



Browser

ΜΑΘΗΜΑ 1.4
ELEMENT: DOM

ΠΕΡΙΕΧΟΜΕΝΑ:

1. Περιεχόμενο
 1. Επιθέσεις XSS
2. Πλοήγηση στο DOM
 1. Διαγραφή Στοιχείου
 2. Δημιουργία - Αντιγραφή Στοιχείου
 3. Αντικατάσταση Στοιχείου
 4. Εισαγωγή Παιδιών

ΜΑΘΗΜΑ 1.4: ELEMENT: DOM

1. Element: Περιεχόμενο

Properties του Element:


- Βρίσκουμε τα εξής βασικά properties

Property	Επεξήγηση
tagName	(read) Επιστρέφει το όνομα του tag
innerHTML	(read-write) Επιστρέφει/Θέτει τον κώδικα HTML που περιέχει το στοιχείο (κατασκευάζοντας κόμβους κ.οκ.)
outerHtml	(read-write) Επιστρέφει/Θέτει τον κώδικα HTML που συμπεριλαμβάνει το στοιχείο (κατασκευάζοντας κόμβους κ.οκ.)
innerText	(read-write) Αναπαριστά το text που περιέχεται στο στοιχείο, όπως εμφανίζεται στον browser.

Διαφορά textContent (του Node) 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 1: inner-outer.html




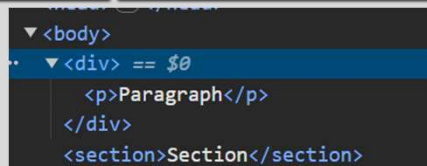
```
<div></div>
<div></div>
```

```
let div1 = document.querySelector("div:first-of-type");
let div2 = document.querySelector("div:last-of-type");


console.log(div1.tagName);

div1.innerHTML = "<p>Paragraph</p>";
div2.outerHTML = "<section>Section</section>";
```







Παράδειγμα 2: innerText-textContent.html



```
<div>Text
<p>Paragraph</p>
</div>
```

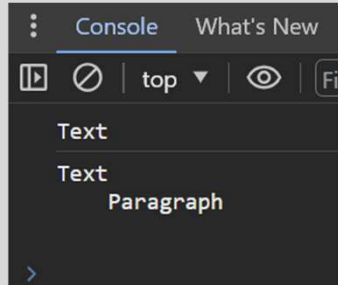


```
p {
  display: none;
}
```



```
let div = document.querySelector("div");

console.log(div.innerText);
console.log(div.textContent);
```



Επιθέσεις XSS:

- XSS (Cross-Site Scripting): Τύπος επίθεσης όπου ο «κακός» φυτεύει κώδικα JavaScript, όταν έχει την ευκαιρία να εισάγει δεδομένα στην εφαρμογή μας.
 - Παράδειγμα:
 - Η εφαρμογή μας έχει ένα πεδίο κειμένου που ο χρήστης συμπληρώνει π.χ. το όνομά του.
 - Έπειτα η εφαρμογή μας προβάλλει αυτό που έβαλε ο χρήστης (π.χ. σε μία επόμενη σελίδα)
 - Ο χρήστης εισάγει (αντί για το όνομα του) κώδικα JS γράφοντας `<script>...</script>` και ο κώδικας εκτελείται στην επόμενη σελίδα.
- Οι `innerHTML`, `outerHtml` μπορεί να χρησιμοποιηθούν σε XSS επιθέσεις.
- Οι `textContent`, `innerText` δεν χρησιμοποιούνται, γιατί βλέπουν τον κώδικα σαν απλό κείμενο
- Εναλλακτικές λύσεις:
 - Χρησιμοποιούμε την `textContent` ή την `innerText` για να προβάλλουμε περιεχόμενο που ενδέχεται να είναι επικίνδυνο (ως είσοδος χρήστη)
 - Άλλες λύσεις: Να κάνουμε `escape` (π.χ. οι χαρακτήρες `<` να αντικαθίστανται από `<`) και `sanitize` (αφαίρεση JS, HTML tags κ.λπ) από την είσοδο

Παράδειγμα 3: xss-attack.html

```
<body>
<input type="text">
<button>Get Data</button>
<button>Fill with XSS attack</button>
<p></p>
<p></p>
<script>
  let input = document.querySelector("input[type='text']");
  let button = document.querySelector("button:first-of-type");
  let button2 = document.querySelector("button:last-of-type");

  let paragraph1 = document.querySelector("p:first-of-type");
  let paragraph2 = document.querySelector("p:last-of-type");

  button.addEventListener("click", ()=>{
    let userData = input.value;
    paragraph1.innerHTML = userData;
    paragraph2.innerText = userData;
  })

  button2.addEventListener("click", ()=>{
    let attack = "<script> " +
      " let body=document.querySelector('body'); " +
      " body.innerHTML = '<p>I AM IN CONTROL</p>'; ";
    input.value = attack;
  })
</script>
</body>
```

Σημείωση:

- Βλέπε βίντεο, γιατί με βάση τις ρυθμίσεις του Chrome, αυτή η επίθεση πιθανότατα θα αποτύχει.

ΜΑΘΗΜΑ 1.4: ELEMENT: DOM

2. Element: Πλοήγηση στο DOM

Υπενθύμιση από μάθημα 1.2:

- Το Element περιέχει properties με τις συσχετίσεις με άλλα στοιχεία στο DOM.

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

Πλοήγηση στο DOM:

- Βρίσκουμε στο Element και τις μεθόδους:

Μέθοδος	Επεξήγηση
querySelector(query)	Επιστρέφει NodeList το πρώτο Element που ταιριάζει με το CSS query
querySelectorAll(query)	Επιστρέφει NodeList που περιέχει όλα τα Element που ταιριάζουν με το CSS query

- (δουλεύουν με τον ίδιο ακριβώς τρόπο με τις αντίστοιχες μεθόδους του document, με τη διαφορά ότι το ερώτημα γίνεται στο στοιχείο και τα περιεχόμενά του)

Παράδειγμα 4: querySelector.html

```
<p>Paragraph 1</p>
<p>Paragraph 2</p>
<p>Paragraph 3</p>
<div>
  <p>Paragraph 4</p>
  <p>Paragraph 5</p>
  <p>Paragraph 6</p>
</div>
```



```
div {
  background-color: lightblue;
}
```

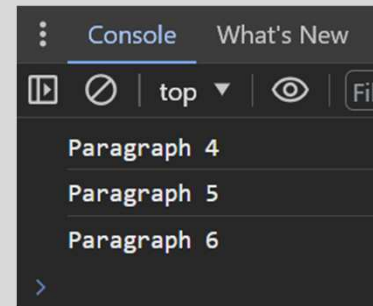
```
let div = document.querySelector("div");

let paragraphs = div.querySelectorAll("p");

for (let paragraph of paragraphs) {
  console.log(paragraph.innerText);
}
```



Paragraph 1
Paragraph 2
Paragraph 3
Paragraph 4
Paragraph 5
Paragraph 6



Διαγραφή Στοιχείου:

- Διαγράφουμε ένα στοιχείο με τις μεθόδους:

Μέθοδος	Επεξήγηση
remove()	Καλείται από το ίδιο το στοιχείο. Το στοιχείο αφαιρείται από το document
removeChild(node)	Καλείται από το γονέα του στοιχείου node, το οποίο και αφαιρείται από το document

Σημείωση:

- Η remove() είναι νεότερη μέθοδος και δεν υποστηρίζεται από παλιούς browsers.

Σημείωση:

- Άλλοι τρόποι για την απόκρυψη/διαγραφή στοιχείων:
 - Θέτοντας το property innerHTML ίσο με την κενή συμβολοσειρά, διαγράφονται όλα τα παιδιά του κόμβου
 - Πολύ συχνά δεν υπάρχει ανάγκη διαγραφής στοιχείων, αλλά προσωρινής απόκρυψής τους. Αυτό επιτυγχάνεται με την τιμή none του property display της CSS

Παράδειγμα 5: remove-element.html

```
<body>
<h1>h1</h1>

<div>
  <p>Paragraph 1</p>
  <p>Paragraph 2</p>
  <p class="toggle">Paragraph 3</p>
</div>

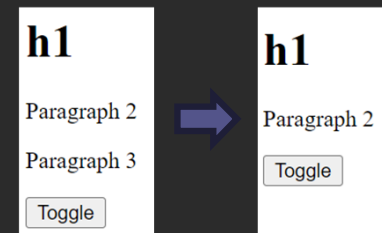
<button>Toggle</button>

<script>
  // delete first paragraph
  let node = document.querySelector("p:first-child");
  node.remove();

  // toggle third paragraph
  document.addEventListener("DOMContentLoaded", function() {
    let toggleButton = document.querySelector("button");

    toggleButton.addEventListener("click", function() {
      let paragraph = document.querySelector("p:last-child");
      console.log(paragraph)

      if (paragraph.style.display === "none") {
        paragraph.style.display = "block";
      } else {
        paragraph.style.display = "none";
      }
    });
  });
</script>
</body>
```



Αντιγραφή Στοιχείου:

- Αν θέλουμε να κατασκευάσουμε ένα αντίγραφο ενός στοιχείου, χρησιμοποιούμε τη μέθοδο (του Node):

Μέθοδος	Επεξήγηση
cloneNode(full)	Καλείται από το στοιχείο που θέλουμε να αντιγράψουμε. Επιστρέφει ένα αντίγραφο του. full: Boolean μεταβλητή: Αν true, αντιγράφονται και όλοι οι απόγονοι του στοιχείου

Σημείωση:

- Αν και θα μελετήσουμε αναλυτικά το document σε επόμενο μάθημα, καλό θα είναι να γνωρίζουμε και τη μέθοδο:

Μέθοδος	Επεξήγηση
createElement(tag)	Κατασκευάζει ένα αντικείμενο τύπου tag

Παράδειγμα 6: clone-element.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);

  // replace the span
  let span = document.querySelector("p:last-child span");
  let newSpan = span.cloneNode(true);
  newSpan.innerText = "2";
  span.parentNode.replaceChild(newSpan, span);
</script>
</body>

</html>
```

h1

Paragraph 1

Paragraph 2

ΜΑΘΗΜΑ 1.4: ELEMENT: DOM

2.3. Αντικατάσταση Στοιχείου

Αντικατάσταση Στοιχείου:

- Στην ακόλουθη μέθοδο, αντικαθιστούμε το στοιχείο με παράθεση άλλων στοιχείων:

Μέθοδος	Επεξήγηση
replaceWith (nodes)	Αντικαθιστά το ίδιο το στοιχείο με τα ορίσματα (στοιχεία, ή συμβολοσειρές κειμένου)

- Ενώ με την ακόλουθη μέθοδο, αντικαθιστούμε τα παιδιά του στοιχείου με παράθεση άλλων στοιχείων:

Μέθοδος	Επεξήγηση
replaceChildren (nodes)	Αντικαθιστά όλα τα παιδιά του στοιχείου με τα ορίσματα (στοιχεία, ή συμβολοσειρές κειμένου)

Υπενθύμιση:

- Επιτυγχάνουμε την αντικατάσταση ενός παιδιού με τη μέθοδο του Node:

Μέθοδος	Επεξήγηση
replaceChild (newNode, oldNode)	Αντικαθιστά το παιδί oldNode με το παιδί newNode

Παράδειγμα 7: replace-element.html

```
<!DOCTYPE html>
<html>

<body>

<div>
  <p>Paragraph 1</p>
  <p>Paragraph 2</p>
</div>
<div>
  <p>Paragraph 3</p>
  <p>Paragraph 4</p>
</div>

<script>

  // make some elements
  let p5 = document.createElement("p");
  p5.innerText = "Paragraph 5";
  let p6 = document.createElement("p");
  p6.innerText = "Paragraph 6";
  let text = "Some additional text";

  // replace first div with a bunch of elements
  let div1 = document.querySelector("div:first-of-type");

  div1.replaceWith(p5, text);

  // replace second div with a bunch of elements
  let div2 = document.querySelector("div:last-of-type");

  div2.replaceChildren(p6, text);
</script>
</body>
```

h1

Paragraph 5

Some additional text

Paragraph 6

Some additional text

Εισαγωγή Παιδιών:

- Μπορούμε να εισάγουμε παιδιά στο στοιχείο με τις μεθόδους:

Μέθοδος	Επεξήγηση
append(...nodes)	Εισάγει τα nodes (elements ή/και strings) ως “τελευταία” παιδιά
prepend(...nodes)	Εισάγει τα nodes (elements ή/και strings) ως “πρώτα” παιδιά
insertAdjacentElement ('afterbegin', element)	Εισάγει το στοιχείο element, ως πρώτο παιδί του στοιχείου
insertAdjacentHTML ('afterbegin', html)	Εισάγει την HTML, ως πρώτο παιδί του στοιχείου
insertAdjacentElement ('beforeend', element)	Εισάγει το στοιχείο element, ως τελευταίο παιδί του στοιχείου
insertAdjacentHTML ('beforeend', html)	Εισάγει την HTML, ως τελευταίο παιδί του στοιχείου

Υπενθύμιση:

- Στο μάθημα 1.2 είδαμε πρόσθετες μεθόδους του αντικειμένου Node για την εισαγωγή παιδιών:

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

Παράδειγμα 8: insert-children.html

```

<body>
  <h1>h1</h1>

  <div>
    <p>Paragraph</p>
  </div>

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

    // Make some paragraphs
    let paragraphs = [];
    for (let i=0; i<=10; i++) {
      paragraphs.push(p.cloneNode(true));
      paragraphs[i].innerText = "Paragraph " + i;
    }

    parent.append(paragraphs[0], paragraphs[1]);
    parent.prepend(paragraphs[2], paragraphs[3]);
    parent.insertAdjacentElement('afterbegin', paragraphs[4]);
    parent.insertAdjacentHTML('afterbegin', "<p>Paragraph A</p>");
    parent.insertAdjacentElement('beforeend', paragraphs[5]);
    parent.insertAdjacentHTML('beforeend', "<p>Paragraph B</p>");

    parent.appendChild(paragraphs[6]);
    parent.insertBefore(paragraphs[7], paragraphs[2]);
  </script>
</body>

```