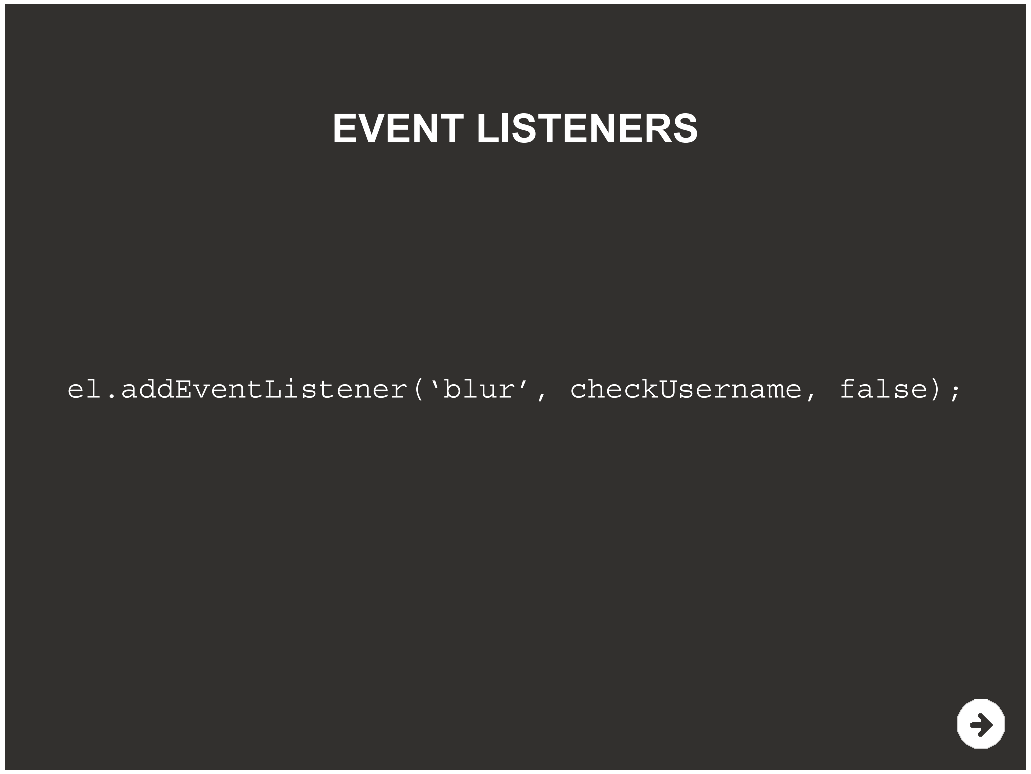
EVENT



TRADITIONAL DOM EVENT HANDLERS

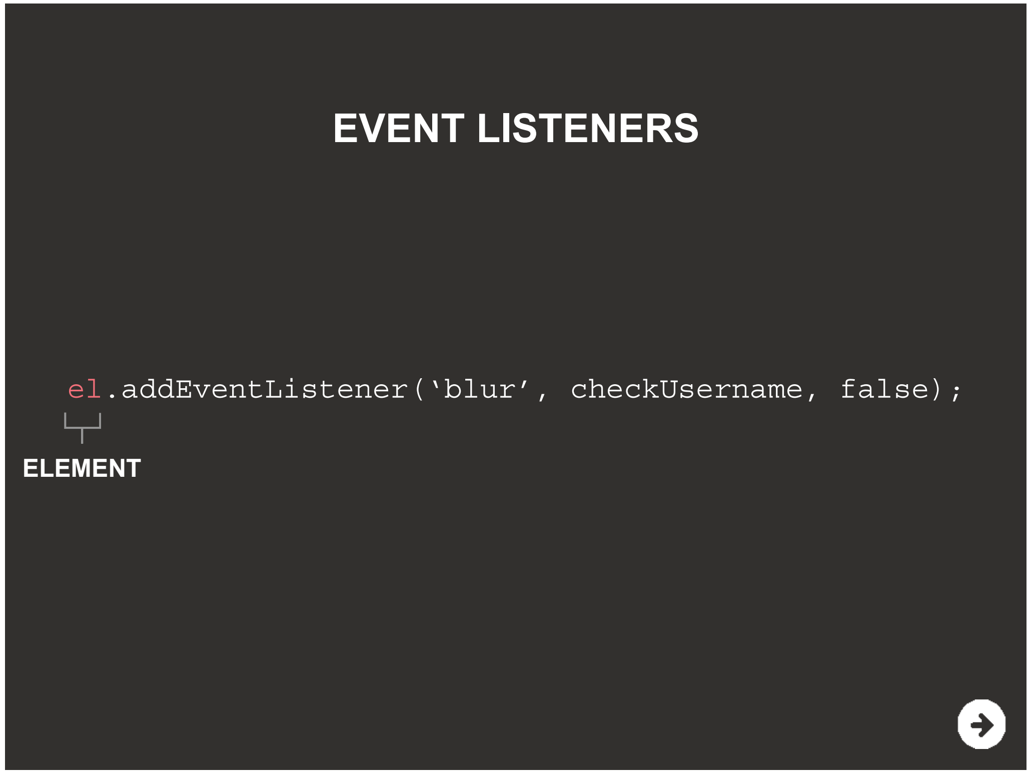
```
el.onblur = checkUsername();
```

FUNCTION



EVENT LISTENERS

```
el.addEventListener('blur', checkUsername, false);
```



EVENT LISTENERS

```
el.addEventListener('blur', checkUsername, false);
```

ELEMENT

EVENT LISTENERS

```
el.addEventListener('blur', checkUsername, false);
```

EVENT

EVENT LISTENERS

```
el.addEventListener('blur', checkUsername, false);
```

FUNCTION

EVENT LISTENERS

```
el.addEventListener('blur', checkUsername, false);
```

BOOLEAN
(OPTIONAL)

Because you cannot have parentheses after the function names in event handlers or listeners, passing arguments requires a workaround.

PARAMETERS WITH EVENT LISTENERS

```
el.addEventListener(‘blur’, function() {
    checkUsername(5);
}, false);
```

PARAMETERS WITH EVENT LISTENERS

```
el.addEventListener(‘blur’, function() {
    checkUsername(5);
}, false);
```

An anonymous function is used as the second argument.

PARAMETERS WITH EVENT LISTENERS

```
el.addEventListener(‘blur’, function() {
    checkUsername(5);
}, false);
```

Inside the anonymous function, a named function is called.

IE5 - 8 had a different event model and did not support `addEventListener()` but you can provide fallback code to make event listeners work with older versions of IE.

SUPPORTING OLDER VERSIONS OF IE

```
if (el.addEventListener) {
  el.addEventListener(‘blur’, function() {
    checkUsername(5);
  }, false);
} else {
  el.attachEvent(‘onblur’, function() {
    checkUsername(5);
  });
}
```

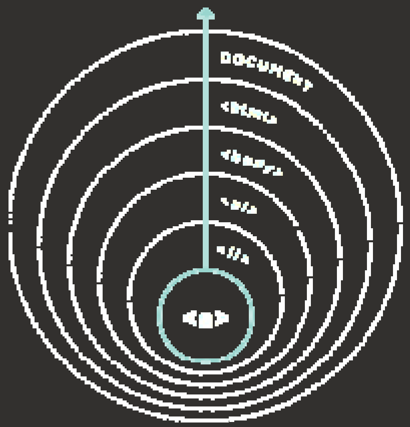
SUPPORTING OLDER VERSIONS OF IE

```
if (el.addEventListener) {
  el.addEventListener(‘blur’, function() {
    checkUsername(5);
  }, false);
} else {
  el.attachEvent(‘onblur’, function() {
    checkUsername(5);
  });
}
```

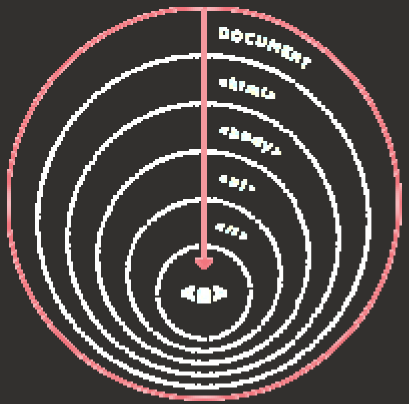
EVENT FLOW

HTML elements nest inside other elements. If you hover or click on a link, you will also be hovering or clicking on its parent elements.

EVENT BUBBLING



EVENT CAPTURING



THE EVENT OBJECT

When an event occurs,the `event` object can tell you information about it and which element it happened upon.

PROPERTIES

```
target
type
cancelable
```

METHODS

```
preventDefault()
stopPropagation()
```

ELEMENT AN EVENT OCCURRED ON

1: EVENT LISTENER CALLS FUNCTION

```
function checkUsername(e) {
  var target = e.target;
}

var el = document.getElementById('username');
el.addEventListener('blur', checkUsername, false);
```

ELEMENT AN EVENT OCCURRED ON

2: EVENT OBJECT PASSED TO FUNCTION

```
function checkUsername(e) {
  var target = e.target;
}

var el = document.getElementById('username');
el.addEventListener('blur', checkUsername, false);
```

ELEMENT AN EVENT OCCURRED ON

3: ELEMENT THAT EVENT HAPPENED ON

```
function checkUsername(e) {
  var target = e.target;
}

var el = document.getElementById('username');
el.addEventListener('blur', checkUsername, false);
```

EVENT DELEGATION

Creating event listeners for a lot of elements can slow down a page, but event flow allows you to listen for an event on a parent element.

Placing an event listener on a container element:

Works with new elements
Solves limitations with the `this` keyword
Simplifies code