Events & Animation

Interactivity

Shadi Lahham - Programmazione web - Frontend - Javascript

Events

Why we need events

- Form validation and processing
- Interactive slideshows
- Games
- Single-page webapps
- Anything that involves user interaction

Adding Event Listeners

Creating an event listener:

```
domNode.addEventListener(eventType, eventListener);
```

HTML:

```
<button id="counter">0</button>
```

Javascript:

```
let counterButton = document.getElementById('counter');

let onButtonClick = function() {
    counterButton.innerHTML = parseInt(counterButton.innerHTML) + 1;
};

counterButton.addEventListener('click', onButtonClick, false);
```

Event Types

Browsers trigger many events. A short list:

- mouse events (MouseEvent)
  - mousedown, mouseup, click, dblclick, mousemove, mouseover, mousewheel, mouseout, contextmenu
- touch events (TouchEvent)
  - touchstart, touchmove, touchend, touchcancel
- keyboard events (KeyboardEvent)
  - keydown, keypress, keyup
- form events
  - focus, blur, change, submit
- window events
  - scroll, resize, hashchange, load, unload

Processing Form Input

Events can be used to process form input:
- by responding to button click events
- or by responding to input click/change events

HTML:

```
<input id="myname" type="text">
<button id="button">Say My Name</button>
```

Javascript:

```
let button = document.getElementById('button');

let onClick = function(event) {
  let myName = document.getElementById("myname").value;
  console.log("Hi, " + myName);
};

button.addEventListener('click', onClick);
```

Event

Represents an event in the DOM

Important methods and properties:

```
event.stopPropagation();
event.target;
event.currentTarget;
event.preventDefault();
```

References:

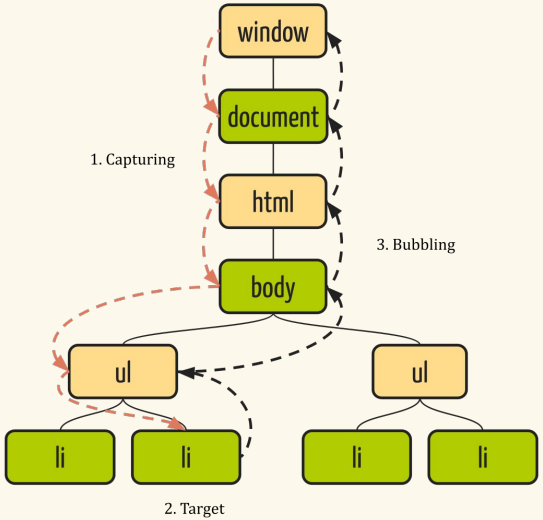
Event.stopPropagation()
Event.target
Event.currentTarget
Event.preventDefault()

Bubbling and capturing

Bubbling and capturing

Phases of event propagation

1. Capturing
2. Target
3. Bubbling



Bubbling and capturing

Events in Javascript propagate through the DOM tree

The concepts in the following links are a fundamental part of the material of this unit

Study them in detail

Bubbling and capturing

Event Bubbling in JavaScript? Event Propagation Explained

Bubbling and capturing - example

```
<div id="container">
  <ul>
    <li>item1</li>
    <li id="special">item2</li>
    <li>item3</li>
  </ul>
</div>

let container = document.getElementById('container');
let special = document.getElementById('special');

special.addEventListener('click', function(event) {
  console.log('Special', getLabel(event.eventPhase));
});
```

Bubbling and capturing - example

```
// capturing
container.addEventListener('click', function(event) {
  console.log('Container', getLabel(event.eventPhase));
},true);

// bubbling
container.addEventListener('click', function(event) {
  console.log('Container', getLabel(event.eventPhase));
});

// get phase label
function getLabel(phase) {
  return phase === 3 ? 'bubbling' : phase === 2 ? 'on target' : 'capturing';
}
```

Removing an Event Listener

```
node.addEventListener('click', handler);
node.removeEventListener('click', handler);      // Succeeds

node.addEventListener('click', handler, true);
node.removeEventListener('click', handler, false);     // Fails
node.removeEventListener('click', handler, true);      // Succeeds
```

EventTarget.removeEventListener()

Animation

The window object

When you run Javascript in the browser, you can access the window object which has many useful properties and methods:

Examples:

```
window.location.href;
window.navigator.userAgent;
window.scrollTo(10, 50);
window.console.log("Hello world!");
```

Window has may other useful properties and methods. Explore them!

The window object is the assumed global object on a page.

```
window.console.log("Hi!");
is the same as:
console.log("Hi");
```

Animation in Javascript

The standard way to animate in JS is to use these window methods which we have seen earlier:

Calls a function once after a delay:

```
window.setTimeout(callbackFunction, delayMilliseconds);
```

Calls a function repeatedly, with specified interval between each call:

```
window.setInterval(callbackFunction, delayMilliseconds);
```

Example:

```
let makeImageBigger = function() {
  let img = document.getElementsByTagName('img')[0];
  img.setAttribute('width', img.width+10);
};
window.setInterval(makeImageBigger, 1000);
```

note: it is highly recommended to use CSS transitions and CSS animations rather than Javascript.

Animating CSS Styles

You can animate CSS styles to change size, transparency, position, color, etc

```
let fadeAway = function() {
  img.style.opacity = img.style.opacity - 0.1;
};

let img = document.getElementsByTagName('img')[0];
img.style.opacity = 1.0;
window.setInterval(fadeAway, 1000);
```

Animating CSS Styles

What happens to kitty?

```
let watchKittyFall = function() {
  let oldTop = parseInt(img.style.top);
  let newTop = oldTop + 10;
  img.style.top = newTop + 'px';
};

let img = document.getElementsByTagName('img')[0];
img.style.position = 'absolute';
img.style.top = '0px';
window.setInterval(watchKittyFall, 1000);
```

Stopping Animations

To stop an animation store the timer into a variable and clear it with:

```
window.clearTimeout(timer);
window.clearInterval(timer);
```

Example:

```
let fadeAway = function() {
  img.style.opacity = img.style.opacity - 0.1;
  if (img.style.opacity < .5) {
    window.clearInterval(fadeTimer);
  }
};

let img = document.getElementsByTagName('img')[0];
img.style.opacity = 1.0;
let fadeTimer = window.setInterval(fadeAway, 100);
```

Your turn

1.Story

Start with the following HTML:

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8" />
  <title>Story</title>
</head>
<body>
  <h1>Story</h1>
  <ul>
    <li>Noun: <input type="text" id="noun">
    <li>Adjective: <input type="text" id="adjective">
    <li>Someone's Name: <input type="text" id="person">
  </ul>
  <button id="gen-button">Lib it!</button>
  <div id="story"></div>
</body>
</html>
```

Continues on next page >>>

1.Story

- Add an event listener to the button so that it calls a makeStory function when clicked.
- In the makeStory function, retrieve the current values of the form input elements, make a story from them, and output that in the story div (like "Joseph really likes pink cucumbers.")

2.Calculate

Start with the following HTML:

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8" />
  <title>Calculator</title>
</head>
<body>
  <label>Square this number:
    <input type="number" id="square-input" size="2">
  </label>
  <button id="square-button">Calculate</button>
  <!--other inputs here -->

  <div id="solution"></div>
</body>
</html>
```

Continues on next page >>>

2.Calculate

- Add inputs for half number, percentage and circle area
- Use the functions from the previous calculator exercises
- For each operation, create an event listener for the button, and when it's clicked, find the value of the appropriate input and show the result of the calculation in the solution div
- Afterwards, change the code so that you respond to key presses so that the user doesn't have to click the button

3.Catwalk

Start with the following HTML:

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8" />
  <title>Cat Walk</title>
</head>
<body>

  <img style="position:absolute;" src="http://www.anniemation.com/clip_art/images/cat-walk.gif">

</body>
</html>
```

Continues on next page >>>

3. Catwalk

- The cat should start from the left side of the screen
- Write a function ‘catWalk()’ that moves the cat 10 pixels to the right
- Make the cat move across the screen by calling that function every 50ms
- Write different versions of the function to handle the following variants:
    - Variant 1: When the cat reaches the right side of the screen it should restart from the left
    - Variant 2: When the cat reaches the right side of the screen, it should move backwards. When it reaches the left it should move forwards
    - Variant 3: When the cat reaches the middle of the screen, replace the img with a different cat image. Keep it in the middle for 10 seconds, and then replace the img with the original image and have it continue the walk as in variant 2

Bonus

4. Advanced Catwalk

- Start with the code from the previous exercise
- Add 4 buttons at the top of the page: ‘start’, ‘faster’, ‘slower’ and ‘stop’
- Add an area to display info
- When the start button is clicked the cat should start moving across the screen
- The cat should stop moving when the stop button is clicked
- The cat moves faster when the faster button is clicked and slower when the slower button is clicked
- Show the current speed on screen in the info area
- Disable the start/stop/faster/slower buttons at the appropriate times
    - e.g. the user shouldn't be able to click "stop" if the cat isn't currently moving

5. Advanced Arrivals

- Start with the ‘Arrivals’ exercise from the previous lesson
- Add the following features:
  - When the user clicks a row, it should expand to show more information about the flight
  - When the user clicks an open row it should close again
  - If the user clicks a row, any other open rows should close
    - Like in this example of an accordion
  - Add a ‘Departures’ section with departing flights
  - The user should be able to switch between Arrivals and Departures with a fade-in/fade-out animation

References

Bubbling and capturing

Event Bubbling in JavaScript? Event Propagation Explained

Event.stopPropagation()

Event.target

Event.currentTarget

EventTarget.removeEventListener()