psounis

ΠΕΡΙΕΧΟΜΕΝΑ:

1. Εξοικείωση με τα Αντικείμενα του DOM
2. Node vs Element
    1. Properties Πλοήγησης στο DOM
    2. Properties Πληροφοριών Κόμβου
    3. Άλλα Properties του Node
3. Μέθοδοι του Node
    1. Προσθήκη/Διαγραφή Παιδιών
        1. Παραδείγματα
    2. Μέθοδοι Σύγκρισης Κόμβων

Το αντικείμενο document:

- Μοντελοποιεί τον κόμβο - ρίζα. Αλυσίδα Πρωτοτύπων:
  - **HTMLDocument:** Διατηρείται για λόγους συμβατότητας
  - **Document:** Μέθοδοι για events κ.α.
  - **Node:** Είναι κόμβος του Δένδρου. Μέθοδοι Επιλογής π.χ. με σχέσεις συγγένειας
  - **EventTarget:** Μπορεί να είναι στόχος σε events
  - **Object:** Το γενικό αντικείμενο της JS

Ένα Αντικείμενο - Στοιχείο (π.χ. ένα div):

- Μοντελοποιεί ένα Element. Αλυσίδα Πρωτοτύπων:
  - **HTMLDivElement:** Ιδιότητες του div
  - **HTMLElement:** Ιδιότητες ως στοιχείο HTML
  - **Element:** Ιδιότητες ως στοιχείο
  - **Node:** Είναι κόμβος του Δένδρου
  - **EventTarget:** Μπορεί να είναι στόχος σε events
  - **Object:** Το γενικό αντικείμενο της JS

Ένα Αντικείμενο - Attribute (π.χ. το charset="utf-8"):

- Μοντελοποιεί ένα Attribute. Αλυσίδα Πρωτοτύπων:
  - **Attr:** Ιδιότητες ενός attribute
  - **Node:** Είναι κόμβος του Δένδρου
  - **EventTarget:** Μπορεί να είναι στόχος σε events
  - **Object:** Το γενικό αντικείμενο της JS

Παρατηρήσεις:

- Βλέπουμε ότι όλα τα στοιχεία είναι Nodes (Κόμβοι του Δενδρου)
  - Και το αντικείμενο NodeList (βλ. προηγούμενο μάθημα) είναι ένα array-like object που περιέχει στοιχεία που έχουν στην αλυσίδα πρωτοτύπων του το Node
- Ενώ τα στοιχεία HTML είναι όλα Element
  - Η εξειδίκευση HTMLElement διαχωρίζει τα στοιχεία από τα SVGElement
  - Και το αντικείμενο HTMLCollection (βλ. προηγούμενο μάθημα) είναι ένα array-like object που περιέχει στοιχεία που έχουν στην αλυσίδα πρωτοτύπων του το HTMLElement

Παράδειγμα 1: dom_example

```
<!DOCTYPE html>
<html>
…
<body>
  <h1>Hello World!</h1>
  <p>Welcome to my page</p>
  <script>
    console.dir(document);
    console.dir(document.querySelector("p"));
    console.dir(document.querySelector("meta").getAttributeNode("charset"));
  </script>
</body>
</html>
```

Άσκηση: Εξερευνήστε την αλυσίδα πρωτοτύπων του h1

ΜΑΘΗΜΑ 1.2: NODE

2. Node vs Element

JS Browser psounis YouTube

Node:

- Αντικείμενο που αναπαριστά έναν κόμβο στο δένδρο:
  - Δεν είναι μόνο τα στοιχεία HTML, αλλά και όλοι οι υπόλοιποι κόμβοι (π.χ. σχόλια, χαρακτηριστικά, κείμενο, doctype κ.α.)

Element:

- Έχει ως πρωτότοτυπο το Node
- Αναπαριστά ένα αντικείμενο που έχει tag και μπορεί να περιέχει χαρακτηριστικά:
  - Όλα τα στοιχεία περιεχομένου HTML (π.χ. body, div, p κ.λπ) είναι Elements
    - Συνεπώς είναι και Nodes (Αφού το πρωτότυπο του Element είναι Node)

Περισσότερα για το Node:

- Στο μάθημα αυτό θα μελετήσουμε περαιτέρω το αντικείμνο Node. Μεταξύ άλλων, περιέχει:
  - Properties και Μεθόδους για τις συσχετίσεις στο DOM (αδέλφια, παιδιά κ.λπ.)
  - Properties πληροφοριών του κόμβου (όνομα, τύπος κ.λπ.)
  - Μεθόδους που ελέγχουν συσχετίσεις του κόμβου με άλλους κόμβους (π.χ. αν ο κόμβος περιέχει έναν άλλο)
  - Μεθόδους που προσθαφαιρούν events επί του κόμβου.

Παράδειγμα 2: node-vs-element.html

```
<!DOCTYPE html>
<html>

<head>
 <meta charset="utf-8">
 <title>My Page</title>
</head>

<body>
 <h1>Hello World!</h1>
 <p class="red">Welcome to my page</p>
 <!-- comment -->

 <script>
    let doctype = document.childNodes[0];
    console.log(doctype instanceof Node, doctype instanceof Element);

    let body = document.querySelector("body");
    console.log(body instanceof Node, body instanceof Element);

    let p = document.querySelector("p");
    console.log(p instanceof Node, p instanceof Element);

    let comment = body.childNodes[5];
    console.log(comment instanceof Node, comment instanceof Element);
 </script>
</body>

</html>
```

```
true false
true true
true true
true false
```

Properties Node που αφορούν συσχετίσεις:

- Με τα ακόλουθα properties, παίρνουμε τους κόμβους που σχετίζονται με τον κόμβο:

Property	Επεξήγηση
parentNode	Κόμβος - γονέας στο DOM
childNodes	NodeList με τα παιδιά του κόμβου
firstChild	Πρώτο παιδί του κόμβου
lastChild	Τελευταίο παιδί του κόμβου
previousSibling	Προηγούμενος κόμβος-αδελφός του τρέχοντος κόμβου
nextSibling	Επόμενος κόμβος-αδελφός του τρέχοντος κόμβου

- Όλα τα παραπάνω properties, επιστρέφουν οποιοδήποτε τύπο κόμβου (π.χ. και σχόλια κ.λπ.). Αν θέλουμε μόνο HTML στοιχεία, χρησιμοποιούμε τα properties (ανήκουν στο Element):

Property	Επεξήγηση
parentElement	HTML Element - γονέας του κόμβου
children	HTMLCollection με τα παιδιά του κόμβου
firstElementChild	Πρώτο HTML Element - παιδί του κόμβου
lastElementChild	Τελευταίο HTML Element - παιδί του κόμβου
previousElementSibling	Προηγούμενο HTML Element του τρέχοντος κόμβου
nextElementSibling	Επόμενο HTML Element του τρέχοντος κόμβου

Παράδειγμα 3: node-relations.html

```
<!DOCTYPE html>
<html>
...
<body>
<h1>Hello World!</h1>
<p class="red">Welcome to my page</p>
<!-- comment -->
<script>
   let body = document.querySelector("body");
   let children = body.children;
   for (let child of children)
      console.log(child);
   console.log("-".repeat(20));

   children = body.childNodes;
   for (let child of children)
      console.log(child);
   console.log("=".repeat(20));

   let p = document.querySelector("p");
   console.log(p.previousElementSibling, p.nextElementSibling);
   console.log(p.previousSibling, p.nextSibling);
   console.log("=".repeat(20));

   console.log(body.firstChild, body.lastChild);
   console.log(body.firstElementChild, body.lastElementChild);
   console.log("=".repeat(20));
   console.log(p.parentElement, p.parentNode);
</script>
</body>
</html>
```

Properties Node που αφορούν πληροφορίες του κόμβου:

- Με τα ακόλουθα properties, παίρνουμε πληροφορίες για τον κόμβο:

Property	Επεξήγηση
nodeType	Ακέραιος για κάθε τύπο κόμβου: • Element: 1 • Attribute: 2 • Text: 3 • Document: 9
nodeName	Όνομα Κόμβου: • Element: Επιστρέφει το tag • Attribute: Επιστρέφει το όνομα του attribute • Text: Επιστρέφει «#text» • document: Επιστρέφει «#document»
nodeValue	Τιμή του κόμβου: • Element: null • Attribute: Τιμή του Attribute • Text: Επιστρέφει το text • Document: null

Παράδειγμα 4: node-info.html

```
<!DOCTYPE html>
<html>
...
<body>
<h1>Hello World!</h1>
<p class="red">Welcome to my page</p>
<!-- comment -->

<script>
   let body = document.querySelector("body");
   console.log(body.nodeType, body.nodeName, body.nodeValue);
   console.log("-".repeat(20));

   console.log(document.nodeType, document.nodeName, document.nodeValue);
   console.log("-".repeat(20));

   let p = document.querySelector("p");
   let attr = p.getAttributeNode("class");
   console.log(attr.nodeType, attr.nodeName, attr.nodeValue);
   console.log("-".repeat(20));

   let text = p.firstChild;
   console.log(text.nodeType, text.nodeName, text.nodeValue);
   console.log("-".repeat(20));

</script>
</body>

</html>
```

Άλλα properties ως Node:

- Επίσης βρίσκουμε τα ακόλουθα properties:

Property	Επεξήγηση
baseURI	Απόλυτο URL του document που περιέχει τον κόμβο
isConnected	Boolean: Αν ο κόμβος έχει συνδεθεί στο document (false, π.χ. όταν δημιουργούμε νέο κόμβο και δεν τον έχουμε ακόμη συνδέσει)
ownerDocument	Επιστρέφει το αντικείμενο document στο οποίο ανήκει το στοιχείο
textContent	Αναπαριστά το text που περιέχεται στον κόμβο. Εφόσον τεθεί με νέα τιμή, αντικαθιστά το περιεχόμενο του κόμβου με τη συμβολοσειρά

Διαφορά textContent και innerText (του Element):

- Η textContent επιστρέφει το περιεχόμενο οποιουδήποτε στοιχείου (ακόμη και των <script> και <style>). Η innerText δεν δουλεύει με αυτά τα στοιχεία.
- Η textContent δεν κάνει parse την HTML, οπότε βλέπει το περιεχόμενο ως απλό κείμενο. Η innerText, αντίθετα, δεν περιέχει στοιχεία HTML.
- Η textContent δεν επηρεάζεται από στυλ που έχουν αλλοιώσει το περιεχόμενο της HTML, ενώ η innerText επιστρέφει το περιεχόμενο με τις αλλοιώσεις του στυλ.

Παράδειγμα 5: node-other-properties.html

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <title>My Page</title>
</head>

<body>
<h1>Hello World!</h1>
<p class="red">Welcome to my page</p>
<!-- comment -->

<script>
   let body = document.querySelector("body");
   console.log(body.baseURI);

   let p = document.querySelector("p");

   body.textContent = "empty";

   console.log(p.isConnected);
   console.log(p.ownerDocument);

   console.log(body.isConnected);
   console.log(body.ownerDocument);

</script>
</body>

</html>
```

Μέθοδοι του Node - Επεξεργασία DOM:

- Με τις ακόλουθες μεθόδους μπορούμε να προσθέσουμε παιδιά στον κόμβο

Μέθοδος	Επεξήγηση
insertBefore (newNode, beforeNode)	Προσθέτει τον κόμβο NewNode, ως παιδί του κόμβου, πριν τον κόμβο beforeNode
appendChild (newNode)	Προσθέτει τον κόμβο NewNode, ως τελευταίο παιδί του κόμβου

- Με τις ακόλουθες μεθόδους μπορούμε να αφαιρέσουμε/αντικαταστήσουμε παιδιά του κόμβου:

Μέθοδος	Επεξήγηση
removeChild (ChildNode)	Αφαιρεί το ChildNode ως παιδί του κόμβου
replaceChild (newNode, oldNode)	Αντικαθιστα το παιδί oldNode με το παιδί newNode

- Με την ακόλουθη μέθοδο μπορούμε να κατασκευάσουμε ένα αντίγραφο του κόμβου:

Μέθοδος	Επεξήγηση
cloneNode(boolean)	Επιστρέφει ένα αντίγραφο του κόμβου. Αν το όρισμα είναι true, αντιγράφει και τα παιδιά του κόμβου

Παράδειγμα 5: insert-node.html

```
<!DOCTYPE html>
<html>
...

<body>
 <h1>h1</h1>

 <div>
  <p>Paragraph <span>1</span></p>
 </div>

 <script>
  // get the parent
  let parent = document.querySelector("div");

  // get the node
  let node = document.querySelector("p");

  // make a clone
  let newNode = node.cloneNode(true);
  newNode.textContent = "Paragraph 2"

  parent.appendChild(newNode);

  // and another one
  newNode = node.cloneNode(true);
  newNode.textContent = "Paragraph 3"

  parent.insertBefore(newNode, node);
 </script>
</body>

</html>
```

Παράδειγμα 6: remove-replace-nodes.html

```
<!DOCTYPE html>
<html>
…
<body>
 <h1>h1</h1>

 <div>
  <p>Paragraph 1</p>
  <p>Paragraph 2</p>
  <p>Paragraph 3</p>
 </div>

 <script>
  // get the parent
  let parent = document.querySelector("div");

  // delete first paragraph
  let firstParagraph = document.querySelector("p:first-child");
  parent.removeChild(firstParagraph);

  // replace last paragraph
  let lastParagraph = document.querySelector("p:last-child");
  let newParagraph = lastParagraph.cloneNode(true);
  newParagraph.textContent = "New Paragraph"
  parent.replaceChild(newParagraph, lastParagraph);
 </script>
</body>

</html>
```

Παράδειγμα 7: clone-node.html

```
<!DOCTYPE html>
<html>
…
<body>
 <h1>h1</h1>

 <div>
  <p>Paragraph <span>1</span></p>
 </div>

 <script>
  // get the parent
  let parent = document.querySelector("div");

  // get the node
  let node = document.querySelector("p");

  // clone the node -> without descendants
  let nodeNoDescendants = node.cloneNode(false);
  // clone the node -> with descendants
  let nodeDescendants = node.cloneNode(true);

  // add the nodes
  parent.appendChild(nodeNoDescendants);
  parent.appendChild(nodeDescendants);
 </script>
</body>

</html>
```

Μέθοδοι του Node - Σύγκριση Κόμβων:

- Με τις ακόλουθες μεθόδους συγκρίνουμε δύο κόμβους:

Μέθοδος	Επεξήγηση
isEqualNode(node)	True, αν ο κόμβος είναι «ίσος» με τον node (περιέχει τα ίδια attributes και τους ίδιους απογόνους)
isSameNode(node)	True, αν ο κόμβος είναι «ίδιο» με τον node (έλεγχος ότι είναι το ίδιο αντικείμενο με τον τελεστή ===)

- Με την ακόλουθη μέθοδο ελέγχουμε αν ο κόμβος περιέχει ως απόγονο έναν άλλο κόμβο:

Μέθοδος	Επεξήγηση
contains(child)	True, αν ο κόμβος περιέχει το child ως απόγονο

- Ενώ με την ακόλουθη μέθοδο, συγκρίνουμε δύο κόμβους για τη σχετική τους θέση στο document:

Μέθοδος	Επεξήγηση
compareDocumentPosition(node)	Επιστρέφει bitmask (μέλος του Node βλ. και βίντεο)

- Οι σταθερές είναι:
  - DOCUMENT_POSITION_CONTAINS
  - DOCUMENT_POSITION_CONTAINED_BY
  - DOCUMENT_POSITION_DISCONNECTED (όχι ίδιο document)
  - DOCUMENT_POSITION_PRECEDING (κόμβος πριν από τον node)
  - DOCUMENT_POSITION_FOLLOWING (κόμβος μετά από τον node)

Παράδειγμα 8: node_compare.html

```
<body>
<h1>Hello World!</h1>
<p class="red">Welcome to my <span>page</span></p>
<p class="red">Welcome to my <span>page</span></p>
<!-- comment -->

<script>
   let p = document.querySelectorAll("p");
   p1 = p[0];
   p2 = p[1];
   console.log(p1.isSameNode(p2));
   console.log(p1.isEqualNode(p2));
   console.log("-".repeat(20));

   let body = document.querySelector("body");
   let span = document.querySelector("span");
   let h1 = document.querySelector("h1")
   console.log(body.contains(span));
   console.log(span.contains(body));
   console.log(h1.contains(span));
   console.log("-".repeat(20));

   let result = body.compareDocumentPosition(span);
   let bitmask1 = Node.DOCUMENT_POSITION_CONTAINED_BY;
   let bitmask2 = Node.DOCUMENT_POSITION_FOLLOWING;
   console.log(result, bitmask1, bitmask2);
   console.log(result & bitmask1);
   console.log(result & bitmask2);
   console.log((result & bitmask1)!=0);
   console.log((result & bitmask2)!=0);
</script>
```