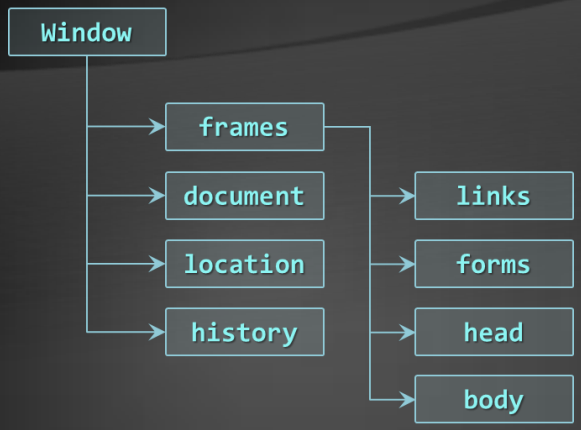
Telerik Academy



DOM Operations

Creating Dynamic Pages

Telerik Software Academy

Learning & Development Team

http://academy.telerik.com

Telerik Academy

Table of Contents

- Traversing the DOM
  - Parents, Children and Siblings
- DOM manipulation
  - Adding, Removing and Altering Elements
- Static and Live NodeLists

Telerik Academy

DOM Elements

Telerik Academy

DOM Elements

- DOM element is a JavaScript object that represents an element from the HTML
  - Selected using any of the DOM selectors
  - Created dynamically from code
- DOM elements can be changed
  - This changes are immediately applied to the DOM, and the HTML page

```
//changes the content of the div
selectedDiv.innerHTML = "changed";
//changes the background of the div to "#456"
selectedDiv.style.background = "#456";
var div = document.createElement("div");
```

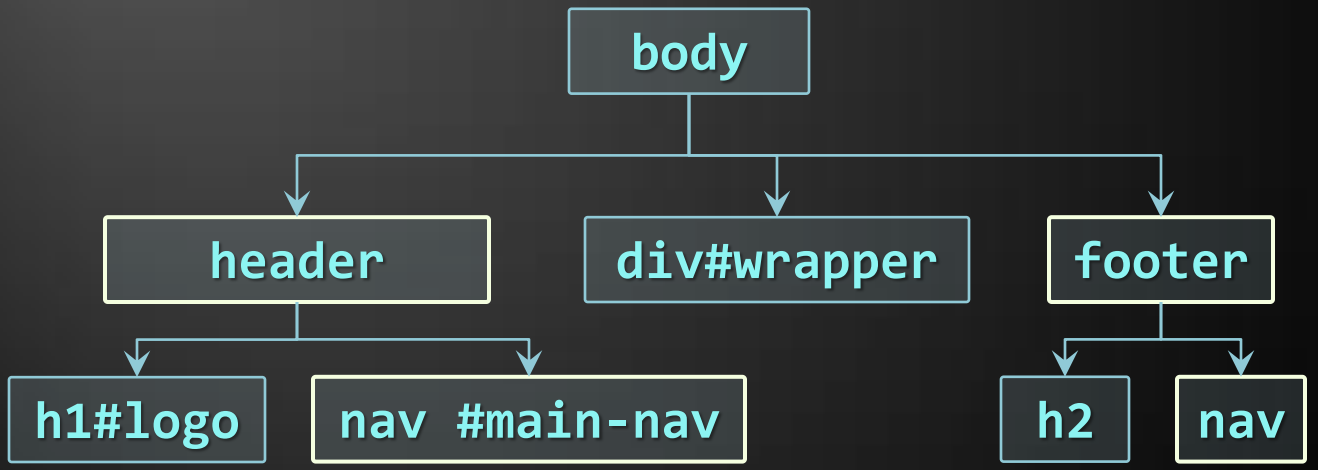
Telerik Academy

DOM Elements

Live Demo

Telerik Academy

Traversing the DOM



Traversing the DOM

- DOM elements have properties about their position in the DOM three
  - Their parent
  - Their children
  - Their siblings
    - Elements before and after the element
- These properties can be used to traverse through the DOM

Telerik Academy

Traversing the DOM (2)

- element.parentNode
  - Returns the direct parent of the element
  - The parent of document is null
- element.childNodes
  - Returns a nodeList of all the child nodes
    - Including the text nodes (whitespaces)

Telerik Academy

Traversing the DOM - Example

```
var trainersList =
  document.getElementsByClassName("trainers-list")[0];

var parent = trainersList.parentNode;
log("parent of trainers-list: " + parent.nodeName + "
     with id: " + parent.id);

var children = trainersList.childNodes;
log("elements in trainers-list: " + children.length);

log("element in trainers-list");
for (var i = 0, len = children.length; i < len; i++) {
  var subItem = children[i]
  log(subItem.nodeName + " content: " +
      subItem.innerText);
}
```

Telerik Academy

Traversing the DOM

Live Demo

Telerik Academy

Using the Named Elements in DOM Objects

- DOM elements have some properties for special elements inside/near them
  - First and last child node
  - The element before/after the current node
- The named elements are:
  - firstChild and lastChild
  - nextSibling / nextElementSibling
  - previousSibling / previousElementSibling

Telerik Academy

Using the Named Elements in DOM Objects

Live Demo

Telerik Academy

Manipulating the DOM

Making a web page dynamic

Manipulating the DOM

- DOM can be manipulated dynamically with JS
  - HTML elements can be created
  - HTML elements can be removed
  - HTML elements can be altered
    - Change their content
    - Change their styles
    - Change their attributes

Creating HTML elements

- The document object has a method for creation of HTML elements
  - `document.createElement(elementName)`
  - Returns an object with the corresponding HTML element type

```
var liElement = document.createElement("li");
 console.log(liElement instanceof HTMLLIElement); //true
 console.log(liElement instanceof HTMLElement); //true
 console.log(liElement instanceof HTMLDivElement); //false
```

Telerik Academy

Creating HTML Elements (2)

- After an HTML element is created it can be treated as if it was selected from the DOM
- When HTML elements are created dynamically they are just JavaScript objects
  - They are still not in the DOM (the web page)
  - New HTML elements must be appended to DOM

```
var studentsList = document.createElement("ul");
var studentLi = document.createElement("li");
studentsList.appendChild(studentLi);
document.body.appendChild(studentsList);
```

Telerik Academy

Appending Elements to the DOM

Live Demo

Telerik Academy

Inserting Elements Before/After Other Element

- The DOM API supports inserting a element before or after a specific element
  - `appendChild()` inserts the element always at the end of the DOM element
  - parent.insertBefore(newNode, specificElement)

Telerik Academy

Inserting Elements After/Before Other Elements

Live Demo

Removing Elements

- Elements can be removed from the DOM
  - Using `element.removeChild(elToRemove)`
  - Pass the element-to-remove to their parent

```
var trainers = document.getElementsByTagName("ul")[0];
var trainer = trainers.getElementsByTagName("li")[0];
trainers.removeChild(trainer);

//remove a selected element
var selectedElement = //select the element
selectedElement.parentNode.removeChild(selectedElement);
```

Telerik Academy

Removing Elements

Live Demo

Altering the Elements

- DOM elements can be remove and/or changed
  - Both the node's children and the node itself
- With the DOM API each DOM element node can be altered
  - Change its properties
  - Change its appearance

Telerik Academy

Altering the Elements (2)

- Keep in mind that each HTML element is unique in the DOM
  - If JavaScript changes its appearance or its position, it is still the same element object

```
<div id="f"><p id="the-p">text</p></div>
<div id="s"></div>
…
var second = document.getElementById("s");
var theP = document.getElementById("the-p");
second.appendChild(theP);
…
//the DOM is:
<div id="f"></div>
<div id="s"><p id="the-p">text</p></div>
```

Telerik Academy

Altering HTML Elements

Live Demo

Altering the Style

- The style of each HTML element can be altered using JavaScript
  - Meaning changing the style attribute
    - The inline styles, not CSS

```
var div = document.getElementById("content");
div.style.display = "block";
div.style.width = "123px";
```

Do not forget the unit

Telerik Academy

Altering HTML Element Style

Live Demo

Telerik Academy

DOM Optimizations

Telerik Academy

Appending DOM Elements

- The DOM API provides a method for appending DOM elements to a element
  - The `appendChild()` method
- `parentNode.appendChild(node)` appends the DOM element `node` to the DOM element `parentNode`
  - If `parentNode` is appended to the DOM, the `node` is also appended

Telerik Academy

Optimizing the Appending of Elements

- Appending elements to the DOM is a very slow operation
  - When an elements is appended to the DOM, the DOM is rendered anew
  - All newly created elements must be appended together
- Here comes the `DocumentFragment` element
  - It is a minimal DOM element, with no parent
  - It is used to store ready-to-append elements and append them at once to the DOM

Telerik Academy

Appending Elements (2)

- Using `DocumentFragment`
  - Append the elements to a `DocumentFragment`
  - Appending `DocumentFragment` to the DOM appends only its child elements
  - http://jsperf.com/append-doc-fragment/2

```
dFrag = document.createDocumentFragment();

dFrag.appendChild(div);
…

document.body.appendChild(dFrag);
```

Telerik Academy

DocumentFragment

Live Demo

Telerik Academy

Faster Creation of Elements

- Creating a DOM element is a slow operation
  - Create the element
  - Set its content
  - Set its style
  - Set its attributes
- This is an issue when creating many elements that have a common structure
  - Only one thing is different for all elements

Telerik Academy

Faster Creation of Elements

- Creating a dynamic list of elements
  - All of the LI elements have the same classes, styles, attributes
  - Only the innerHTML is different
- DOMElement.cloneNode(true) can be used
  - Creates a full copy (deep copy) of the element

```
var clonedNode = someElement.cloneNode(true)
```

Telerik Academy

Faster Creation of Elements

Live Demo

Telerik Academy

DOM and DOM Manipulation

Questions?

http://academy.telerik.com

Homework

1. Write a script that creates a number of div elements. Each div element must have the following
   - Random width and height between 20px and 100px
   - Random background color
   - Random font color
   - Random position on the screen (position:absolute)
   - A strong element with text "div" inside the div
   - Random border radius
   - Random border color
   - Random border width between 1px and 20px

Telerik Academy

Homework (2)

2. Write a script that creates 5 div elements and moves them in circular path with interval of 100 milliseconds
3. Create a text area and two inputs with type="color"
   - Make the font color of the text area as the value of the first color input
   - Make the background color of the text area as the value of the second input

Homework (3)

4. Create a tag cloud:
   - Visualize a string of tags (strings) in a given container
   - By given minFontSize and maxFontSize, generate the tags with different font-size, depending on the number of occurrences

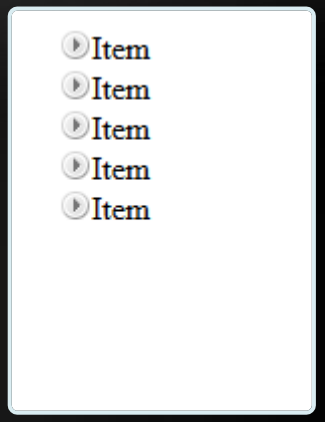
```
var tags = ["cms", "javascript", "js",
"ASP.NET MVC", ".net", ".net", "css",
"wordpress", "xaml", "js", "http", "web",
"asp.net", "asp.net MVC", "ASP.NET MVC",
"wp", "javascript", "js", "cms", "html",
"javascript", "http", "http", "CMS"]

var tagCloud = generateTagCloud(tags,17,42);
```

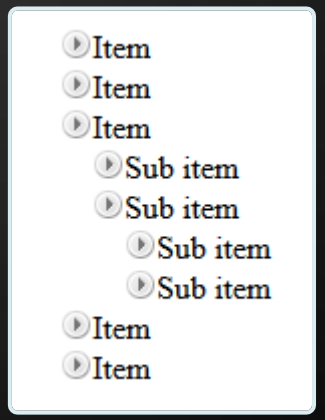
Telerik Academy

Homework (4)

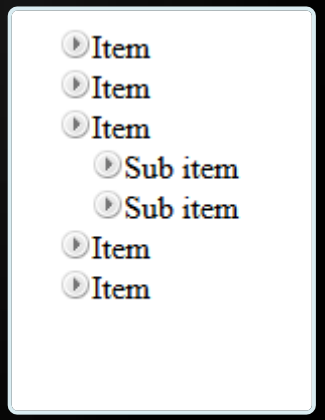
5. *Create a TreeView component
   - Initially only the top items must be visible
   - On item click
     - If its children are hidden (collapsed), they must be made visible (expanded)
     - If its children are visible (expanded), they must be made hidden (collapsed)
   - Research about events



Initial



Sub item expanded



Top level expanded