Keyboard & Form Events, Strings

ITP 301
Spring 2021

Keyboard Events

Triggered by keyboard on HTML elements.

Element needs to be focusable* (*see next slide*).

`onkeydown`	Key is pressed.
`onkeyup`	Key is released.
`onkeypress`	Triggered immediately after `onkeydown`. Only works with keys used for “typing” text. Not triggered by Ctrl, Alt, arrows, etc.

```
document.onkeydown = function(){
  // Key is pressed.
}

document.onkeyup = function(){
  // Key is released.
}

document.onkeypress = function(){
  // "Typing" key is pressed.
}
```


tabindex="1"

Focusable Elements

By default, most HTML elements are focusable, including:

- HTML form elements,
- Buttons,
- anchor tags,
- etc.

`<div>` tags are **not** focusable by default.

To make any HTML element focusable, add `tabindex` attribute set to `1`.

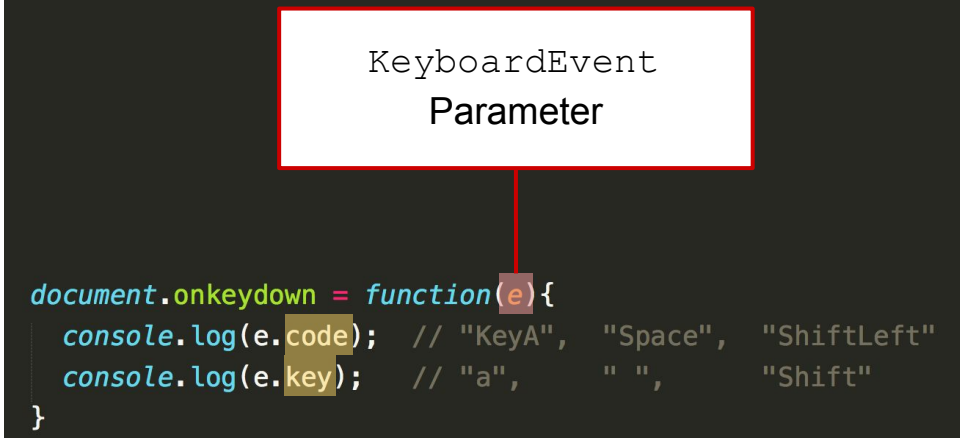
```
<div id="focusable-div" tabindex="1">
  Focusable Div
</div>
```

Detecting Keys

Each keyboard event passes a `KeyboardEvent` parameter to handler function.

Properties used to determine the key:

`code`	Physical key on the keyboard.
`key`	Value of the key.



Form Events

Triggered by interaction with forms or form elements.

`oninput`	Form element value changed.
`onchange`	Form element state changed. Triggered by text fields when they lose focus.
`onsubmit`	Form submission attempted. Returning `FALSE` will stop form submission.

```
document.querySelector('#text-field').oninput = function(){
  console.log(this.value);
  // Triggered with each key input.
  // Output text in the field to console.
};

document.querySelector('#checkbox').onchange = function(){
  console.log(this.checked);
  // Triggered when the checkbox is clicked.
  // Output whether the checkbox is checked.
};

document.querySelector('form').onsubmit = function(){
  // Triggered when form submission is attempted
  // ("Submit" button or "Enter" key pressed)

  if ( isValid ) {
    return true;
    // Proceed with form submission.
  } else {
    return false;
    // STOP form submission.
  }
}
```

Strings

Strings are series of characters.

- Sometimes treated like arrays with letters in each slot.

0	1	2	3	4	5	6	7	8	9	10	11
H	e	l	l	o		W	o	r	l	d	!

`.length`	Length of the string.
`.toLowerCase()`	Convert to lower case.
`.toUpperCase()`	Convert to upper case.
`.indexOf()`	Return index of first substring occurrence. If substring is not found, return `-1`.

```
var myString = "Hello World!";

console.log( myString[0] ); // Outputs 'H'.
console.log( myString[1] ); // Outputs 'e'.

console.log( myString.length ); // Outputs '12'.

console.log( myString.toLowerCase() ); // Outputs 'hello world!'.
console.log( myString.toUpperCase() ); // Outputs 'HELLO WORLD!'.

console.log( myString.indexOf('o') ); // outputs '4'.
console.log( myString.indexOf('llo') ); // outputs '2'.
console.log( myString.indexOf('Hi') ); // outputs '-1'.
```