Tartalom

Alapfogalmak

A Javascript egy **gyengén típusos** és főleg **kliens oldali** programozási nyelv melynek vannak már szerver oldali változatai is. A kód a HTML-be integrált, a kliens onnan olvassa ki, és értelmezi.

Mit is jelentenek ezek a fogalmak?

Gyengén típusos

A változók létrehozásakor nem kell megadnunk annak típusát, csupán a nevét és utalni kell annak változó mivoltára, a típusát a programozás nyelv magától „beállítja” erről később még lesz szó.

Röviden: A változók típusát az aktuális értéke határozza meg.

A szerver és kliens oldal

A kód lefuttatásának helyét jelöli és viszonylag egyértelmű. Szerver oldali nyelvek eseténben szükség van egy azt futtató szerverre, kliens oldali esetében ahogy majd meglátjuk ezt a JavaScriptnél is, nincs szükségünk külső szerverre, a kódunkat egyszerűen akár a böngészőnkből is futtathatjuk.

Szintaktikai hiba (Fordítási hiba)

A szintaktikai hiba során a nyelv FORMAI szabályai ellen vétünk.

Ezt általában egyetlen fordító sem tűri, mert az utasítások pontos végrehajtásához pontos és egzakt bemeneti adatok, valamint hibátlan futtató program szükséges.

*Sokféle formai hiba létezik, egy tipikus JavaScript-hiba például, ha nem megfelelően írunk le egy parancsot pl . (**Math.random** helyett **math.random**-ot használunk), vagy épp a nem minden nyitó zárójelhez van záró elem.*

Szemantikai hiba (Futtatási hiba)

A szemantikai hiba során a nyelv TARTALMI szabályai ellen vétünk.

A **szemantikai hiba** programozáskor egy algoritmus rossz implementálásából ered. Akkor beszélünk szemantikai hibáról, ha a program lefordul, elindul, de nem azt csinálja, amit a programozó szeretne, esetleg a program futása közben hiba miatt meg is áll. Ezt a futtatás közbeni hibát jellemzően a programozó veszi észre tesztelés közben. Ez úgy történik, hogy fejlesztő elindítja a programot, megadja kívánt bemeneti adatokat, ami az algoritmus futásához szükséges, de a kimeneten már nem az jelenik meg, amit a programozó vár, akkor - bár a program ***szintaktikailag*** helyes, ***a fordító elfogadta*** - mégis a program **szemantikai hibát tartalmaz**.

Ilyen hiba lehet, ha mondjuk rossz változóra hivatkozunk a kódban, így a kódunk más értéket dolgozik fel, és nem az elvárt eredményt adja

Importálás

Az importálás **2 féle** módon történhet.

Mindkét módra való hivatkozás a **<script> </script> tag-ek között** történik, mindkét verziót lehet használni a **<head> vagy a <body> tag-ek-belsejében** is.

Az egyszerűbb mód, hogy készítünk egy **külön .js** kiterjesztésű **fájlt** és ebben helyezzük el a kódunkat, erre a weboldal külön hivatkozni kell:

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

Ilyenkor erre a fájlra, máskor is hivatkozhatunk és kódját **több helyen is felhasználhatjuk**, hasonlóan, mint css formázások esetén.
Megadható relatív és abszolút formában az elérési útvonal esetleg konkrétan webes linkként is.
Illetve **annyi külső fájlt adhatunk meg amennyit csak szeretnénk**, egy dologra kell ügyelni.

Fontos a sorrend!

A fájlok beolvasása sorban történik, így, ha egy adott elem, egy későbbi fájlban is megtalálható (pl.: függvény) azonos néven, azt a későbbi file felülírja

Valamint szintén a <script> tagokban közvetlenül elhelyezhetjük a kódunkat, a weboldalunk head vagy épp a body részében,

```
<script>
{
        document.write("Szeretem a programozást");
}
</script>
```

Head vs. Body

Head tag-be akkor helyezzük el a JS kódunkat, ha szeretnénk, hogy az a oldal betöltődése előtt már lefuthasson. (pl.: felugró ablak)

Body tag-be pedig akkor, ha szeretnénk, a tartalmat manipulálni, vele, amihez szükséges, hogy maga az oldal betöltődjön, amennyiben a JS kódunkban olyan elemre hivatkozunk, ami a meghívás pillanaptában még nem létezik, hibára futunk. pl.: a kiíratő parancs nem találja a hivatkozott id-t, vagy épp a beolvasó parancs a létrehozott mezőt, mert arra „túl korán” hivatkozunk!

A legegyszerűbb, ha a body részben, annak is a végére, meghívjuk a JS fájlokat, amiket használunk az oldalon, és minden JavaScript kódot abban helyezünk el. Így elkerülhetjük a meghívásból adódó összes hibalehetőséget.

Megjegyzés, amennyiben külső JavaScipt könyvtárat (pl.: jquery, bootstrap) importálunk, azt nyugodtan megtehetjük a weboldalunk elején a head tag-en belül.