ELŐFELTÉTELEK

A tananyag megértéséhez és elsajátításához szükséges előzetes ismeretek

- általános webes felhasználói ismeretek
- HTML nyelv ismerete
- CSS nyelv ismerete
- alapvető programozási ismeretek

SZEREPLŐK

A webes működésnek két alapvető szereplője van:

- Kliens
- Szerver

A kliensen jellemzően egy böngésző programot értünk, amelyet egy felhasználó irányít.

A szerveren egy web-szerver alkalmazást értünk, amely szolgáltatja a webes tartalmat. A szerverbe gyakran beleértük a szerveren található egyéb alkalmazásokat, például adatbázis-szervert

TARTALOMTÍPUSOK

Statikus tartalom

Statikus weboldalnak nevezzük az olyan oldalakat, amelyek tartalma állandó, a forráskódban le van írva.

Statikus weboldal készítéséhez nem szükséges programozás, elegendő két jelölő nyelv ismerete: a HTML és a CSS

(Valójában elég a HTML, de korszerű, igényes weboldal készítéséhez szükséges a CSS nyelv.)

TARTALOMTÍPUSOK

Dinamikus tartalom

Dinamikus weboldalnak olyan oldalakat nevezünk, amelyek tartalmát program állatja elő jellemzően egy változó forrásból kiolvasva, például adatbázis, fájl, stb.

A dinamikus weboldal felépítését végző programkódokat két csoportra osztjuk aszerint, hogy hol hajtódik végre:

- kliens oldali program
- szerver oldali program

KLIENS OLDALI PROGRAMOZÁS

A kliens oldali programkódot a böngésző hajtja végre a kliens oldal eszközén. Így csak olyan forrásokat tud kezelni, amelyek a kliens gépén megtalálhatóak. Ebből következik, hogy

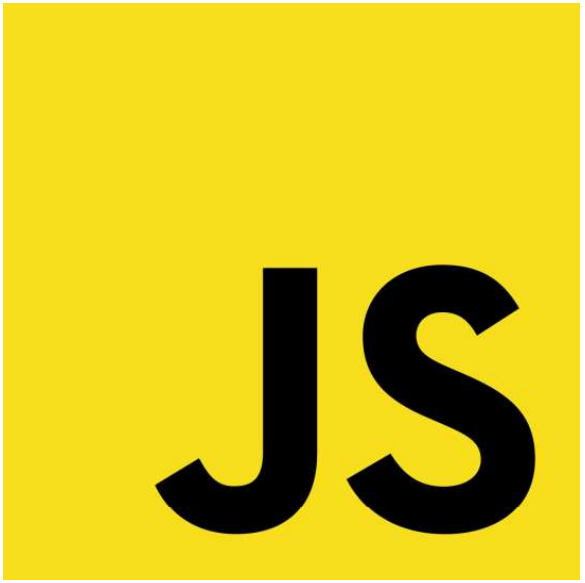
- a kliens eszközt terheli,
- a forráskód a kliens oldalon elérhető,
- az alkalmazás szempontjából kevésbé biztonságos.

A legnépszerűbb kliens oldali programnyelv a *JavaScript*, illetve az erre épülő nyelvek

JAVASCRIPT BEVEZETŐ

Mi a JavaScript?

- A JavaScript egy HTML kódba ágyazott script nyelv
- Végrehajtásához értelmező (interpreter) szükséges
- A JavaScript értelmezőtől független része a Core JavaScript
- A HTML oldalba ágyazott JavaScriptet a böngésző értelmezi



JAVASCRIPT BEVEZETŐ

Hova írható?

A HTML kódban bárhol szerepelhet, akár a `head`, akár a `body` részben

Egy oldalon belül több JavaScript rész is lehet

A `<script> … </script>` tag-ek közé írható

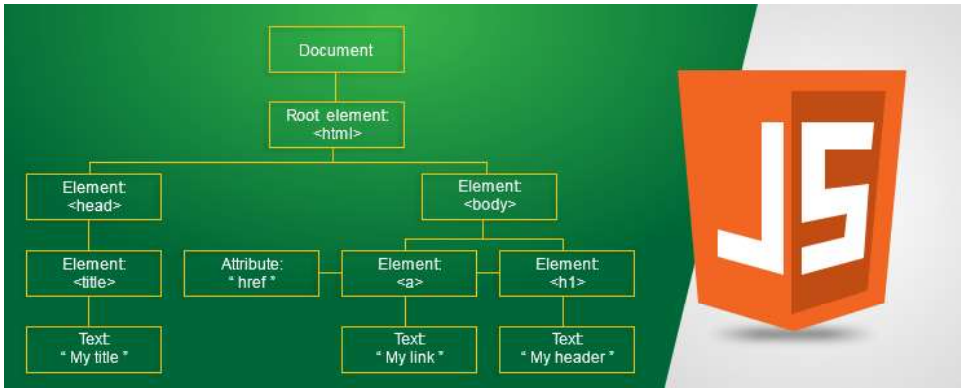
Kerülhet külön állományba, ennek a kiterjesztése jellemzően *.js*

Külső állomány beillesztése a HTML kódba:

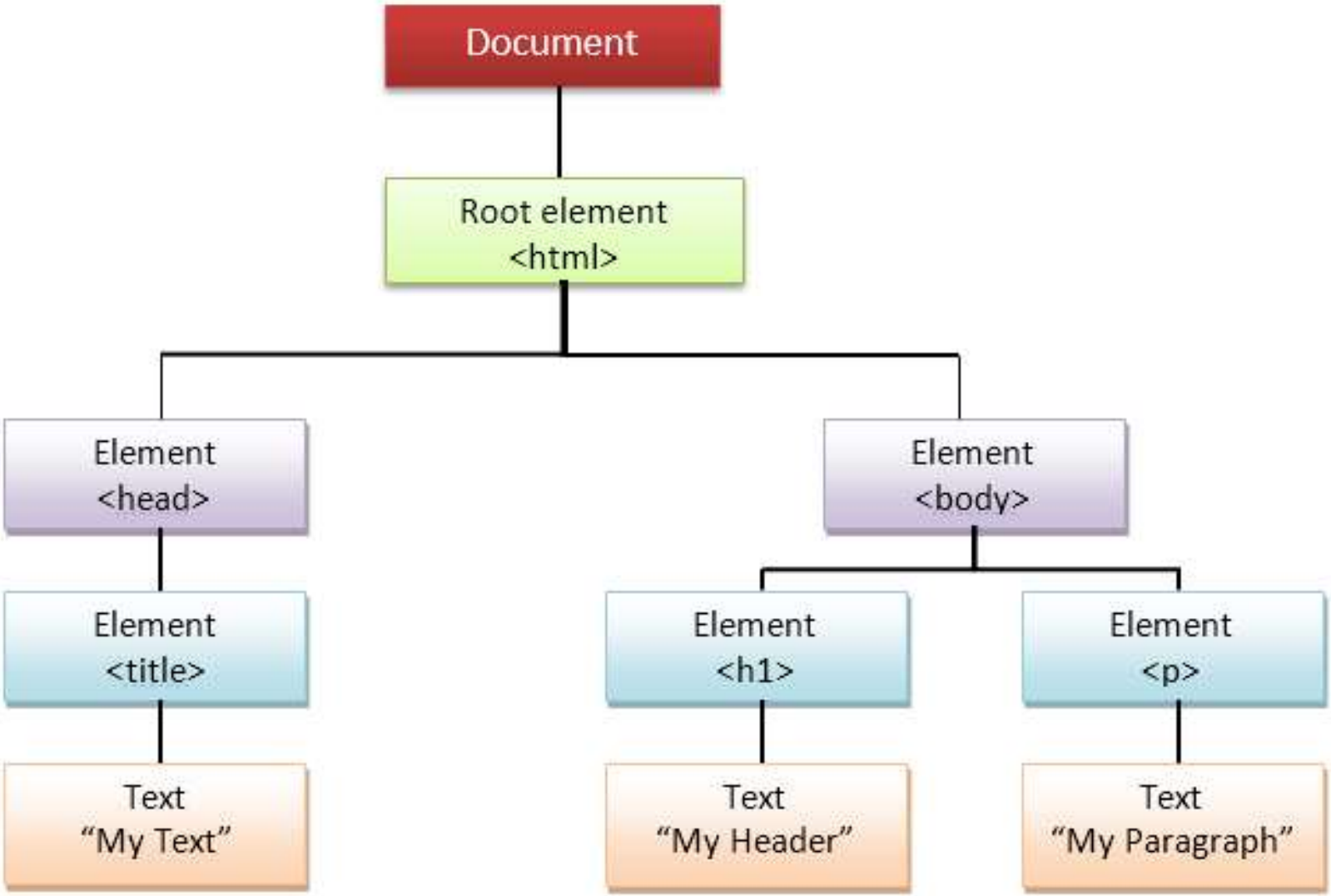
```
<script src="JavaScriptFile.js"></script>
```

DOM

- A **DOM** (Document Object Model) egy objektummodell többek között a HTML formátumához az elemek kezelésére. Leírja a HTML elemek és tartalmak szerkezetét.
- A böngésző ez alapján építi fel a forráskód által megadott HTML oldal szerkezetét.
- Az elemeket egy fa-szerkezetbe építi, ahol a csomópontok az oldal elemei, amelyek között alá és fölérendeltségi viszont definiál (szülő és gyerek)



DOM FELÉPÍTÉS



JQUERY

A jQuery egy szabadon felhasználható JavaScript függvénykönyvtár.

Használatával könnyebben lehet weboldalakhoz olyan műveleteket elkészíteni, amelyek csak JavaScript használatával hosszabb programozás feladata lenne.

A jQuery minden (elismertebb, jelenlegi verziójú) böngészőben egyformán működik pc-n és mobilon egyaránt.

A jQuery open source

JQUERY

jQuery főbb szerepe

- HTML elemek (DOM) kezelése
- Stílusok (CSS) kezelése
- HTML események kezelése
- Effektek és animációk
- egyéb

A jQuery-t folyamatosan fejlesztik, bővítik az újabb igényeknek megfelelően.

jQuery használata

Letölthető a jQuery weboldaláról

https://jquery.com/download/

Majd a weboldalhoz hozzákapcsolható, mint lokális JavaScript állomány

```
<script src="jquery-3.3.1.min.js">
</script>
```

Előnye, hogy független más szervertől

jQuery használata

Használható letöltés nélkül a szerverre hivatkozással (CDN – Content Delivery Network)

Előnye, hogy ha már korábban betöltött egy felhasználó olyan oldalt, ahol ez a jQuery használva volt, akkor az állomány megtalálható a böngésző lokális tárterületén, így nem tölti le újra, ezáltal csökkentve a hálózati forgalmat és az oldal betöltődését

JQUERY

Hivatkozható több helyről

Google

```
<script src="https://ajax.googleapis.com/ajax/
libs/jquery/3.3.1/jquery.min.js"></script>
```

Microsoft

```
<script src="https://ajax.aspnetcdn.com/ajax/
jQuery/jquery-3.3.1.min.js"></script>
```

CDNJS

```
<script src="https://cdnjs.cloudflare.com/ajax/
libs/jquery/3.3.1/jquery.min.js"></script>
```

JQUERY

Két fő típusa azonos funkcionalitással

- Teljes verzió – *uncompressed: jquery.js*

Fejlesztéshez van, könnyen olvasható formátumú, kommentezett kód

- Csökkentett méretű – *minimized: jquery.min.js*

Tömörített verzió, ahol törölve vannak a megjegyzések, white-space karakterek, ezáltal csökkentve a letöltés és értelmezés sebességét

Szintaktika

Az utasítás formája

```
$(kijelölő).művelet()
```

Ahol a *kijelölő* meghatározza azt (azokat) az elemet, amivel az adott *műveletet* végre kell hajtani.

Például

```
$(”div.box”).toggle()
```

Kijelölő

A jQuery kijelölés megegyezik a CSS kijelöléssel

Elemkijelölés

```
$(”p”)
```

Osztálykijelölés

```
$(”.osztaly”)
```

Azonosító kijelölés

```
$(”#azon”)
```

Stb. (például)

```
$(”header ul > li.subM”)
```

Események

A jQuery események a böngésző és a weboldallal történt műveletek ”kezelése”

Az események jó része megegyezik a DOM eseményekkel

Az eseményekre műveleteket készítünk, aminek végre kell hajtódnia az esemény bekövetkezésekor, ezeket függvényként adjuk meg

```
$(”button”).click(function(){
    utasítások;
});
```

JQUERY

Célszerű biztosítani, hogy a jQuery kód csak akkor fusson le, amikor az oldal már betöltődött, ne próbáljon egy olyan elemmel műveletet végezni, ami még létre se jött (nem töltődött be)

Minden műveletet be kell tenni egy eseménybe

```
$(”document”).ready(function(){
    …
})
```

Rövidebb formában ugyanez

```
$(function(){
    …
})
```