Web és PHP leckék

forrás: www.oktatovideok.hu

Tartalomjegyzék

HTML alapok

Mi az a HTML dokumentum?

A HTML dokumentum egy standard szövegfájl .html vagy .htm kiterjesztéssel, mely speciális formázó utasításokat tartalmaz.

Legegyszerûbben a Notepad (Jegyzettömb) nevû programmal hozhatjuk létre, szerkeszthetjük és menthetjük el. Megjelenítésére leggyakrabban az Internet Explorer-t vagy a Netscape-et használjuk, de a standard HTML kódot sok más internetes böngészõ is értelmezni tudja.

Mi az a „TAG”?

A „TAG” (ejtsd: "teg", továbbiakban egyszerûen tag-nek írjuk) a HTML dokumentumok formázó utasítása. Minden esetben a „kisebb” és „nagyobb” jelek közé kell tenni:

```
<AZONOSITO>
```

Minden tag-nek van nyitó és záró része. A záró részbe az azonosító elé egy „ / ” jelet kell tenni:

```
</AZONOSITO>
```

A tag nyitó részébe az azonosító után megadhatunk ún. paramétereket is. Ezek a paraméterek a tag-re vonatkozó tulajdonságokat állítják be. A paraméter neve és az egyenlőségjel után a paraméter értékét dupla idézőjelek közé kell tenni. A paramétereket egy szóközzel választjuk el egymástól:

```
<AZONOSITO paraméter="érték" paraméter="érték">
```

A tag nyitó és záró része közé kerül az a rész, amelyre a tag vonatkozik. Ha például egy szöveget kék színnel és 5-ös betûnagysággal szeretnénk kiírni:

```
<BETŰ szín="kék" betűnagyság="5"> Szöveget ide... </BETŰ>
```

A HTML dokumentumok alapszerkezete

Minden HTML oldal a következõképpen épül fel:

```
<HTML>
    <HEAD>
        <TITLE> [.......] </TITLE>
    </HEAD>
    <BODY>
        [............]
        [............]
        [............]
    </BODY>
</HTML>
```

Jól látható, hogy minden nyitó tag-nek megvan a záró párja, és hogy mindig azt a tag-et zárjuk le először, amelyiket utoljára megnyitottuk. A HTML oldalak mind ilyen keretes szerkezetűek! Néhány tag esetében a záró tag-et elhagyhatjuk, de erre majd mindig külön felhívjuk a figyelmet!

Most vegyük sorra ezt a 4 alapvető tag-et:

<HTML> : azt jelzi, hogy a közte lévő szöveget a HTML szabvány szerint kell értelmeznie a böngészőnek. Ennek a tag-nek nincsenek paraméterei, és mindig kötelező lezárni.

<HEAD> : ez a HTML oldal fej-része. Sok hasznos információt elhelyezhetünk benne, de mi most csak a következő tag-et tettük bele:

<TITLE> : A HTML oldal címe, vagy ha úgy tetszik fejléce, ami a böngésző fejlécében jelenik meg. Paraméterei nincsenek, és mindig kötelező lezárni!

<BODY *bgcolor="crimson" background="hatterkep.jpg"***>** : a HTML oldal törzse. A nyitó és záró tag közé beírt szöveget a böngésző megpróbálja értelmezni, majd megjeleníteni. Mindig le kell zárni. A következő paramétereket szoktuk megadni:

bgcolor : az oldal háttérszínének hexa kódja (#FF01DC) vagy rövid angol neve
background : a háttérkép neve (útvonal, fájlnév)

Ha azt szeretnénk, hogy a háttérkép középre igazítva, fixen rögzítve jelenjen meg, a következő paramétert kell felvennünk a <BODY> tag nyitó részébe:

```
style="background-position:center; background-repeat:no-repeat; background-attachment:fixed"
```

Bekezdések, sortörés és szóköz

<P *align="center"***>** : új bekezdés (paragraph) nyitása. Mindig le kell zárni! Paraméterként azt szoktuk megadni, hogy a bekezdésen belül balra, középre vagy jobbra legyen igazítva a szöveg (**left**=balra, **center**=középre, **right**=jobbra, **justify**=sorkizárt).

A HTML dokumentumban lévő, egymást követő szóközöket a böngésző egyetlen szóköznek fogja fel. Ugyanez a helyzet a sortöréssel is: hiába kezdünk új sort a szövegszerkesztőben, a böngésző csupán egy szóközt fog beilleszteni a sortörés helyére.

Ahhoz tehát, hogy igazi sortörést illetve sok egymást követő szóközt helyezzünk el a dokumentumban, speciális vezérlő elemekre van szükségünk:

**
** : sortörés. **Nem kell lezárni!**

** ** : egy mesterséges szóköz beszúrása.

Betűtípus megváltoztatása

A következő alaplehetőségek közül választhatunk:

```
<FONT size="5" color="navy" face="arial">
  ... ide jön a szöveg ...
</FONT>
```

A betűtípust a **style** paraméter segítségével is beállíthatjuk:

```
<FONT style="font-size:36; color:navy; font-family:arial">
  ... ide jön a szöveg ...
</FONT>
<B> Vastag betűk. </B>
<I> Dőlt betűk. </I>
<U> Aláhúzott betűk. </U>
```

Képek, linkek, elválasztó vonal

Kép beszúrása: **(Nem kell lezárni!)**

```
<IMG src="kepneve.jpg" border="10" width="100" height="50"
style="border-color:crimson">
```

Link: `<A href="http://www.idemutat.hu"> Link szövege. </A>`

Vízszintes elválasztó vonal: **(Nem kell lezárni!)**

```
<HR color="blue" size="10" width="50%" align="right">
```

Táblázatok

<TABLE> : Egy táblázat beillesztése. A megadható paraméterek:

align="center" : a táblázat balra, középre vagy jobbra igazítása

border="1" : látható vonalaknál 1, láthatatlan vonalak esetében 0

height="80%" : a lehetséges max. magasság hány százalékát töltse ki a táblázat

width="60%" : a lehetséges max. szélesség hány százalékát töltse ki a táblázat

cellspacing="0" : a cellák közti szünet nagysága

cellpadding="0" : a cellákon belüli margó nagysága

<TR> : A táblázaton belül egy új sor beillesztése. A paraméterek:

height="25%" : a sor magassága a táblázat magasságának hány százaléka legyen

bgcolor="lightyellow" : a sor háttérszíne (angol név vagy #hexa kód)

<TD> : A soron belül egy új oszlop (cella) beillesztése. A paraméterek:

align="center" : a cella tartalmának vízszintes igazítása (left, center, right)

valign="top" : a cella tartalmának függőleges igazítása (top, center, bottom)

width="20%" : a cella szélessége a táblázat szélességének hány százaléka legyen

bgcolor="lightblue" : a cella háttérszíne (angol név vagy #hexa kód)

colspan="4" : hány cella helyét foglalja el vízszintesen

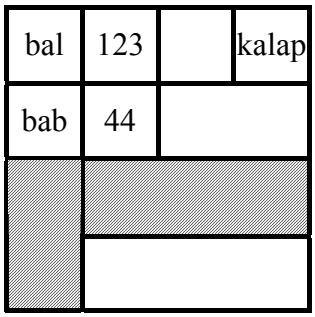
rowspan="2" : hány cella helyét foglalja el függőlegesen

A fenti tag-ek egymásba ágyazva helyezkednek el a kódban: a táblázaton belül vannak a sorok, a sorokon belül vannak az oszlopok (cellák). Ügyeljünk arra, hogy mindig az utoljára megnyitott tag-et zárjuk be először!

Ha egy cellába nem írunk semmit, akkor a keret nem fog rendesen megjelenni. Ezért az üres cellákba mindig tegyünk egy mesterséges szóközt: *** ***

Érdemes a táblázatot kívülről befele létrehozni, tehát előbb a táblázatot szúrjuk be, majd a sorokat, végül a sorokon belül a cellákat! Így nem fogunk eltévedni a táblázat labirintusában...

Vegyünk egy egyszerű példát! Ezt a táblázatot szeretnénk beilleszteni:



Íme a kód:

```
<TABLE align="center" border="1" bordercolor="black"
height="200" width="200" cellspacing=0 cellpadding=0>
  <TR height="25%">
    <TD width="25%" align="center"> bal </TD>
    <TD width="25%" align="center"> 123 </TD>
    <TD width="25%">   </TD>
    <TD width="25%" align="center"> kalap </TD>
  </TR>
  <TR height="25%">
    <TD align="center"> bab </TD>
    <TD align="center"> 44 </TD>
    <TD colspan="2">   </TD>
  </TR>
  <TR height="25%">
    <TD rowspan="2" bgcolor="yellow">   </TD>
    <TD colspan="3" bgcolor="yellow">   </TD>
  </TR>
  <TR height="25%">
    <TD colspan="3">   </TD>
  </TR>
</TABLE>
```

Keretes szerkezetű oldalak

Kétféleképpen készíthetünk keretes szerkezetű oldalakat.

1. Táblázaton belüli belső keretek (IFRAME = Inline Frame) segítségével:

```
<TABLE>
  <tr height="80%">
    <td align=center valign=top rowspan=2>
      <a href="egy.htm" target="fokeret">Menü 1</a>
      <br>
      <a href="ketto.htm" target="fokeret">Menü 2</a>
    </td>
    <td align=center>
      <iframe name="fokeret" src="egy.htm" width="100%"
      height="100%" frameborder=0 framespacing=0 border=0
      marginheight=0 marginwidth=0 tabindex=0></iframe>
    </td>
  </tr>
</TABLE>
```

2. Hagyományos keretek (FRAMESET) használatával:

```
<FRAMESET cols="150, *" framespacing=0 border=0>
  <frameset rows="50, *">
    <frame name="logo" src="logo.htm" frameborder=0
     scrolling=no>
    <frame name="menu" src="menu.htm" frameborder=0
     scrolling=no>
  </frameset>
  <frameset rows="50, *, 25">
    <frame name="fejlec" src="fejlec.htm" frameborder=0
     scrolling=no>
    <frame name="fooldal" src="elso.htm" frameborder=0>
    <frame name="labjegyzet" src="labjegyzet.htm"
     frameborder=0 scrolling=no>
  </frameset>
</FRAMESET>
```

HTML űrlapok

Mi az a form?

A form tulajdonképpen egy űrlap, amelyet a HTML kódon belül egy külön egységként kezelünk.

Látható és rejtett elemek

A form tartalmazhat látható, és nem látható (rejtett) elemeket. Látható elemek például a szövegbeviteli mezők, legördülő listák és a gombok, míg a rejtett elemekben olyan információkat tárolhatunk, amelyekre a form feldolgozásakor szükségünk lehet, de nem szeretnénk, ha az űrlapot kitöltő személy tudomására jutna. Például a saját email címünket, amelyre a kitöltött űrlapot el fogjuk küldeni, elrejthetjük egy ilyen rejtett mezőbe.

Formok feldolgozási folyamata

Miután az űrlapot kitöltöttük, elküldjük egy feldolgozó programnak. Ez a feldolgozó program egyesével beolvassa az űrlap elemeit és értékeiket, majd elvégzi rajtuk az általunk megadott utasításokat. Például eltárolhatjuk az adatokat egy adatbázisban, bármilyen formában kiírathatjuk az adatokat a képernyőre, vagy akár el is küldhetjük az adatokat egy email címre.

A form felépítése

<FORM> : A form kezdete, mindig le kell zárni! Amit ezen belül elhelyezünk, az mind a form része lesz. Ez lehet szöveg, kép, link, és lehetnek standard form-elemek is, melyeket az alábbiakban részletesen bemutatunk. A megadható paraméterek a következők:

`name="form1"` : a form neve

`action="feldolgozo.xxx"` : a feldolgozó oldal neve (és elérési útvonala)

`method="POST"` : a form elemeinek átadási módja. (**POST** vagy GET)

Beviteli mezők és típusaik

<INPUT> : Beviteli mező. Nem kell lezárni! A megadható paraméterek:

name="neve" : a beviteli mező neve
type="text" : egyszerű szöveges mező (alaphelyzetben ez van kiválasztva)
"password" : jelszó típusú mező, *-okat látunk a beírt karakterek helyén
"hidden" : rejtett mező, később részletezzük
"button" : egyszerű gomb
"submit" : gomb a form elküldéséhez
"reset" : gomb a form alaphelyzetbe állításához
"checkbox" : kijelölő négyzet (checkbox)
"radio" : rádiógomb
size="30" : a szöveges beviteli mező hossza
value="valami" : a beviteli mező alapértéke (gomboknál a gomb felirata)

Kijelölő négyzetek (checkbox)

```
<INPUT name="auto" type="checkbox" value="van autónk">
<INPUT name="karos" type="checkbox" value="dohányzunk">
<INPUT name="karos" type="checkbox" value="iszunk">
<INPUT name="baratno" type="checkbox" value="van barátnőnk">
```

A négyzetek közül bármelyiket kiválaszthatjuk (akár az összeset is), de üresen is hagyhatjuk őket. Ha több mezőnek ugyanazt a nevet adjuk, a form feldolgozásakor a kiválasztott mezők értékeit vesszővel elválasztva kapjuk vissza. Tehát a mezők értékei a feldolgozás után (ha mindet kiválasztottuk):

auto = van autónk
karos = dohányzunk, iszunk
baratno = van barátnőnk

Ha paraméterként beírjuk, hogy `checked`, akkor az adott mező alaphelyzetben ki lesz választva.

Rádiógombok

```
<INPUT name="auto" type="radio" value="van autónk">
<INPUT name="auto" type="radio" value="nincs autónk" checked>
```

Ha ugyanazt a nevet adjuk nekik, akkor a kis gombok (köröcskék) közül maximum egyet választhatunk ki! A `checked` paraméter megadásakor itt is alaphelyzetben ki lesz választva az adott gomb. Ha több helyre is beírjuk, hogy `checked`, akkor az utolsó lesz az érvényes!

Legördülő listák

<SELECT> : Legördülő lista kezdete. Mindig le kell zárni! A megadható paraméterek:

name="neve" : a lista neve

multiple : több sor is kijelölhető egyszerre (Ctrl + klikk)

size : hány sor legyen látható egyszerre (alapérték: 1)

<OPTION> : A lista egy sora. Nem kell lezárni! A tag után kell írni azt a szöveget, amit a listában látni szeretnénk. A megadható paraméterek:

value="ertek" : ha ezt a sort választjuk ki, ez lesz a legördülő lista értéke.

selected : az adott sor lesz alaphelyzetben kiválasztva.

Egy példa a legördülő listára:

```
<SELECT name="eletkor" size=1>
    <OPTION value="15"> 15 éves vagyok
    <OPTION value="20"> 20 éves vagyok
    <OPTION value="25" selected> 25 éves vagyok
    <OPTION value="30"> 30 éves vagyok
</SELECT>
```

Szöveges területek

```
<TEXTAREA name="szoveges"> alapszöveg </TEXTAREA>
```

Vigyázzunk arra, hogy a nyitó és záró rész között a szóközöket és a sortöréseket is komolyan veszi a böngésző! Az alapértéket ide kell beírnunk, és nem a **value** paraméterbe, mint a fenti esetekben!

JavaScript alapok

JavaScript: objektumokra és értékeikre való hivatkozás

Egy HTML oldalon belül az oldal elemeire a következőképp hivatkozhatunk:

1.szint: `document`
2.szint: a form neve, pl. `form1`
3.szint: az elem neve a formon belül, pl. `eletkor`
4.szint: az elem értékére való hivatkozás: `value`

A szinteket egymástól egy pont választja el:

```
document.form1.eletkor.value
```

Vegyük a következő példát:

```
<form name="pelda">
     <b>Név:</b>
     <input name="nev" size=20 value="Nevem">
     <br>
     <b>Életkor:</b>
     <select name="eletkor">
          <option value="10"> 10
          <option value="20" selected> 20
          <option value="30"> 30
     </select>
     <br>
     <b>Dohányzik?</b>
     <input name="dohany" type="radio" value="nem" checked>
     <input name="dohany" type="radio" value="igen">
     <br><br>
     <input type="button" value="Nyomd meg!">
</form>
```

Ebben az esetben az értékek alakulása a következő:

`document.pelda.nev.value` : Nevem
`document.pelda.eletkor.value` : 20

Az onClick() esemény bemutatása, értékadás

Az előző példában lévő gombot módosítsuk a következőképp:

```
<input type="button" value="Nyomd meg!"
onClick="document.pelda.nev.value='Géza'">
```

Ekkor gombnyomásra a név mezőbe bekerül a „Géza” szöveg.

JavaScript függvények szintaktikája, elhelyezkedése a HTML kódon belül

A fenti kóddal egyező eredményt kapunk, ha egy JavaScript függvényt használunk:

```
<input type="button" value="Nyomd meg!" onClick="ujnev()">
```

A HTML kód HEAD részébe a következő JavaScript kódot kell beilleszteni:

```
<SCRIPT language="JavaScript">
function ujnev() {
  document.pelda.nev.value='Géza';
}
</SCRIPT>
```

Változók használata JavaScript-ben

A változó az tulajdonképpen egy tároló, aminek neve van. A változóban bármit tárolhatunk: számokat, szövegeket, objektumokat, stb...

Két beviteli mező értékének felcserélésekor például szükséges egy változót is használnunk:

```
function csere() {
  valtozo = document.form1.egyik.value;
  document.form1.egyik.value = document.form1.masik.value;
  document.form1.masik.value = valtozo;
}
```

Függvények paraméterezése

A függvény neve utáni zárójelbe változók nevét írhatjuk. A változóknak a függvény meghívásakor adhatunk értéket:

Függvény meghívása egy paraméterrel:

```
<input type="button" value="Nyomd meg!"
onClick="ujnev('Ági')">
```

A módosított függvény:

```
function ujnev(valtozo) {
  document.pelda.nev.value=valtozo;
}
```

Üzenetablakok megjelenítése

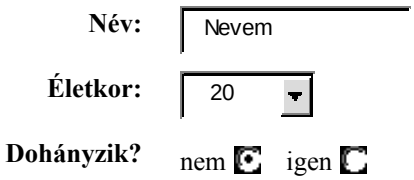
Üzenetek kiírására a beépített **`alert()`** függvényt használhatjuk. A függvény egyetlen paramétereként a kiíratni kívánt szöveget kell megadnunk.

Módosítsuk a fenti függvényt a következőképp:

```
function ujnev(valtozo) {
  document.pelda.nev.value=valtozo;
  alert('Név megváltoztatva!');
}
```

Formok táblázatba rendezése

A táblázatok egyik haszna, hogy segítségükkel a form elemeit szépen egymás alá igazíthatjuk, illetve a mezők címkéit és a mezőket elválaszthatjuk egymástól:



Feltételes elágazás a JavaScript kódban

A feltételes elágazás szerkezete a JavaScript-ben:

```
if (feltétel) {
  ha a feltétel teljesül, ezt hajtja végre;
}
else if (második feltétel) {
  ha a második feltétel teljesül, ezt hajtja végre;
}
else {
  ha egyik feltétel sem teljesül...;
}
```

Checkbox és Radio gombok kijelölésének vizsgálata:

`document.form1.gombok[x].checked`

A "*form1"*-en található *"gombok"* nevű elem (*x+1)*-edik gombja ki van-e választva.

A számozás 0-tól indul! A kifejezés értéke `true` ha igaz, és `false` ha hamis.

Pl: ha a 3. gomb be van jelölve, akkor a `document.form1.gombok[2].checked` kifejezés értéke igaz, tehát `true`.

Egy összetett onClick() esemény bemutatása

Próbáljuk ki, hogy mi történik, ha a függvényt így módosítjuk:

```
function ujnev() {
 document.pelda.nev.value='Géza';
 document.pelda.eletkor.value=30;
 if (document.pelda.dohany[1].checked) {
  document.pelda.nev.value=document.pelda.nev.value+' [igen]';
 }
 else {
  document.pelda.nev.value=document.pelda.nev.value+' [nem]';
 }
 alert('Kész vagyok!');
}
```

Apache – PHP - MySQL

!! FIGYELEM !! !! FIGYELEM !! !! FIGYELEM !!

Az alábbi leírásban az aktuális meghajtó betűjele "C:". Ha az általunk használt operációs rendszer másik partíción helyezkedik el, ne feledjük el a betűjelet ennek értelmében módosítani (pl. D: vagy E: stb.)

Ugyancsak módosítanunk kell az alapesetben használt "Windows" könyvtárat, ha Windows NT vagy Windows2000 operációs rendszert használunk! Ebben az esetben a "Windows" könyvtár nevét mindenhol "WINNT"-re kell árírnunk!!!

!! FIGYELEM !! !! FIGYELEM !! !! FIGYELEM !!

Apache, PHP és MySQL: a három jóbarát

Aki PHP-ben programozik, annak érdemes az Apache WEB-szervert, valamint a MySQL adatbáziskezelő rendszert használnia még akkor is, ha Windows operációs rendszer alatt dolgozik. Az Apache alternatívája lehetne a **Microsoft IIS**, a MySQL-t pedig helyettesíthetné a **Microsoft Access** adatbáziskezelő, viszont ha Linux-os környezet alatt kell folytatni a munkát, akkor ezeket a Windows-os programokat és környezeteket már nem áll módunkban használni!

- Az **Apache** egy virtuális WEB-szerver, ami lehetővé teszi számunkra a PHP programok saját gépünkön való futtatását.
- A **PHP** egy szerveroldali szkript-nyelv, melyel egyszerűen és hatékonyan tudunk aktív WEB-oldalakat készíteni.
- A **MySQL** egy egyszerű adatbázis-kezelő környezet, mely az SQL lekérdező nyelv segítségével tud kommunikálni a PHP oldalakkal.

Mindhárom területet részletesen megvizsgáljuk majd, előtte azonban meg kell teremtenünk a munkához szükséges feltételeket. Az Apache, PHP és MySQL sikeres telepítése és beállítása érdekében az alábbi műveleteket kell végrehajtanunk, szigorúan az itt megadott sorrendben!

Normál esetben a teljes telepítés időtartama nem több 5 percnél!

Az Apache 2 WEB-szerver telepítése, indítása és tesztelése

1. Telepítsük az Apache 2-t:

a) Indítsuk el a telepítőt: **apache_2.0.44-win32-x86-no_ssl.exe**

b) Ha megjelenik egy üzenet a "Windows Installer"-ről szóló szöveggel, kattintsunk az "**OK**" gombra

c) Az üdvözlő oldalon ("Welcome...") kattintsunk a "**Next**" gombra

d) Jelöljük be a rádiógombot az "**I accept...**" szöveg mellett, és kattintsunk a "**Next**" gombra

e) Kattintsunk a "**Next**" gombra

f) A beviteli mezőkbe fentről lefele sorrendben a következőket írjuk:

localhost

localhost

me@localhost.com

g) Ellenőrizzük, hogy a "**for All Users, on Port 80**" melletti rádiógomb be van-e jelölve, majd kattintsunk a "**Next**" gombra

h) Ellenőrizzük, hogy a "**Typical**" melletti rádiógomb be van-e jelölve, majd kattintsunk a "**Next**" gombra

i) A felkínált útvonalat fogadjuk el a "**Next**" gombra kattintva

j) Kattintsunk az "**Install**" gombra, és várjunk amíg a telepítés végbemegy

k) Végül kattintsunk a "**Finish**" gombra

2. Indítsuk el az Apache szervert:

Elvileg telepítés után az Apache szerver automatikusan elindul. Ha jobb oldalt alul látunk egy ikont, melyben egy jobbra mutató kis **zöld háromszög** van, akkor a szerver fut. Ha ugyanitt kis **piros négyzetet** látunk, a szerver áll.

A kis ikonra egyet kattintva, majd az Apache 2 menüpontot (több sor nincs is) választva elindíthatjuk (**Start**), leállíthatjuk (**Stop**) vagy újraindíthatjuk (**Restart**) a szervert.

3. Teszteljük le a szerver működését:

Ha a szerver fut, a böngészőbe (IE vagy Netscape) írjuk be a következő címet:

http://localhost

Ha az "**Enter**" gomb lenyomása után megjelenik egy WEB-oldal, rajta vastag betűkkel a következő szöveg: "**Seeing this instead of the website you expected?**", akkor sikeresen telepítettük az Apache 2 WEB-szervert!

A PHP4 telepítése és beállításai

1. Telepítsük a PHP4-et:
   a) Indítsuk el a telepítőt: **php-4.2.3-installer.exe**
   b) Kattintsunk a "**Next**" gombra
   c) Kattintsunk az "**I agree**" gombra
   d) Ellenőrizzük, hogy a "**Standard**" melletti rádiógomb be van-e jelölve, majd kattintsunk a "**Next**" gombra
   e) A felkínált útvonalat a "**Next**" gombra kattintva fogadhatjuk el. Ha nem a C meghajtóra telepítjük a PHP-t, akkor kattintsunk a "**Browse**" gombra, írjuk be a helyes útvonalat (pl. E:\PHP) és kattintsunk az "**OK**" gombra. Ha az útvonal helyes, kattintsunk a "**Next**" gombra
   f) Ellenőrizzük, hogy a beviteli mezőkben a következő adatok szerepelnek-e:

      localhost

      me@localhost.com

      Ha nem, módosítsuk őket ennek megfelelően.
   g) Kattintsunk a "**Next**" gombra
   h) Jelöljük be a rádiógombot az "**Apache**" szöveg mellett, és kattintsunk a "**Next**" gombra
   i) Kattintsunk a "**Next**" gombra
   j) Ha megjelenik egy "*Sorry, the software...*" kezdetű üzenet, kattintsunk az "**OK**" gombra
   k) Végezetül kattintsunk az "**OK**" gombra

2. Végezzük el a szükséges beállításokat:
   a) Hozzunk létre egy új könyvtárat: **C:\phpweb**
   b) Nyissuk meg szerkesztésre a **C:\Windows\php.ini** fájlt
   c) Keressük meg és módosítsuk a következő sorokat:

      doc_root = "C:\phpWeb"
      extension_dir = "C:\php"

      register_globals = On
   d) Mozgassuk át a **C:/PHP/php4ts.dll** fájlt a **C:\Windows** könyvtárba

Az Apache szerver felkészítése PHP fájlok futtatására

1. Módosítsuk az Apache konfigurációját:
   a) Nyissuk meg szerkesztésre a következő fájlt:

      C:/Program Files/Apache Group/Apache/Conf/httpd.conf

   b) Keressük meg és módosítsuk a következő sorokat:

      DocumentRoot "C:/phpweb"

      <Directory "C:/phpweb">

   c) 3. Keressük meg a "AddType image/x-icon .ico" sort, és alá vegyük fel a következő sorokat:

      ScriptAlias /php4/ "C:/PHP/"
      AddType application/x-httpd-php .php
      AddType application/x-httpd-php .php3
      AddType application/x-httpd-php .php4
      Action application/x-httpd-php "/php4/php.exe"

FIGYELEM!!!

Mint az a fenti sorokból egyértelműen kitűnik, az Apache konfigurációs fájl-jában a megszokottól eltérően \ (backslash) helyett / (slash) jelet kell használnunk!

2. Teszteljük le az Apache – PHP párost:
   a) Állítsuk le, majd indítsuk újra az Apache szervert
   b) Másoljuk be a mellékelt **test.php** fájlt a **C:\phpweb** könyvtárba
   c) A böngészőbe írjuk be a következő címet:

      http://localhost/test.php

Sikeres telepítés és beállítás esetén egy nyugtázó üzenetet látunk, alatta az aktuális dátummal!

A MySQL telepítése és indítása, és a végső tesztelés

1. Telepítsük a MySQL-t:
   a) Indítsuk el a telepítőt: **SETUP.EXE**
   b) Kattintsunk a "**Next**" gombra
   c) Kattintsunk a "**Next**" gombra
   d) A felkínált útvonalat a "**Next**" gombra kattintva fogadhatjuk el. Ha nem a C meghajtóra telepítjük a MySQL-t, akkor kattintsunk a "**Browse**" gombra, írjuk be a helyes útvonalat (pl. E:\MySQL) és kattintsunk az "**OK**", majd a "**Yes**" gombra. Ha az útvonal helyes, kattintsunk a "**Next**" gombra
   e) Ellenőrizzük, hogy a "**Typical**" melletti rádiógomb be van-e jelölve, majd kattintsunk a "**Next**" gombra
   f) A telepítés végeztével kattintsunk a "**Finish**" gombra

2. Indítsuk el a MySQL szolgáltatást:
   a) Keressük meg és futtassuk a **C:\mysql\bin\winmysqladmin.exe** programot
   b) Ha be kell írnunk egy felhasználónevet és jelszót, mindkettőhöz írjuk ezt: "**proba**".
   c) Indulás után jobb oldalt alul az ún. "System tray"-ben megjelenik egy kis "forgalomirányító" lámpa. Ha ez a lámpa **zölden világít**, a MySQL szolgáltatás fut, nincs egyéb teendőnk!
   d) Ha a lámpa **pirosan világít**, kattintsunk rá egyszer, majd a menüből válasszuk ki az operációs rendszerünknek megfelelőt: Windows95/98/Me esetén a "**Win 9x**"-et, míg Windows NT/2000/XP esetén a "**Win NT**"-t. Az újonnan felbukkanó menüsorból most válasszuk az "**Install the Service**" menüpontot, majd kattintsunk a "**Yes**" gombra. Ezután ismét kattintsunk a kis lámpa ikonra, és a megfelelő operációs rendszer kiválasztása után a felbukkanó menüsorból most válasszuk a "**Start the Service**" menüpontot, és kattintsunk a "**Yes**" gombra. Ezután - ha mindent jól csináltunk - a kis lámpa már **zölden világít**!

3. Az Apache - PHP - MySQL együttes végső tesztelése:
   a) Másoljuk be a mellékelt **testmysql.php** fájlt a **c:\phpweb** könyvtárba
   b) A böngészőbe írjuk be a következő címet:

      http://localhost/testmysql.php

Ha mindent jól csináltunk, megjelenik a teszt-oldal. Az **F5** billentyű minden egyes lenyomása után egy új sornak kell megjelennie. Ha ez így történik, akkor büszkék lehetünk: sikeresen telepítettük az Apache – PHP – MySQL hármast!!!

PHP alapok

Mi az a PHP?

A PHP kezdetben csak egy makrógyűjteménynek indult, ami a személyes honlapok programozását volt hivatott támogatni. Neve is innen ered: **P**ersonal **H**ome **P**age (más források szerint **P**ersonal **H**omepage **P**rogramming). A PHP azonban túlnőtt kezdeti feladatán, és mára már egy komplett WEB-programozói nyelvvé alakult.

A PHP program szerkezete

Egy PHP program nem más, mint szokványos HTML kód, melybe PHP nyelvű kódrészleteket (szkripteket) illeszthetünk be.

A HTML kódon belül a PHP kód a **<?** és a **?>** jelek közé kerül. Többsoros kód esetében a sorok végére pontosvesszőt kell tennünk. Íme egy egyszerű példa:

```
<HTML>
  <BODY>
    <b>Egy egyszerű összeadási művelet:</b><br>
    <?
    $eredmeny = 10 + 13;
    print "Tíz plusz tizenhárom = $eredmeny";
    ?>
    <br><br>
    <b>Ez pedig a pontos idő:</b> <?= date("H:i:s")?>
  </BODY>
</HTML>
```

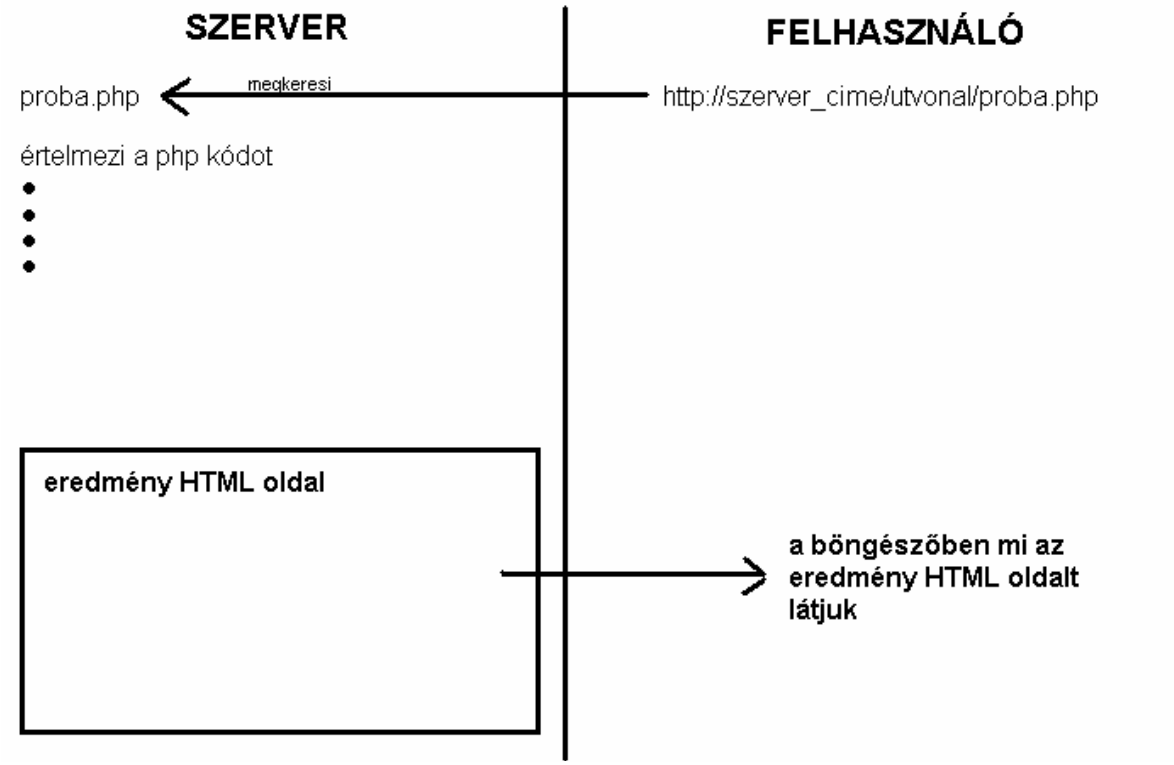
Szerveroldali szkriptek és működésük

A PHP tulajdonképpen egy szerveroldali szkript nyelv. Ez a megfogalmazás elsőre bonyolultnak tűnhet, ám roppant egyszerű megérteni (és ha ezt megértjük, akkor a PHP lényegét már el is sajátítottuk).

A szerveroldali szkriptek a következőképp működnek:

1. A felhasználó beírja a böngészőbe a futtatni kívánt PHP program nevét (elérési útvonallal együtt).
2. A szerver, ahol a PHP program található, megkeresi a szóban forgó PHP fájlt, és egy értelmező program segítségével lefuttatja azt.
3. Futtatás közben egy úgynevezett "eredmény HTML oldalt" készít, melyet - miután a PHP kód végére ért - visszaküld a felhasználói oldalra a böngészőnek.
4. A felhasználó már csak ezt a PHP kód alapján készült HTML oldalt látja.

A folyamatot az alábbi ábra szemlélteti:



Az eredmény HTML oldal elkészítésének szabályai

A PHP-értelmező program az alábbi szabályokat veszi figyelembe a PHP kód értelmezése közben:

1. Ha **HTML kódot** talál, azt változtatás nélkül **átmásolja** az eredmény oldalra.

2. Ha **PHP kódot** lát, azt **lefuttatja**. Ha azt a parancsot kapja, hogy "Írd ki!", akkor a kiírandó szöveget az **eredményoldalra írja ki**, mégpedig az aktuális pozícióba. Ha bármilyen más parancsot kap, azt végrehajtja, és nem nyúl az eredmény oldalhoz.

A fentiekben ismertetett PHP kód eredmény oldala a következőképp nézne ki:

```
<HTML>
  <BODY>
    <b>Egy egyszerű összeadási művelet:</b><br>
    Tíz plusz tizenhárom = 13
    <br><br>
    <b>Ez pedig a pontos idő:</b> 19:35:06
  </BODY>
</HTML>
```

Kiíró parancs a PHP-ben

Mint azt a fenti példából is látjuk, a PHP-ben kétféleképpen adhatjuk ki az "Írd ki!" parancsot. Az első lehetőség a **PRINT()** függvény használata, melyet a következőképp használhatunk:

```
print("szöveg, amit ki szeretnénk írni");
```

A print függvény esetében a zárójeleket nyugodtan elhagyhatjuk:

```
print "szöveg, amit ki szeretnénk írni";
```

A kiíratásnak létezik egy egyszerűbb módja is, melyet kizárólag akkor alkalmazhatunk, ha a nyitó és záró PHP jelek között csak ezt az egy kiíró parancsot szeretnénk kiadni. Ekkor a **nyitó jel után közvetlenül egy egyenlőségjelet** kell kitennünk, majd ezt követi a kiírni kívánt szöveg:

```
<?= "szöveg, amit ki szeretnénk írni"?>
```

Első ránézésre szokatlan lehet, de PHP-vel nemcsak látható szöveget írhatunk ki az eredmény oldalra, de HTML kódot is. A következő két kódrészlet eredménye ugyanaz:

```
<HTML>
  <BODY>
    <?= "valami"?>
  </BODY>
</HTML>
```

```
<HTML>
  <?
  print "<BODY>";
  print "valami";
  print "</BODY>";
  ?>
</HTML>
```

Változók használata

A PHP-ben a változók neve elé egy dollárjelet kell tennünk. A változók létrehozásának legegyszerűbb módja az értékadás:

```
$szam = 15;
$szoveg = "valami";
```

A változókat szövegeken belül is elhelyezhetjük. A szöveg kiírásakor a változó értéke a megadott helyen kerül kiírásra:

```
<?
$eletkor = 23;
print "Én most $eletkor éves vagyok."
?>
```

A fenti kód eredménye: "Én most 23 éves vagyok."

Szövegrészek és változók összefűzése

A PHP-ben a szövegösszefűzés jele a **pont**. A következő szöveget 3 részletben írjuk ki:

```
<?
print "Én most" . " 23 éves" . " vagyok."
?>
```

Szövegrészeket és különböző típusú változókat nyugodtan összefűzhetünk, az eredmény mindig szöveges típusú lesz:

```
<?
$eletkor = 23;
$szoveg = "éves";
print "Én most" . $eletkor . $szoveg . " vagyok."
?>
```

A fenti kódrészlet helyett a következőt is írhattuk volna:

```
<?
$eletkor = 23;
$szoveg = "éves";
print "Én most $eletkor $szoveg vagyok."
?>
```

Dátum- és időfüggvények használata

A dátum és időpont lekérdezésére leggyakrabban a DATE() függvényt használjuk:

```
<?= date("Y-m-d, H:i:s")?>
```

A vezérlő karakterek jelentése a következő: **Y**: év, **m**: hó, **d**: nap, **H**: óra, **i**: perc, **s**: mp.

Ezeket a betűket bármilyen sorrendben felhasználhatjuk, közéjük tetszőleges karaktereket tehetünk. A vezérlő karakterek helyére az aktuális érték kerül, a többi karakter érintetlen marad. Például kiírhatjuk a hónapot és az évet egy per-jellel elválasztva:

```
<?= date("m/Y")?>
```

Feltételes elágazás a PHP-ben

A PHP-ben a feltételes elágazás szerkezete leginkább a JavaScript-hez hasonlít:

```
if (feltétel) {
  [parancsok]
}
elseif (újabb feltétel) {
  [parancsok]
}
else {
  [parancsok]
}
```

Függvények szerkezete, meghívása és paraméterezése

A függvények létrehozása a PHP-ben szintén a JavaScript-hez hasonlóan történik:

```
function osszeadas ($szam1,$szam2) {
  $eredmeny = $szam1 + $szam2;
  print "$szam1 + $szam2 = $eredmeny";
}
```

A függvény meghívása:

```
osszeadas(15,30);
```

Az eredményoldalon ez kerül kiírásra: "45".

A fenti példában meghívtuk a függvényt, átadtunk két paramétert, majd a függvényen belül kiadtunk egy kiíró parancsot. Másik módja a függvények használatának, hogy a meghívott függvény egy értéket ad vissza, amit utána felhasználunk (például kiíratunk):

```
function osszead($egy,$ket) {
  $eredmeny = $egy + $ket;
  return $eredmeny;
}
```

A függvény meghívása:

```
print osszead(15,30);
```

Az eredményoldalon ismét ez kerül kiírásra: "45".

Űrlapok feldolgozása PHP-vel

Mezők az űrlapon

Az űrlapon minden mezőnek, amit el szeretnénk küldeni feldolgozásra, nevet kell adnunk. Általában bármilyen ékezet nélküli nevet használhatunk, erre nézve nincs különösebb szabály. A mező neve legyen rövid, és utaljon a mező tartalmára (pl. nev, jelszo, eletkor, stb.).

Gomboknak nem kell nevet adnunk, hiszen azokat nem küldjük el feldolgozásra.

Rejtett mezőknek nemcsak nevet, de értéket is kell adnunk, mivel a felhasználó ezt nem tudja megtenni.

Példaként vegyük a következő űrlapot, ahol minden mezőnek alapértéket is adunk:

```
<FORM action="feldolgozo.php" method="POST">
  <input type="hidden" name="rejtett" value="valami">
  Név: <input type="text" name="nev" value="Eza Nevem">
  <br>
  Jelszó: <input type="password" name="jelszo" value="123">
  <br><br>
  <input type="submit" value="Elküld!">
</FORM>
```

Mezők felhasználása változóként

A PHP feldolgozó-program futtatásakor az űrlapról **átadott mezőket** automatikusan **változókként** tudjuk kezelni: a változó neve a mező neve, értéke pedig a mező értéke lesz. Természetesen a változó neve előtt a kötelező dollárjelet is használnunk kell.

Ha a fenti űrlapot változtatás nélkül elküldjük feldolgozásra, a "**feldolgozo.php**" oldalon az átadott értékeket a következő kóddal tudjuk kiíratni az eredmény oldalra:

```
<?
print "<b>Név:</b> $nev <br>";
print "<b>Jelszó:</b> $jelszo <br>";
print "</b>Rejtett:</b> $rejtett";
?>
```

Ekkor az eredményoldalon a következőt olvashatjuk:

Név: Eza Nevem
Jelszó: 123
Rejtett: valami

Paraméterek átadása manuálisan egy URL-ben

Ha egy űrlapot küldünk el feldolgozásra, akkor paraméterként az űrlapon szereplő mezők nevét és értéküket adjuk át a feldolgozó PHP programnak.

Van viszont egy másik lehetőség is a paraméterek és értékeik átadására: egy URL-ben (például egy linkre kattintva) a feldolgozó program neve után egy kérdőjellel jelezzük, hogy paramétereket kívánunk átadni, majd ezután felsoroljuk a paraméterek nevét és értékét. A paraméter neve után egy egyenlőségjelet követően írjuk be a paraméter értékét. A paramétereket egymástól egy & jellel választjuk el.

Ha tehát a következő URL-t hívjuk meg (például egy link segítségével), akkor a feldolgozó program szemszögéből ugyanaz történik, mintha a fentebb kidolgozott űrlapot küldenénk el neki:

```
feldolgozo.php?nev=Eza Nevem&jelszo=123&rejtett=valami
```

A "**feldolgozo.php**" oldalon ekkor ugyanúgy három változót tudunk felhasználni (és kiíratni), névszerint: **$nev**, **$jelszo** és **$rejtett**.

Egy név alatt több érték átadása: tömb típusú mezők

Ha checkbox-ok egy csoportjának ugyanazt a nevet adjuk, és az összes négyzetet kipipáljuk, akkor egy név alatt több értéket fogunk elküldeni a feldolgozó programnak.

Hasonló a helyzet a többszörös kiválasztást engedélyező listák esetében is: a listának egy neve van, de több értéket is át szeretnénk adni (a kiválasztott sorok értékeit).

A feldolgozó PHP oldalon ezeket a mezőket nem változóként, hanem tömbként tudjuk majd felhasználni. Ehhez viszont az szükséges, hogy a mezők nevének végéhez a tömbre utaló **[ ]** jeleket is hozzá kell fűznünk!

Az alábbi példában három azonos nevű checkbox-ot készítünk, majd mindhármat bejelöljük, és így küldjük el az űrlapot a feldolgozó programnak:

```
<FORM action="feldolgozo.php" method="POST">
  Audi <input type="checkbox" name="auto[]" value="Audi">
  Lada <input type="checkbox" name="auto[]" value="Lada">
  BMW <input type="checkbox" name="auto[]" value="BMW">
  <br><br>
  <input type="submit" value="Elküld!">
</FORM>
```

Elküldés után a "**feldolgozo.php**" oldalon egy **$auto** nevű tömböt kapunk, melyet majd fel kell dolgoznunk (például kiíratjuk az elemeit, vesszővel elválasztva). A PHP oldalon a tömb neve után már nem kell kitennünk a **[ ]** jeleket!

Először azonban azt kell megvizsgálnunk, hogy kaptunk-e egyáltalán ilyen nevű tömböt! Ha ugyanis egyik checkbox-ot sem jelöljük ki, akkor egyáltalán nem kapunk **$auto** nevű tömböt, így azt fel sem tudjuk dolgozni! Ez csak a checkbox-okra és a rádiógombokra igaz, a többi mezőtípusra nem! Ha tehát egy szöveges mezőt üresen küldünk el, akkor ettől még lesz olyan nevű változónk, csak az értéke üres lesz.

Változó és tömb létezésének vizsgálatára a következő függvényt használhatjuk:

`isset($valtozo)` vagy `isset($tomb)`

Az **isset()** függvény visszaadott értéke igaz (**true**), ha létezik ilyen nevű változó vagy tömb, és hamis (**false**) ha nem létezik.

A fenti, checkbox-okat tartalmazó űrlapot feldolgozó PHP oldal kódja tehát így kell, hogy kezdődjön:

```
if (isset($auto)) {
  ...tömb feldolgozása...
}
```

Ha létezik **`$auto`** nevű tömb, akkor elkezdhetjük annak feldolgozását.

Tömbök feldolgozása

Egy tömb elemein az alábbi ciklus segítségével futhatunk végig:

```
foreach ($tomb as $elem) {
  ...műveletek...
}
```

A ciklus minden egyes lépésében a következő tömbelemet beolvassuk az **`$elem`** nevű változóba, amivel már azt teszünk, amit szeretnénk. Ha elfogytak a tömbelemek, a ciklusnak vége, és a program futása a ciklus után folytatódik.

Ha tehát az űrlapon kiválasztott autók neveit ki szeretnénk iratni egymás alá, a következő feldolgozó programot kell elkészítenünk:

```
if (isset($auto)) {
  foreach ($auto as $autonev) {
    print $autonev . "<br>";
  }
}
```

A fenti kódban először megvizsgáljuk, hogy egyáltalán választottunk-e ki autót, majd a tömbön végiglépkedve kiírjuk a tömbelemek értékét, melyhez mindig hozzáfűzünk egy sortörést is.

Adatbázis alapok, MySQL

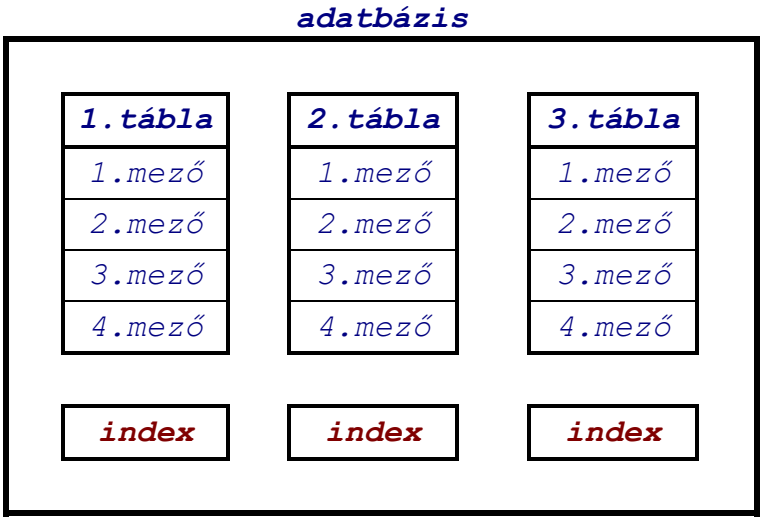
Mező, rekord és adattábla fogalma

Az alábbi táblázat egy adattábla, melynek 4 mezője van, és fel van töltve 3 rekorddal.
A mezők neve a fejléc sorban látható. Egy mező egy oszlopnak, egy rekord a táblázat egy sorának felel meg:

	1.mező	2.mező	3.mező	4.mező
fejléc	Név	Telefon	Életkor	Szül. idő
1. rekord	Kiss Miska	125-3654	56	1985.04.06
2. rekord	Balogh Heni	589-6589	14	1919.10.25
3. rekord	Magyari Miki	256-6589	19	2002.12.31

Adatbázis és az adattáblák kapcsolata

Az adatbázis nem más, mint olyan adattáblák együttese, amelyek egy alkalmazáshoz tartoznak. Az adattáblákon kívül az adatbázis tartalmazza az indexfájlokat is, illetve egyéb kiegészítő információkat, melyek a felhasználó számára „láthatatlanok”.



Indexek és az egyedi azonosítók fontossága

Minden adattáblában szükség van egy olyan mezőre, aminek a tartalma minden rekord esetében más lesz, tehát egyértelműen azonosítja a rekordokat. Célszerű ezt a mezőt először létrehozni, és "**id**"-nak elnevezni.

Például a következő táblában az "**id**" mező nélkül a 2. és a 4. rekordot nem lehetne megkülönböztetni:

id	nev	eletkor
1	Miki	24
2	Zoli	15
3	Laci	33
4	Zoli	15

Az index-fájl nem más, mint egy adott mező szerint sorbarendezett tábla, mely az eredeti tábla minden rekordját képviseli. Az index-fájlok segítségével egy adott mező szerinti keresést sokkal gyorsabban hajt végre a program, mint index-fájl nélkül. Az index-fájl két mezőt tartalmaz: egyrészt a kiválasztott mezőt, ami szerint sorba van rendezve, másrészt egy azonosító mezőt, amely megadja, hogy az adott rekord az eredeti táblában **fizikailag hányadik** helyen található (ez utóbbi nem mindig egyezik az id mező értékével!!!).

Példaként vegyünk egy táblát, majd a hozzá tartozó, a "**nev**" mező szerinti index-táblát:

id	nev	eletkor
1	Zolika	6
3	Petike	8
5	Andráska	12
7	Sanyika	7

#	nev
3	Andráska
2	Petike
4	Sanyika
1	Zolika

A MySQL Control Center telepítése, indítása

Az adatbázisok kezelésére mi a MySQL rendszert választottuk, melyet egy grafikus felületen keresztül fogunk kezelni. Ez a grafikus felület a MySQL Control Center, melynek telepítő programját a következő helyről indíthatjuk el:

`X:\4. Adatbazis alapok, MySQL\MySQL CC setup\Setup.exe`

A telepítési folyamat egyszerű: a telepítő indítása után kattintsunk a "**Next**" gombra, válasszuk ki az "**I accept the license agreement**" rádiógombot és kattintsunk a "**Next**" gombra, majd még négyszer ugyancsak a "**Next**" gombra kell kattintanunk, és végül a "**Finish**"-re.

A MySQL Control Center-t a Start-menüből indíthatjuk: "**Start**" - "**Programs**" - "**MySQL Control Center**" - "**MySQL Control Center**".

Adatbázis-szerver létrehozása és megnyitása

Ha a "**MySQL servers**" ablakban még nem látunk egy szervert sem, létre kell hoznunk egyet. Nyomjuk le a **Ctrl+N** billentyűket, a "**Name**" mezőbe írjuk be a szerver nevét (pl. "proba"), a "**Host Name**" mezőbe pedig azt, hogy "**localhost**". Az "**User Name**" mezőt nem kötelező kitöltenünk, de mégis inkább írjuk be, hogy "**root**". Végül jobb oldalt alul kattintsunk az "**Add**" gombra.

Az újonnan létrehozott szerverünkre duplán kattintva megnyithatjuk azt.

Adatbázis és adattábla létrehozása, indexelés

Az adatbázis-szerver megnyitása után kattintsunk duplán a "**Databases**" mappára, mire láthatóvá válnak a már létrehozott adatbázisok.

A "**Databases**" mappára a jobb egérgombbal kattintva az előugró menüből válasszuk a "**New Database**" opciót, adjuk meg a létrehozandó adatbázisunk nevét, és kattintsunk az "**OK**" gombra.

Az újonnan létrehozott adatbázison duplán kattintva megnyithatjuk azt. Ekkor megjelenik a "**Tables**" sor, melyen jobb egérgombbal kattintva, és a "**New Table**" opciót választva elkészíthetjük első adattáblánkat.

Az első oldalon (**Field Properties**) kell megadnunk az adattábla szerkezetét. Először mindig a mezők neveit adjuk meg, majd az adattípusukat és a hosszukat. Az első mező mindig az "**id**" legyen! Típusa "**bigint**", hossza **20** legyen. Alul jelöljük be az "**AUTO_INCREMENT**" négyzetet, és felül kattintsunk a kis **kulcs** ikonra. Ezzel az "**id**" mező automatikusan növekvő számokat fog tartalmazni, és ő lesz az elsődleges kulcsunk (egyedi azonosító).

A többi mezőnél az alábbi adattípusok közül válasszunk:

int: nem túl nagy egész szám
bigint: nagy egész szám
double: tizedes szám
varchar: vegyes szöveges (betűk, számok és írásjelek)
date: dátum
text: nagyon hosszú szöveg
datetime: dátum és időpont egymás után

A következő oldalon (**Indexes**) a táblához tartozó indexeket tudjuk elkészíteni. A zöld plusz (**+**) jelre kattintva készíthetünk új indexet, amibe a bal oldalról tudunk mezőket felvenni. Mezőnként is készíthetünk indexeket, de egy index több mezőt is tartalmazhat. A fontosabb mezők szerint érdemes indexelni, de ezen még menet közben is változtathatunk.

A harmadik oldalon (**Table Properties**) a "**Table Name**" mezőbe írjuk be adattáblánk nevét, majd felül a mentés ikonra kattintva, vagy a **Ctrl+S** billentyűket lenyomva tudjuk elmenteni a kész táblát. A sikeres menésről alul kapunk egy értesítést (**Table created successfully**), ezután bezárhatjuk az ablakot.

Rekordok felvétele, módosítása és törlése

Az újonnan létrehozott adattábla nevére duplán kattintva megnyithatjuk azt. Kezdetben még üres, hiszen nincsenek rekordok felvéve.
A **Ctrl+I** billentyűket lenyomva, vagy az új rekord felvétele ikonra kattintva vehetünk fel új rekordot. Az "**id**" mezőt nem kell kitöltenünk, hiszen az automatikusan számozódik. A többi mezőbe az adattípusnak megfelelő adatokat írhatunk. A mezők között a **TAB** billentyű lenyomásával, vagy a megfelelő mezőre való egérkattintással válthatunk. Az utolsó érték beírása után célszerű lenyomni az **ENTER** billentyűt.
Rekordok módosításához a módosítani kívánt mezőn kattintsunk duplán, és írjuk felül a régi értéket az újjal.
Rekord törléséhez kattintsunk jobb egérgombbal bármelyik mezőjén, és válasszuk a "**Delete Record**" opciót. A törlést erősítsük meg a "**Yes**" gombra kattintva.

Röviden az SQL nyelvről

Az SQL egy rövidítés, jelentése "**Structured Query Language**", vagyis struktúrált lekérdező nyelv. Ma már szabványnak tekinthető, mégis van egy kis eltérés az egyes változatok között. Az alapvető parancsok és szintaktikájuk azonban minden esetben megegyeznek.

Az SQL parancsokat mi először a MySQL CC-ben fogjuk gyakorolni, majd megnézzük, hogy PHP kódból hogyan tudjuk őket futtatni. A négy alapvető parancs a leválogatás, a felvétel, a módosítás és a törlés.

SQL parancs: leválogatás (szelektív lekérdezés)

A legegyszerűbb parancs, melynek legegyszerűbb formája a következő:

```
SELECT * FROM tablanev
```

A parancs futtatása után elénk tárul az úgynevezett eredménytábla, melyben az általunk megjeleníteni kívánt rekordokat látjuk.

A **SELECT** kulcsszóval adjuk az adatbáziskezelő tudtára, hogy egy lekérdezést szeretnénk lefuttatni.

A ***** (csillag) azt jelenti, hogy az eredménytáblában a rekordoknak az összes mezőjét látni szeretnénk. Ha csak bizonyos mezőkre vagyunk kíváncsiak, akkor a ***** helyett a mezők neveit kell felsorolnunk, egymástól vesszővel elválasztva. Ha például egy "**emberek**" nevű táblából szeretnénk az emberek nevét, életkorát és lakcímét megjeleníteni, az alábbi parancsot kell kiadnunk:

```
SELECT nev,eletkor,lakcim FROM emberek
```

A mezők nevei (vagy a csillag) után következik a **FROM** kulcsszó, majd ezt követi a tábla neve, amiből a rekordokat le szeretnénk kérdezni.

A lekérdezés eredménytábláját különböző feltételek megadásával szűkíthetjük. A feltételeket a **WHERE** kulcsszó után kell megadnunk. Például listázzuk ki azon rekordok összes mezőjét, ahol az életkor értéke kisebb mint 20:

```
SELECT * FROM emberek WHERE eletkor<20
```

Ha több feltételt szeretnénk megadni, azokat logikai műveletekkel kapcsolhatjuk össze. Felhasználható logikai művelet az **AND** (és), az **OR** (vagy) és a **NOT** (nem). Válogassuk ki például azokat a férfiakat, akik 20 évnél idősebbek, vagy életkoruk nem több mint 10:

```
SELECT * FROM emberek
WHERE neme='ferfi' AND (eletkor>20 OR NOT eletkor>10)
```

Hogy egyértelmű legyen a logikai műveletek sorrendje, a feltételek megadásánál használjunk zárójeleket!

Megadhatjuk azt is, hogy a feltételnek megfelelő rekordokat milyen sorrendben szeretnénk az eredménytáblában látni. Ehhez a parancs végén az **ORDER BY** záradékot kell használnunk, mely után fel kell sorolnunk azokat a mezőket, amelyek szerint a sorbarendezést végre szeretnénk hajtani. Több mező megadása esetén a mezőneveket vesszővel választjuk el egymástól. Először rendezzük az embereket név szerint növekvő sorrendbe:

```
SELECT * FROM emberek ORDER BY nev
```

Majd rendezzük őket név szerint növekvő, azon belül életkor szerint növekvő sorrendbe (ez azt jelenti, hogy ha kettő vagy több embernek ugyanaz a neve, őket életkor szerint fogjuk egymás után rendezni):

```
SELECT * FROM emberek ORDER BY nev,eletkor
```

Ha egy mező szerint nem növekvő, hanem csökkenő sorrendben szeretnénk rendezni, a mező neve után illesszük oda a **DESC** kulcsszót is. Például rendezzünk név szerint csökkenő, azon belül életkor szerint növekvő sorrendbe:

```
SELECT * FROM emberek ORDER BY nev DESC,eletkor
```

Az eddig megismert kulcsszavakat és záradékokat együtt is alkalmazhatjuk, ezáltal jó bonyolult lekérdezéseket hozhatunk létre:

```
SELECT nev,eletkor FROM emberek
WHERE neme='ferfi' AND (eletkor>20 OR NOT eletkor>10)
ORDER BY nev DESC,lakhely,eletkor DESC
```

A fenti parancsot így foglalhatnánk szavakba:

"Válogassuk ki azokat a férfiakat, akik 20 évnél idősebbek, vagy életkoruk nem több mint 10. A feltételnek megfelelő embereket rendezzük név szerint csökkenő sorrendbe. Ha vannak azonos nevű emberek, akkor rendezzük őket lakóhely szerint növekvő sorrendbe. Előfordulhat, hogy több embernek ugyanaz a neve, és a lakóhelyük is megegyezik. Ezeket az embereket rendezzük életkor szerint csökkenő sorrendbe. Az embereknek csak a nevére és az életkorára vagyunk kíváncsiak, tehát az eredménytáblában csak ezeket a mezőket szeretnénk látni."

Ugye nem is olyan nehéz? :)

SQL parancsok végrehajtása PHP-ből

Kapcsolódás a MySQL szerverhez és az adatbázis kiválasztása

Ahhoz, hogy PHP-ből SQL parancsokat tudjunk futtatni, először kapcsolódnunk kell a MySQL szerverhez:

```
$kapcsolat = mysql_connect("localhost","userid","jelszo");
```

A fenti sorban létrehoztuk a kapcsolatot, melyet a **$kapcsolat** változóban tárolunk el. Három paramétert adhatunk meg: a szerver nevét, a felhasználónevet és a hozzá tartozó jelszót. Ha a MySQL szerveren létezik a **@%** nevű felhasználó, akkor ezzel is beléphetünk, ekkor nem kell felhasználónevet és jelszót beírnunk. Ha a felhasználónak nincs jelszava (pl. a **root** nevű felhasználó), akkor a jelszó paraméter elhagyható.

Miután létrehoztuk a kapcsolatot (amit bárhogyan elnevezhetünk, de mi az egyszerűség kedvéért **$kapcsolat**-nak fogjuk hívni), ki kell választanunk az adatbázist, amiben dolgozni szeretnénk:

```
mysql_select_db("adatbazis", $kapcsolat);
```

A fenti sorban megadtuk, hogy az **adatbazis** nevű adatbázist szeretnénk használni, mégpedig a **$kapcsolat** nevű kapcsolaton keresztül.

Az adatbázis-kapcsolat lezárása

Ha már nincs szükségünk az adatbázis-kapcsolatra, le kell zárnunk azt (általában a PHP kód végén, az utolsó sorban):

```
mysql_close($kapcsolat);
```

Fájl beillesztése a PHP kódba

Az előzőekneb tárgyalt 2 sort egy külön fájlban is eltárolhatjuk. A fájl neve legyen például **dbconn.php** (az angol **database connect** rövidítéséből ered), és tartalma legyen a következő:

```
<?
$kapcsolat = mysql_connect("localhost","userid","jelszo");
mysql_select_db("adatbazis", $kapcsolat);
?>
```

Ez akkor lesz hasznos, ha több PHP fájlban is ugyanazt az adatbázis- kapcsolatot szeretnénk használni. Ekkor nem kell minden fájl elejére beírnunk a fenti kódot, elég az úgynevezett include-fájlt beillesztenünk a következő paranccsal:

```
<? include("dbconn.php") ?>
```

A fenti parancssor helyére a PHP értelmező program beilleszti a **dbconn.php** fájl tartalmát, és utána értelmezi a kódot.

SQL parancsok futtatása PHP-ből

Miután elkészült az adatbázis-kapcsolat, és kiválasztottuk a megfelelő adatbázist, a létrehozott kapcsolaton SQL parancsokat futtathatunk.

Először egy szöveges változóban eltároljuk a parancsot:

```
$parancs = "SELECT * FROM tablanev";
```

Majd a parancsot lefuttatjuk a megadott kapcsolaton:

```
mysql_query($parancs);
```

Eredménytábla eltárolása és vizsgálata

A fenti példában egy lekérdezést futtattunk le, viszont a lekérdezés eredményét nem jelenítettük meg. Ahhoz, hogy a lekérdezés eredményét - az úgynevezett eredménytáblát - fel tudjuk dolgozni, először el kell tárolnunk egy tömbben, a következő módon:

```
$eredmeny = mysql_query($parancs);
```

Az eredménytáblát ekkor az **$eredmeny** nevű tömbben tároljuk el.

Ha arra vagyunk kíváncsiak, hogy az eredménytáblának hány sora van, használjuk a következő függvényt:

```
mysql_num_rows($eredmeny)
```

Ha a fenti függvény visszatérési értéke 0 (nulla), akkor az eredménytábla üres, tehát nem találtunk a keresési feltételeknek megfelelő rekordot. Ha tehát a visszatérési érték nagyobb mint nulla, akkor feldolgozhatjuk az eredménytáblát, ellenkező esetben kiírhatunk egy figyelmeztető üzenetet:

```
if (mysql_num_rows($eredmeny)>0) {
  ... feldolgozó parancsok ...
}
else {
  print "Az eredménytábla üres!!!";
}
```

Az eredménytábla feldolgozása sorról sorra

Miután meggyőződtünk róla, hogy az eredménytábla nem üres, sorról sorra feldolgozhatjuk azt. Ehhez egy ciklust fogunk használni, aminek minden fordulójában egy **$sor** nevű tömbbe olvassuk ki az aktuális rekord mezőinek nevét és értékét:

```
while ($sor = mysql_fetch_array($eredmeny)) {
  ... az aktuális rekord feldolgozása ...
}
```

Ha már birtokunkban van egy adott rekord, akkor egyszerűen kiolvashatjuk bármelyik mezőjének értékét, amivel azt csinálunk, amit csak akarunk (például kiíratjuk):

```
while ($sor = mysql_fetch_array($eredmeny)) {
  print $sor["mező1"] . "<br>";
  print $sor["mező2"] . "<br>";
  print $sor["mező3"] . "<br>";
  print $sor["mező4"] . "<br><br>";
}
```

A fenti kódrészlettel végiglépkedünk az eredménytábla sorain, és minden rekordnál kiírjuk az összes mező tartalmát (amennyiben 4 mezője van, amiket **mező1**...**mező4**-nek hívnak). A mezők értékeit mindig új sorba írjuk, a rekordok között egy üres sort hagyunk.

SQL parancs: új rekord felvétele

Új rekordot a következő paranccsal tudunk felvenni egy adattáblába:

```
INSERT INTO táblanév (mező1,mező2) VALUES (érték1,érték2)
```

A parancs formája tehát a következő: az **INSERT INTO** parancsot követi a táblanév, majd zárójelben azoknak a mezőknek a nevei, amiket ki szeretnénk tölteni. Ezután a **VALUES** parancs következik, ami után zárójelben, a megadott mezőnevek sorrendjében fel kell tüntetnünk a mezőkbe írandó értékeket. Ha egy érték típusa szöveges vagy dátum típusú, az értéket aposztrófok közé (**'....'**) kell tennünk!

Vegyünk fel például egy új rekordot az **emberek** táblába, töltsük ki a **nev** és **eletkor** mezőket a következő adatokkal: **Kiss János**, **23**:

```
INSERT INTO emberek (nev,eletkor) VALUES ('Kiss János',23)
```

A fenti SQL parancsot PHP-ből is kiadhatjuk (feltételezve, hogy előzőleg már létrehoztuk a kapcsolatot és kiválasztottuk az adatbázist):

```
$parancs = "INSERT INTO emberek (nev,eletkor) VALUES ('Kiss
János',23)";
mysql_query($parancs);
```

SQL parancs: megadott rekord(ok) törlése

Egy bizonyos feltételnek megfelelő rekordot (rekordokat) a következő paranccsal törölhetünk az adattáblából

```
DELETE FROM táblanév WHERE feltétel
```

FONTOS!!! Ha nem adunk meg feltételt, **az összes rekordot törölni fogjuk** a táblából! A **MySQL Command Center** rá fog ugyan kérdezni, hogy biztosan törölni akarunk-e minden rekordot, viszont ha **PHP**-ből adjuk ki a parancsot, akkor az összes rekordunk menthetetlenül elvész!!!

A **WHERE** kulcsszó után hasonló feltételeket írhatunk, mint a **SELECT** parancs esetében! (Lásd a 4. leckét)

Ha például az emberek táblából törölni szeretnénk mindenkit, aki 20 évnél fiatalabb, akkor a következő parancsot kell futtatnunk:

```
DELETE FROM emberek WHERE eletkor<20
```

A fenti SQL parancsot PHP-ből is kiadhatjuk (feltételezve, hogy előzőleg már létrehoztuk a kapcsolatot és kiválasztottuk az adatbázist):

```
$parancs = "DELETE FROM emberek WHERE eletkor<20";
mysql_query($parancs);
```

SQL parancs: megadott rekord(ok) adatainak módosítása

Egy bizonyos feltételnek megfelelő rekord(ok) bizonyos mezőinek értékét az alábbi paranccsal írhatjuk felül új értékekkel.

```
UPDATE táblanév SET mező1=érték, mező2=érték WHERE feltétel
```

Ha az új érték szöveges vagy dátum típusú, az értéket itt is aposztrófok közé (**'....'**) kell tennünk!

Ha több mező értékét is módosítani szeretnénk, akkor a mezőnév/érték párosokat vesszővel választjuk el egymástól. A parancs formátuma abban különbözik az új rekord felvételétől, hogy míg felvételnél előbb felsoroltuk a mezőket, majd utána az értékeket, addig módosításnál a mezőneveket és értékeket párosával kell megadnunk!

FONTOS!!! Ha nem adunk meg feltételt, **az összes rekordot módosítani fogjuk** a táblában! A **MySQL Command Center** rá fog ugyan kérdezni, hogy biztosan módosítani akarunk-e minden rekordot, viszont ha **PHP**-ből adjuk ki a parancsot, akkor az összes rekordunk visszavonhatatlanul átesik a módosításon!!!

Ha például az **emberek** táblában minden 90 év feletti ember **megjegyzes** mezőjét át szeretnénk írni arra, hogy "aggastyán", akkor a következő parancsot kell kiadnunk:

```
UPDATE emberek SET megjegyzes='aggastyán' WHERE eletkor>90
```

A fenti SQL parancsot PHP-ből is kiadhatjuk (feltételezve, hogy előzőleg már létrehoztuk a kapcsolatot és kiválasztottuk az adatbázist):

```
$parancs = "UPDATE emberek SET megjegyzes='aggastyán' WHERE
eletkor>90";
mysql_query($parancs);
```

A böngésző átirányítása PHP paranccsal

Előfordulhat, hogy egy PHP kód futtatása közben a felhasználó böngészőjét egy új címre szeretnénk irányítani. Ezt a következő paranccsal tehetjük meg:

```
header("Location: http://www.ujoldal.hu");
```

Ha például a jelenleg futó PHP fájllal azonos könyvtárban lévő ujoldal.php fájlt szeretnénk megnyitni, a következő parancsot kell kiadnunk:

```
header("Location: ujoldal.php");
```

FONTOS!!! Igaz, hogy a fenti kóddal böngészőt átirányítottuk, de **a jelenleg futó PHP kód értelmezése folytatódik** egészen az utolsó sorig! Ha a böngészőben egy másik PHP oldalt nyitunk meg, annak az értelmezése egy **új szálon indul el**, és nem befolyásolja a jelenleg futó PHP kód értelmezését! Ha tehát egy ilyen átirányítás után zárjuk le az adatbázis-kapcsolatot, a parancs ugyanúgy le fog futni.

A PHP kód eredmény HTML oldalának bufferelése

Ha a böngészőt egy PHP kódon belül a fenti paranccsal átirányítjuk, a PHP kód legelejére a következő kódot kell írnunk:

```
<? ob_start() ?>
```

FONTOS!!! A fenti kód előtt még egy szóközt sem szabad írnunk, tehát abszolút a kód elejére kell beillesztenünk!

A fenti parancs kiadása után az eredményoldal nem futásidőben kerül át a böngészőhöz, így ha a böngészőt menet közben egy új oldalra irányítjuk, nem kapunk hibaüzenetet.

Az eredményoldalt csak a következő parancs kiadása után küldjk el a böngészőnek:

```
<? ob_end_flush() ?>
```

A fenti parancs jelentése a következő: zárjuk le az eredményoldal bufferelését, és az eddig felgyülemlett HTML kódot küldjük ez a felhasználó böngészőjének.

Ezt a parancsot mindeig a PHP kód legvégére írjuk, az esetleges adatbázis-kapcsolat lezárás után!

PHP és a WAP

Mi az a WML?

Egy WML fájl ugyanolyan szöveges fájl, mint a HTML. Mivel mobiltelefonokra lett szabva, a nyelvezete sokkal egyszerűbb, de több benne a kötöttség.

WML oldalakat nem tudjuk internetes böngészőkkel megjeleníteni, erre kizárólag a mobiltelefonok, vagy WAP-szimulátorok képesek!

Mi a különbség a WAP és a WML között?

A WAP a "**Wireless Application Protocol**" rövidítése, magyarul "**drótnélküli alkalmazás protokoll**"-nak lehetne fordítani. Ez tehát egy rendszer, egy módszer, melynek segítségével mobiltelefonokról is elérhetünk egyszerű WEB-oldalakat. A WML pedig a WAP nyelvezete, tehát a WAP oldalak WML nyelven íródnak.

A WML kód sajátosságai

A WML fájlok elejére a következő sorokat kell beszúrni:

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">
```

Ezután következik a WML "tag", melyet a kód legvégén zárunk be:

```
<WML>
   ...
   ...
</WML>
```

A nyitó és záró "tag" közé több WAP-oldalt (ún. **kártyát**) is beilleszthetünk, melyeket egymásból hívhatunk meg. Alaphelyzetben mindig az első WAP-oldal jelenik meg!

A WAP-oldalakat (kártyákat) a következő TAG azonosítja:

```
<card id="azonosító" title="Oldal címe">
   ...
   ...
</card>
```

PHP oldalak meghívása mobiltelefonról

Mobiltelefonokon is meghívhatunk PHP oldalakat, csakúgy mint az internetes böngészőnkben. Ebben az esetben is **előbb a szerver futtatja le a PHP kódot**, ám itt nem HTML, hanem **WML formátumú** lesz az előállított **eredmény oldal**!

Bekezdések fontossága

Minden kártyának tartalmaznia kell legalább egy bekezdést! Bizonyos mobiltelefonok a szöveget csak akkor jelenítik meg, ha az egy bekezdésen belül helyezkedik el:

```
<card id="nyitolap" title="Üdvözöllek!">
  <p align=”center”>
    Ez egy szoveg…
  </p>
</card>
```

Sortörés: le kell zárni!

A WML nyelv szigorúan számon kéri a tag-ek lezárását! Így olyan tag-eket is **le kell zárnunk**, amelyeket a HTML-ben nem zártunk le. Ha a tag-nek nincsen zárórésze, akkor a nyitó rész végére egy / jelet kell tennünk, így jelezvén a lezárást:

```
<br/>
```

Szövegformázás

A WAP oldalakon nem alkalmazhatjuk a formázó utasítások széles skáláját. Némelyik mobiltelefon abszolút nem képes a szövegformázást értelmezni. A WML nyelvben a következő formázó parancsokat használhatjuk:

** … **: vastag betűk
** … **: dőlt betűk
<u> … </u>: aláhúzott betűk
<small> … </small>: kisbetűk

Linkek

A HTML-hez hasonlóan a WML-ben is alkalmazhatjuk a linkek beillesztését:

```
<a href=”idemutat.wml”> Link szövege </a>
```

Itt viszont nemcsak egy másik WML oldalra hivatkozhatunk, hanem ugyanazon az oldalon belül egy másik kártyára is! Ilyenkor az URL helyett a kártya azonosítóját („**id**”) adjuk meg egy kettőskereszt után:

```
<a href=”#felvetel”> Új ember felvétele </a>
```

Beviteli mezők

A WAP oldalakon nem kell űrlapokat (formokat) létrehoznunk, egyszerűen beillesztjük a kívánt helyre a beviteli mezőket. Figyelem: az **<INPUT>** tag-et is le kell zárnunk, mégpedig önmagán belül! Alapértéknek célszerű egy üres sztringet megadni:

```
<card id="felvetel" title="Felvétel">
  <p>
    Név: <br/>
    <input name=”nev” value=””/> <br/>
    Telefon: <br/>
    <input name=”telefon” value=””/>
  </p>
</card>
```

Beviteli mezők értékeire való hivatkozás linkekben

Mivel a WML-ben nem használunk űrlapokat, a beviteli mezők értékeit más módon kell elküldenünk a feldolgozó programnak. Erre a célra linkeket fogunk használni, ahol a feldolgozó program neve után felsoroljuk a paraméterek neveit és értékeit. A paraméterek között az **&** jelet a WML-ben az **&** kóddal kell helyettesítenünk! A beviteli mezők értékeit a linken belül a **$(mezőnév)** formula segítségével szúrhatjuk be:

```
<a href=”feldolgozo.php?
  mode=Felv&amp;amp;nev=$(nev)&amp;amp;telefon=$(telefon)”>
  Elküldés
</a>
```

Listák (legördülő menük)

A HTML-hez képest csak annyi az eltérés, hogy az **<OPTION>** tag-et itt **le is kell zárnunk**:

```
<select name=”eletkor”>
  <option value=”15”> 15 éves </option>
  <option value=”20”> 20 éves </option>
  <option value=”40”> 40 éves </option>
</select>
```

A listákat minden mobiltelefon más és más formában jeleníti meg, tehát ne csodálkozzunk, ha a megjelenítés eltér a WEB-oldalakon megszokottól!

Funkciógombok (vagy extra menüpontok)

Minden kártyán elhelyezhetünk ún. funkciógombokat. Ezeket a gombokat némelyik mobiltelefon megjeleníti az oldal alján, míg más készülékek a menübe építik bele őket (ezeket hívom én extra menüpontoknak).

A gombok szerkezete a következő:

```
<do type="[típus]" label="[címke]"> [parancs] </do>
```

Minden gombnál megadhatunk egy típust („**type**”) és egy címkét („**label**”), majd a nyitó és záró tag közé elhelyezhetjük a parancsot, amit a gomb kiválasztásakor szeretnénk végrehajtani. Ez a parancs lehet egy URL-re (vagy kártyára) való ugrás, de számos beépített parancs közül is válogathatunk. A gombokat minden esetben a kártya alján, a bekezdés lezárása után kell elhelyeznünk!

Mi most kétféle gombbal ismerkedünk meg:

1. Ha egy olyan gombot szeretnénk elhelyezni az oldalon, amelyet kiválasztva visszalépünk az előző oldalra, a következő kódot alkalmazzuk:

```
<do type="prev" label="Vissza"> <prev/> </do>
```

Ebben az esetben a típus a „**prev**”, a címke a „**Vissza**”, a parancs pedig a **<prev/>**.
(A **<prev/>** tag jelentése: visszalépés az előző oldalra.)

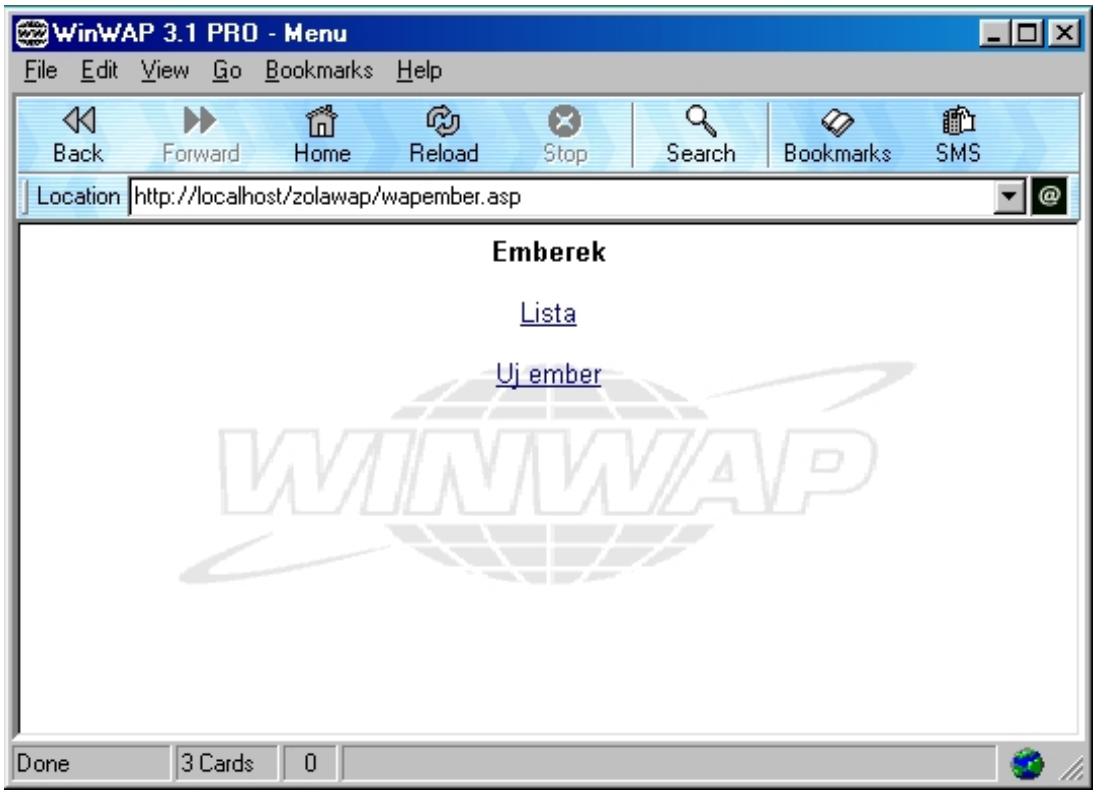
2. Ha a gomb kiválasztásakor egy URL-re (vagy az oldalon belül egy másik kártyára) szeretnénk ugrani, ezt írjuk be a bekezdés lezárása után:

```
<do label="Masik oldal"> <go href="masikoldal.wml"/> </do>
```

Ebben az esetben a típust nem adtuk meg, a címke a „**Másik oldal**”, a parancs pedig a **<go/>**. A <go> tag-nek egy paramétere van, a “**href**”, amely – hasonlóan a linkekhez – azt az URL-t tartalmazza, ahová ugrani szeretnénk. A <go> tag-et is önmagán belül kell lezárni!

Tesztelés a WinWap programmal

Azért, hogy ne kelljen állandóan a mobiltelefonokon tesztelni a készülő WAP oldalakat, célszerű egy windows-os WAP-böngészőt használni! Ilyen program például a WinWap, amelyet telepítve és futtatva egy kis WAP-böngésző ablak tárul elénk. Itt a címsorba WAP oldalak címét kell beírnunk, és az eredmény oldal megjelenik a böngészőben:



A mobiltelefon felkészítése a WAP-olásra

Hogy milyen paramétereket kell beállítanunk a telefonunkon, az mindig az adott készüléktípustól és a mobilszolgáltatótól függ. Érdemes a tudakozót felhívni, vagy a szolgáltatók WEB-oldalán megkeresni a helyes beállításokat! Érdemes a díjszabásról is tájékozódni, hiszen a WAP-olás percdíjas!

Általában meg kell adnunk egy nyitóoldalt („**homepage**”), egy IP címet („**IP address**”), ki kell választanunk a vonal típusát („**analog**” vagy „**digital**”), a kapcsolat típusát („**connectionless**” vagy „**connection oriented**”), és a tárcsázandó telefonszámot („**Phone number**”).

Helyes beállítások használatával mindössze egy pár gombnyomás, és a telefon máris kapcsolódik az internethez! A mobiltelefonok használati útmutatójában részletesebb leírást találhatunk a WAP-funkciókról és a beállításokról.

Mi az a GPRS?

A GPRS tulajdonképpen egy **nagyságrendekkel gyorsabb kapcsolódási módszer**, melynek használatával sokkal hatékonyabban tudunk a mobiltelefonunkkal internetezni. A különbség a sima WAP és a GPRS között körülbelül olyan, mint a 9.6K-s modemes kapcsolat és az ISDN között. A GPRS-t támogató mobiltelefonok egyre gyakoribbak, viszont áruk is borsosabb!

WAP oldalak elhelyezése a tárhelyünkön

Ahhoz, hogy a tárhelyünkön elhelyezett WAP oldalakat mobiltelefonunkról el tudjuk érni, az adott tárhely-szolgáltatónak támogatnia kell a WAP oldalak futtatását! Mindenképpen járjunk utána ennek a szolgáltatónknál, mielőtt WAP oldalak fejlesztésébe kezdünk!

Például a Lycos támogatja a WAP oldalakat, tehát ez esetben nem kell aggódnunk! Egyszerűen töltsük fel a fájlokat egy külön könyvtárba, majd a mobiltelefonunkba írjuk be az elérési címet, mintha csak a WEB-böngészőnkbe írnánk be azt!