PHP – MySQL, Java – JDBC – MySQL

Adatbázisok az iskolában – 2012

Dr. Balázs Péter

Palatinus Endre és Erdőhelyi Balázs diái alapján

Mi a PHP?

- A PHP (PHP: Hypertext Preprocessor) egy nyílt forráskódú, számítógépes szkriptnyelv, legfőbb felhasználási területe a **dinamikus weboldalak készítése**.
- A PHP-kódunk általában **egy web-szerveren fut**, ahol egy PHP-elemző végrehajtja a kódban lévő PHP nyelvbeli utasításokat, és az eredményt továbbítja a kliens böngészője felé.
- Tehát ha a PHP-programunk HTML-kódokat írat ki, akkor dinamikus HTML oldalakat generálhatunk

XAMPP

- A XAMPP elnevezés egy betűszó:
  - X (azaz, platformfüggetlen)
  - Apache HTTP Server – webszerver
  - MySQL - adatbázis
  - PHP – programozási nyelv
  - Perl – programozási nyelv
- Számunkra elegendő a XAMPP Lite csomag is.
- http://www.apachefriends.org/en/xampp.html
- Itt válasszuk ki az oprendszert és a Lite verziót!
- Pendrive-ra is telepíthető

XAMPP - folytatás

- Miután feltelepítettük a XAMPP-ot, indítsuk el a **xampp control center** alkalmazást!
- Itt kezelhetjük a XAMPP-ot alkotó alkalmazások elindítását és leállítását. Indítsuk el az **Apache** webszervert (előtte a Skype-ot zárjuk be)!
- A XAMPP telepítési könyvtárában lévő **htdocs** mappába kell elhelyeznünk azokat a fájlokat, amelyeket elérhetővé szeretnénk tenni a weboldalunkról.
- A böngészőbe beírva ezen a címen érhetjük el ezeket a fájlokat: http://localhost/relatív_útvonal

A PHP kód

- A PHP kód egy **.php kiterjesztésű** fájlban található.
- Ebben a **<?php** és **?>** tag-ek közé kell elhelyeznünk a PHP-kódunkat.
- Ami ezen kívül esik, az HTML-kódként értelmeződik.
- Tetszőlegesen váltogathatják egymást a fájlban a PHP és HTML kódrészletek.
- Hozzunk létre egy index.php nevű fájlt a példáknak a webszerverünk megfelelő mappájában! (...\XAMPP\htdocs)

Az első programunk

- A PHP-kódunkat legtöbbször egy HTML oldal törzsébe ágyazzuk. Erre egy példa:
- ```
  <html>
   <head>
   <title>”Elso programunk”</title>
   </head>
   <body>
   <?php
   echo "Hello World!\n";
   ?>
   </body>
  </html>
  ```
  ```

Változók

- Változókat a **`$változónév = kezdőérték;`** utasítással deklarálhatunk.
- A változók neve deklaráláskor és használatukkor is a $-jellel kezdődik.
- A típust nem kell megadnunk, mert azt a PHP automatikusan kitalálja, és mellesleg dinamikus is.
- ```
  <?php
  $txt="Hello World!";
  $x=16;
  ?>
  ```
  ```

Szöveges változók

- Egyesítés (konkatenáció): . (pont)
- ```
  <?php
  $txt1="Hello World!";
  $txt2="What a nice day!";
  echo $txt1 . " " . $txt2;
  ?>
  ```
- Sztring hossza: `strlen("Hello world!");`
- Keresés: `strpos("Hello world!","world");`
- Az első egyezés pozícióját, vagy `FALSE`-ot ad vissza.
  ```

Operátorok

- A C nyelvben megszokott módon. Kivétel az osztás, ami mindig valós osztást jelent, a % pedig egész értéket ad vissza.
- Az aritmetikai operátoroknál itt is használhatjuk az összevont értékadási formákat: `$a += $b;`
- Logikai operátorok: `==, !=, <, <=, >, >=`

If, elsif, else

```
<?php
$d=date("D");
if ($d=="Fri")
  echo "Kellemes hétvégét!";
elseif ($d=="Mon")
  echo "Hétfőn még a fű se nől!";
else
  echo "Szép napot!";
?>
```

- `date("D")` – a hét napjának lekérdezése

Tömbök

- Kétféleképpen hozhatunk létre tömböket:
- A tömbelemek felsorolásával az indexek automatikusan hozzárendelődnek 0-tól kezdődően:
```
$autok=array("Saab","Volvo","BMW",
"Toyota");
```
- A tömbelemeket egyenként megadva az indexeket mi magunk határozzuk meg:
```
$autok[0]="Saab"; $autok[1]="Volvo";
$autok[2]="BMW"; $autok[3]="Toyota";
```
- ```
echo $autok[0] . " és " . $autok[1] .
" svéd autók.";
```
```

Asszociatív tömbök

- Kulcs-érték párokat tartalmazó adatszerkezet.
- Az elemek felsorolásával:

```
$eletkorok = array("Péter"=>32,
"András"=>30, "Tibor"=>34);
```

- Az elemek egyenkénti megadásával:

```
$eletkorok['Péter'] = "32";
$eletkorok['András'] = "30";
$eletkorok['Tibor'] = "34";
```

-
```
echo "Péter " . $eletkorok['Péter'] .
" éves.";
```

Ciklusok

- ```
  $i=1;
  while ($i<=5)  { $i++;  }
  ```
- ```
  $i=1;
  do  {  $i++;  }
  while ($i<=5);
  ```
- ```
  for ($i=1; $i<=5; $i++)  {
    echo "i = " . $i . "<br />";
  }
  ```
  ```

A foreach ciklus

- Tömböket járhatunk be a segítségével.
- ```
  <html>
  <body>
  <?php
  $szamok=array("egy","keet","haar");
  foreach ($szamok as $ertek) {
    echo $ertek . "<br />";
  }
  ?>
  </body>
  </html>
  ```
  ```

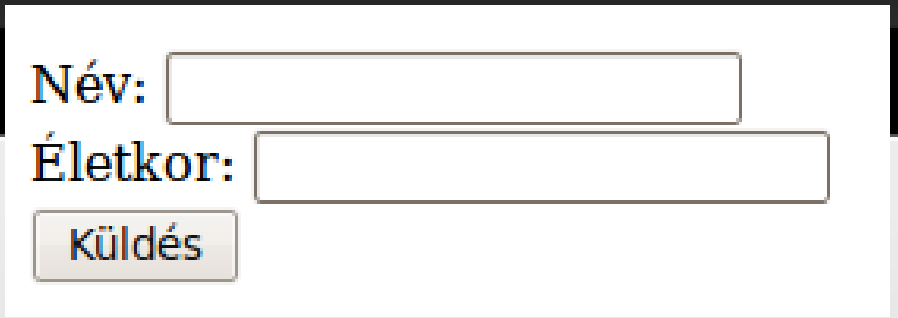
A foreach ciklus - folytatás

- Asszociatív tömbök esetén a kulcs-érték párokat is bejárhatjuk vele:
- ```
  <?php
  $szamok=array("egy"=>1,"keet"=>2,
  "haar"=>3);
  foreach ($szamok as $kulcs => $ertek)
  {
    echo $kulcs."=".$ertek."<br>";
  }
  ?>
  ```
  ```

Függvények

- ```
  <?php
  function osszead ($a,$b) {
  $osszeg=$a+$b;
  return $osszeg;
  }
  echo "2 + 2 az néha " . osszeg(2,3);
  ?>
  ```
- A visszatérési érték típusát nem kell megadni, mert a típusok dinamikusak a PHP-ben.
- Nem kötelező visszaadni egy értéket, lehet egyszerű eljárást is írni (a `return` elhagyásával).
  ```

Űrlapok kezelése




- ```
  <html>
  <body>
  <form action="udvozol.php" method="post">
  Név: <input type="text" name=„Név" /> <br>
  Életkor: <input type="text" name=„Életkor"
  /> <br>
  <input type="submit" value=”Küldés”/>
  </form>
  </body>
  </html>
  ```
- Az űrlapot kitöltve, és a „Küldés” gombra kattintva a böngészönk az „udvozol.php” oldalra ugrik.
  ```

Űrlapok kezelése - folytatás

- Az „udvozol.php” tartalma a következő:
- ```
  <html>
  <body>
  Szia <?php echo $_POST["nev"]; ?>!<br />
  Te <?php echo $_POST["eletkor"]; ?> éves
  vagy.
  </body>
  </html>
  ```
- A `$_GET` és a `$_POST` változókkal kérhetjük le az űrlapokban megadott információkat.
  ```

A $_GET és $_POST asszociatív tömbök

- Ha HTML űrlapunkban, azaz a form elemünkben a method attribútumnak a "get" és a "post" értékeket adhatjuk meg.
- Ettől függően kell a `$_GET` vagy a `$_POST` tömböt használnunk a PHP kódunkban az űrlapban megadott adatok elérésére.

Get vagy post?

- Ha a get metódussal küldjük az adatokat tovább, akkor azok a URL-ben meg fognak jelenni, és azok nem lehetnek bármilyen hosszúak, míg a POST metódusra ezek nem igazak.
- `http://localhost/proba/udvozol.php?nev=Endre&eletkor=22` – ez get esetén lesz így.
- A get metódussal jelszavakat ne küldjünk!
- Ha könyvjelzőt szeretnénk létrehozni egy oldalhoz, akkor viszont szükséges a get metódus használata.

Gyakorló feladatok

- Írjunk egy olyan rekurzív függvényt, amely visszaadja az első n szám összegét!
- Hozzunk létre egy asszociatív tömböt, amely az általunk felvett kurzusok név-kreditérték párosait tartalmazza, majd számítsuk ki az általunk felvett kreditek számát egy foreach ciklussal!
- Kérd be egy személy adatait egy űrlapon, majd írassuk azokat ki egy másik PHP-oldalon!

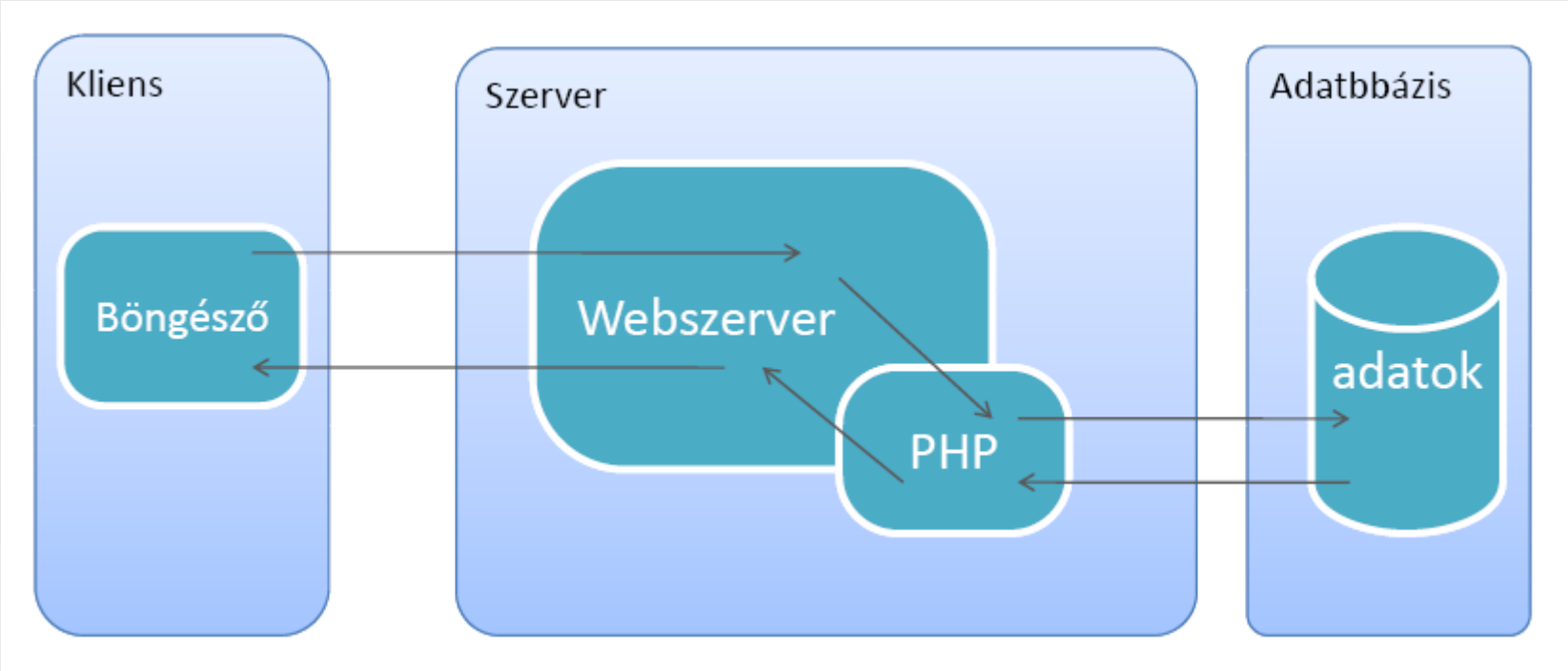
Weboldalak és adatbázisok

- A dinamikus weboldalak az esetek többségében valamilyen **adatbázist** használnak az oldalon **megjelenített adatok forrásaként**.
- Példák:
  - Webárúház: áruk, ügyfelek, rendelések
  - Fórum: témák, hozzászólások, felhasználók
  - Képgaléria: albumok, képek

MySQL

- A MySQL a legnépszerűbb nyílt-forráskódú **relációs adatbázis-kezelő rendszer**.
- A leggyakrabban a **PHP-val együtt** használják dinamikus weboldalak készítésére.
- Míg a PHP-t az Apache webszerver egy beépülő moduljaként használtuk, addig a MySQL egy **különálló szerverként** fut.
- Ehhez tudunk **konzolos** vagy **grafikus kliens** alkalmazásokkal, vagy a különböző **programozási nyelvekben API**-kon keresztül kapcsolódni.

A kapcsolat


Kliens
Böngésző
Szerver
Webszerver
PHP
Adatbbázis
adatok

Szükséges szoftverek

- Windows alatt a XAMPP (Lite) csomag
- phpMyAdmin: a MySQL könnyebb kezelhetőségét szolgáló, böngészőn keresztül elérhető alkalmazás

Kapcsolódás egy adatbázis kiszolgálóhoz

- Első lépésként csatlakoznunk kell egy MySQL kiszolgálóhoz, és azon kiválasztani egy adatbázist.
-
```
<?php
$kapcsolat = mysql_connect("szerver",
    "azonosító", "jelszó");

mysql_select_db("proba", $kapcsolat);
?>
```

Kapcsolódás egy adatbázis kiszolgálóhoz - folytatás

- A `mysql_connect()` sikeres csatlakozás esetén a kapcsolatra mutató változóval tér vissza, ellenkező esetben pedig `FALSE`-szal.
- A `mysql_select_db("adatbázis neve", $kapcsolat)` paranccsal tudjuk kiválasztani azt az adatbázist, amelyet használni szeretnénk.
- A MySQL kiszolgálóval létesített kapcsolat a szkript lefutásáig él, utána automatikusan megszakad.

Hibakezelés

- A kapcsolódás után hibakezelést is végezhetünk:
- ```
  if (!$kapcsolat) {
    die('Hiba a csatlakozáskor: ' .
  mysql_error());
  }
  ```
- A `die(”Hibaüzenet”)` paranccsal félbeszakíthatjuk a program futását, és kiírathatjuk a hiba okát.
- A `mysql_error()` függvény a legutóbbi MySQL-nek küldött utasítás során keletkezett esetleges hibaüzenettel tér vissza.
  ```

A példa adatbázisunk

- Feladat: Szeretnénk nyilvántartani a kurzusainkat egy adatbázisban. Szükségünk van egy weblapra, ahol kilistázhatjuk a kurzusaink nevét és kreditértékét, valamint új kurzusokat szúrhatunk be az adatbázisunkba.
- Relációséma: Kurzus (kód, név, kredit)
- Ezt a táblát fogjuk létrehozni és adatokkal feltölteni a phpMyAdmin segítségével. Később ezt SQL-parancsokkal is meg tudjuk majd oldani.

Létrehozás phpMyAdmin-ban

- Lépjünk be a phpMyAdmin alkalmazásba a következő címen: http://localhost/phpmyadmin
- Hozzunk létre egy **új adatbázist** ”proba” néven!
- Ebben hozzunk létre egy **új táblát** ”kurzusok” néven 3 mezővel!
- Adjuk meg a 3 **mezőt** a következők szerint:
  - kod: VACHAR típusú, 10 hosszúságú, PRIMARY Index
  - nev: VACHAR típusú, 30 hosszúságú
  - kredit: INT típusú

Adatok lekérdezése

- Listázzuk ki a kurzusaink nevét és kreditértékét!
- ```
  <?php
  $res = mysql_query("SELECT * FROM
  kurzusok");
  while($sor = mysql_fetch_array($res))
  {
   echo $sor['nev']." ".$sor['kredit'];
   echo "<br />";
  }
  ?>
  ```
  ```

Adatok lekérdezése - folytatás

- A `mysql_query("SELECT * FROM kurzusok")` parancs végrehajtja a paraméterben kapott lekérdezést és visszatér az eredménnyel.
- Ebben az esetben a `kurzusok` táblával.
- A `fetch_array` függvény lekérdezi egy eredménytábla következő sorát, amit **asszociatív** és egyszerű **tömbként** is használhatunk!
- Asszociatív tömbként kezelve az adattábla **oszlopainak nevére hivatkozva** érhetjük el a tábla sorainak egyes **mezőit**.

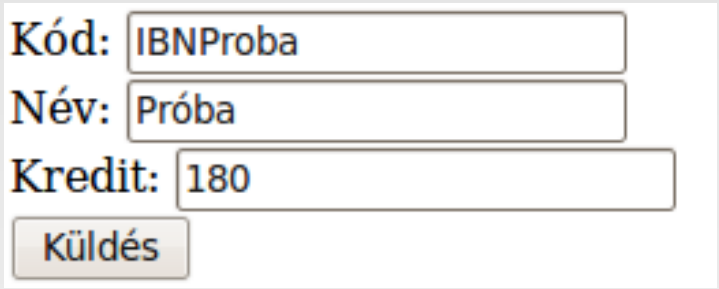
Írassuk ki táblázatban!

```
echo "<table border='1'>
<tr>
<th>Név</th>
<th>Kredit</th>
</tr>";
while($sor = mysql_fetch_array($res))  {
  echo "<tr>";
  echo "<td>" . $sor['nev'] . "</td>";
  echo "<td>" . $sor['kredit'] . "</td>";
  echo "</tr>";
}
echo "</table>";
```

Űrlapokból származó adatok

- Készítsük el az alább látható űrlapot, és a feldolgozást végző PHP-lapot (kurzus.{html,php})!
- ```
  $sql="INSERT INTO kurzusok (kod, nev, kredit)
  VALUES
  ('$_POST[kod]','$_POST[nev]','$_POST[kredit]')";

  $res = mysql_query($sql);
  ```

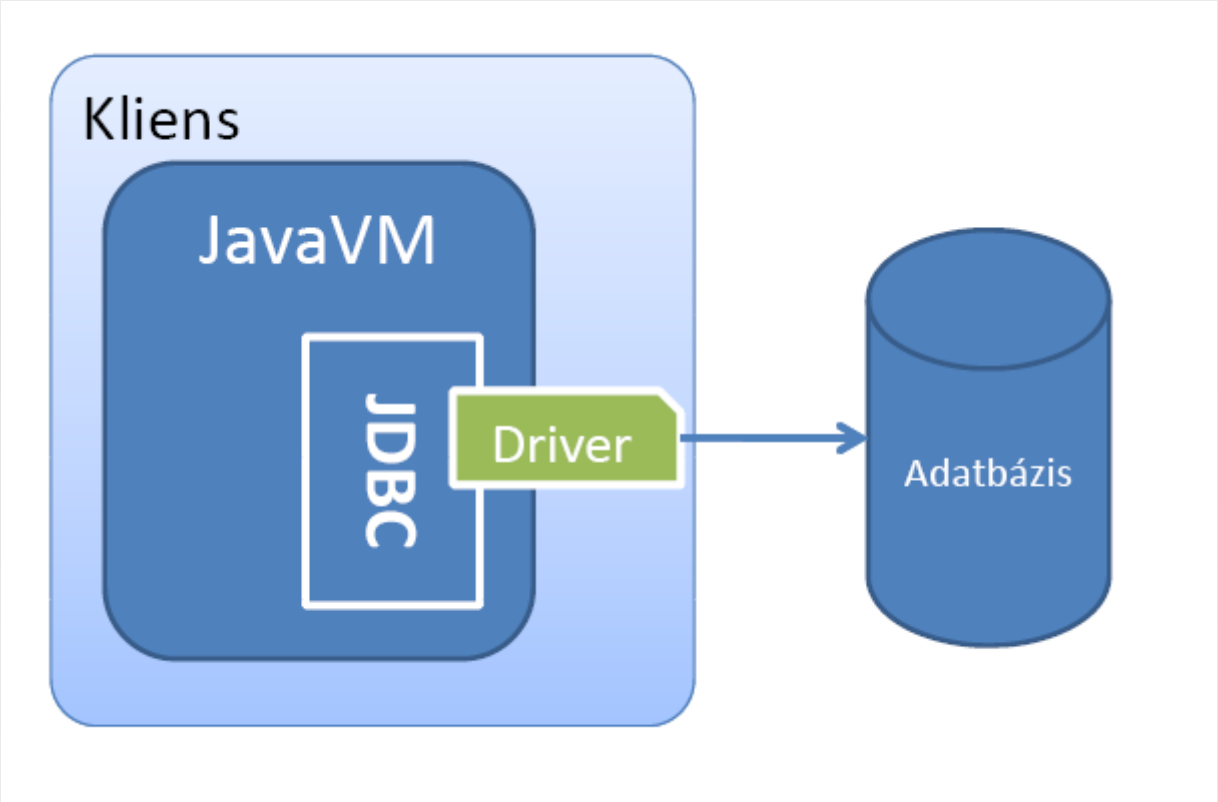


- Az INSERT INTO tábla(mezők) VALUES (értékek) utasítással adatokat szúrhatunk be egy táblába.
  ```

Gyakorló feladatok

- Hozzuk létre a következő sémával rendelkező táblát: Könyv (sorszám, szerző, cím) !
- Készítsünk egy oldalt, amelyen megjelennek a táblában lévő könyvek adatai táblázatos formában!
- Készítsünk egy oldalt az új könyvek felvételéhez!
- Készítsünk egy legördülő listát, amely a könyvek címét tartalmazza (, a kulcsa pedig a könyvek sorszáma). Ezt helyezzük el egy űrlapba, amiről egy olyan oldalra juthatunk, ami a kiválasztott könyv adatait jeleníti meg!

Java – JDBC – MySQL



Példa

```
Class.forName("com.mysql.jdbc.Driver");

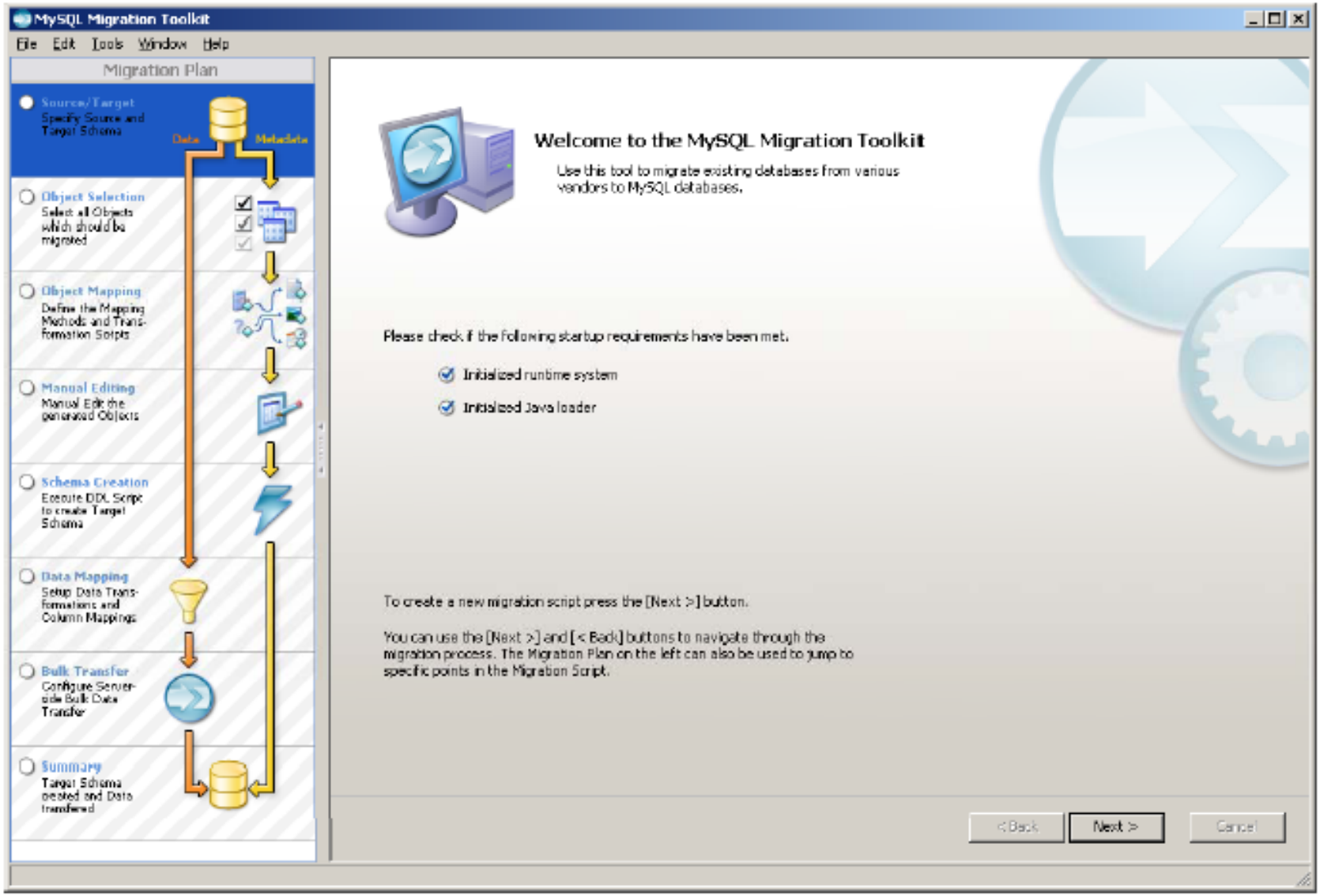
Connection conn = DriverManager.getConnection(
  "jdbc:mysql://servername/dbname", "", "");

Statement stmt = conn.createStatement();

ResultSet rs = stmt.executeQuery(
  "select * from uzenofal");

while ( rs.next() ) {
  System.out.println( rs.getObject("columnName"));
}
conn.close();
```

MySQL Migration Toolkit



Főbb lépések

- Forrás kijelölése
- Cél kiválasztása
- Objektumok kiválasztása
- Megfeleltetések módosítása
- Séma elkészítése
- Adatok átvitele