Java és web programozás

Kovács Kristóf

Budapesti Műszaki Egyetem

2015. 04. 08.

10. Előadás

Ami kimearad múlthéten

Ha már megvan a **KeyListener** vagy **MouseListener** osztályunk a következõ módon tudjuk hozzárendelni egy **JFrame** vagy **JPanel**-hez:

Ami kimearad múlthéten

Ha már megvan a **KeyListener** vagy **MouseListener** osztályunk a következõ módon tudjuk hozzárendelni egy **JFrame** vagy **JPanel**-hez:

- **addKeyListener(KeyListener kl)**: hozzáadja a paraméterként kapott **KeyListener**-t (tehát példányosítani kell a megírt **KeyListener**-ünket)
- **addMouseListener(MouseListener ml)**: hozzáadja a paraméterként kapott **MouseListener**-t
- **addMouseMotionListener(MouseMotionListener mml)**: a **MouseMotionListener** nem csak a kattintásokat, hanem az egér folyamatos mozgását figyeli, így kattintás nélkül kapunk belõle egér pozíció eseményeket

Ami kimearad múlthéten

Ha már megvan a **KeyListener** vagy **MouseListener** osztályunk a következõ módon tudjuk hozzárendelni egy **JFrame** vagy **JPanel**-hez:

- **addKeyListener(KeyListener kl)**: hozzáadja a paraméterként kapott **KeyListener**-t (tehát példányosítani kell a megírt **KeyListener**-ünket)
- **addMouseListener(MouseListener ml)**: hozzáadja a paraméterként kapott **MouseListener**-t
- **addMouseMotionListener(MouseMotionListener mml)**: a **MouseMotionListener** nem csak a kattintásokat, hanem az egér folyamatos mozgását figyeli, így kattintás nélkül kapunk belõle egér pozíció eseményeket

Ezeket mind a **JFrame** vagy **JPanel** konstruktorában kell meghívni tehát valójában amit futtatni fogtok az a **this.addMouseListener(...)**, de a **this**-t nem szükséges kiírni.

Ami kimearad múlthéten Pl

```
public class Game extends JFrame  {
...
    public GameBoard gameBoard;
    public Game() {
        gameBoard = new GameBoard();

        add(gameBoard);
        addMouseListener(new MouseController(this));

        setTitle("Sudoku");
        ...
    }
    ...
}
```

A **MouseController** konstruktora egy **GameBoard** objektumot kap.

SQLite

SQLite:

- Adatbázis kezelő rendszer
- SQL standardokat nagyrészt követi
- Nagyon elterjedt, pl böngészőkben is használt
- Nehéz olyan programnyelvet találni aminek ne lenne valamilyen SQLite kezelő könyvtára
- A forrása már hozzáférhető

SQLite

SQLite:

- Adatbázis kezelő rendszer
- SQL standardokat nagyrészt követi
- Nagyon elterjedt, pl böngészőkben is használt
- Nehéz olyan programnyelvet találni aminek ne lenne valamilyen SQLite kezelő könyvtára
- A forrása már hozzáférhető

sqlite3

- Frontend az SQLite-hoz
- Egyszerű használni, beszédes a kód

Nagy vonalakban az SQLite-ot úgy képzelhetjük el, mint egy Excel-t grafikus felület nélkül, táblázataink vannak, megnevezett oszlopokkal. Adatokat rakhatunk beléjük, törölhetünk adatokat, kiolvashatunk stb.

sqlite3

Nagy vonalakban az SQLite-ot úgy képzelhetjük el, mint egy Excel-t grafikus felület nélkül, táblázataink vannak, megnevezett oszlopokkal. Adatokat rakhatunk beléjük, törölhetünk adatokat, kiolvashatunk stb.

Terminálból az "sqlite3" paranccsal nyithatjuk meg az sqlite3-at, adhatunk neki rögtön egy adatbázis file-t, amin dolgozunk. Ha ez már létezik akkor azt használja, ha még nem, akkor létrehozza.

sqlite3

Nagy vonalakban az SQLite-ot úgy képzelhetjük el, mint egy Excel-t grafikus felület nélkül, táblázataink vannak, megnevezett oszlopokkal. Adatokat rakhatunk beléjük, törölhetünk adatokat, kiolvashatunk stb.

Terminálból az "sqlite3" paranccsal nyithatjuk meg az sqlite3-at, adhatunk neki rögtön egy adatbázis file-t, amin dolgozunk. Ha ez már létezik akkor azt használja, ha még nem, akkor létrehozza. Pl:
$ sqlite3 adatbazis.db

sqlite3

Nagy vonalakban az SQLite-ot úgy képzelhetjük el, mint egy Excel-t grafikus felület nélkül, táblázataink vannak, megnevezett oszlopokkal. Adatokat rakhatunk beléjük, törölhetünk adatokat, kiolvashatunk stb.

Terminálból az "sqlite3" paranccsal nyithatjuk meg az sqlite3-at, adhatunk neki rögtön egy adatbázis file-t, amin dolgozunk. Ha ez már létezik akkor azt használja, ha még nem, akkor létrehozza.
Pl:
$ sqlite3 adatbazis.db
Mostantól sqlite3-ban vagyunk.

Meta parancsok

Hasznos meta parancsok:

- *.tables* – kiadja az aktuális adatbázisban található táblázatok nevét
- *.mode column* – szépen oszlopokba adja ki a keresett adatokat
- *.headers on* – az oszlopok nevét kiírja a keresett adatok felé
- *.show* – megmutatja milyen opciókat állítottunk be
- *.schema* – megadja egy táblázat struktúráját, hogyan hoztuk létre
- *.dump* – kiadja a parancsokat, amikkel újra tudjuk építeni az adott táblázatot
- *.help* – ha elvesznétek

Táblázatok

CREATE TABLE nev(...); paranccsal hozhatunk létre egy táblázatot, a ... helyére az oszlopaink nevét és a bennük tárolt adatok típusát kell megadni. A lehetséges tipusok:

Táblázatok

CREATE TABLE nev(...); paranccsal hozhatunk létre egy táblázatot, a ... helyére az oszlopaink nevét és a bennük tárolt adatok típusát kell megadni. A lehetséges tipusok:

- *NULL* null érték
- *INTEGER* előjeles egész
- *REAL* előjeles lebegőpontos
- *TEXT* string
- vannak még mások is, pl. dátum típus is van

Táblázatok

CREATE TABLE nev(...); paranccsal hozhatunk létre egy táblázatot, a ... helyére az oszlopaink nevét és a bennük tárolt adatok típusát kell megadni. A lehetséges tipusok:

- *NULL* null érték
- *INTEGER* előjeles egész
- *REAL* előjeles lebegőpontos
- *TEXT* string
- vannak még mások is, pl. dátum típus is van

Pl:

CREATE TABLE Neptun(id integer, NeptunKod text, Nev text, Atlag real);

Táblázatok

CREATE TABLE nev(...); paranccsal hozhatunk létre egy táblázatot, a ... helyére az oszlopaink nevét és a bennük tárolt adatok típusát kell megadni. A lehetséges tipusok:

- *NULL* null érték
- *INTEGER* előjeles egész
- *REAL* előjeles lebegőpontos
- *TEXT* string
- vannak még mások is, pl. dátum típus is van

Pl:

CREATE TABLE Neptun(id integer, NeptunKod text, Nev text, Atlag real);

- *DROP TABLE nev;* táblazat törlése
- *ALTER TABLE tablazat1 RENAME TO tablazat2;* táblázat átnevezése
- Az *ALTER TABLE tablazat1 ADD COLUMN Email text;* táblázatba új oszlop felvétele

Szintaxis

Amiket eddig észrevehettünk a szintaxissal kapcsolatban:

- a parancsok végere ; (pontosvesszőt) kell raknunk
- a kis és nagybetű változónevektől eltekintve nem számít, tehát a *create table...* ugyanazt eredményezi mint a *CREATE TABLE...*
- egy parancson belül a változókat / oszlopokat , (vesszővel) választjuk el
- sortörések nem számítanak
- a parancsok ismerete nélkül is szinte olvasható

Insert

Az *INSERT* paranccsal írhatunk új sorokat a táblázatunkba, a szintaxis a következő (feltéve hogy a TablazatNeve táblázatunk 3 oszlopos):

- INSERT INTO TablazatNeve(oszlop1, oszlop2, oszlop3) VALUES(ertek1, ertek2, ertek3);
- INSERT INTO TablazatNeve(oszlop1, oszlop3) VALUES(ertek1, ertek3);
- INSERT INTO TablazatNeve VALUES(ertek1, ertek2, ertek3);

Insert

Az *INSERT* paranccsal írhatunk új sorokat a táblázatunkba, a szintaxis a következő (feltéve hogy a TablazatNeve táblázatunk 3 oszlopos):

- INSERT INTO TablazatNeve(oszlop1, oszlop2, oszlop3) VALUES(ertek1, ertek2, ertek3);
- INSERT INTO TablazatNeve(oszlop1, oszlop3) VALUES(ertek1, ertek3);
- INSERT INTO TablazatNeve VALUES(ertek1, ertek2, ertek3);

PL:
INSERT INTO Neptun(id, NeptunKod, Nev) VALUES(23, 'C3WRGQ', 'Kovacs Kristof');

Delete

Sort törölni a *DELETE* paranccsal tudunk:

- DELETE FROM TablazatNeve; – kitörli az összes sorát a táblázatnak
- DELETE FROM TablazatNeve WHERE oszlop1 = valamiErtek; – kitörli azt a sorát a táblázatnak aminek az oszlop1-e valamiErtek
- DELETE FROM TablazatNeve WHERE oszlop2 like valamiResz; – azokat a sorokat törli ki aminek az oszlop2 oszlopában olyan adat van amiben szerepel a valamiResz

Delete

Sort törölni a *DELETE* paranccsal tudunk:

- DELETE FROM TablazatNeve; – kitörli az összes sorát a táblázatnak
- DELETE FROM TablazatNeve WHERE oszlop1 = valamiErtek; – kitörli azt a sorát a táblázatnak aminek az oszlop1-e valamiErtek
- DELETE FROM TablazatNeve WHERE oszlop2 like valamiResz; – azokat a sorokat törli ki aminek az oszlop2 oszlopában olyan adat van amiben szerepel a valamiResz

Pl:
DELETE FROM Neptun WHERE id = 23;
DELETE FROM Neptun WHERE NeptunKod like '3RG';

Update

Az *UPDATE* paranccsal tudunk már létező helyre új adatot beírni:

- UPDATE TablazatNeve SET oszlop2 = valami1 WHERE oszlop1 = valami2;

Update

Az *UPDATE* paranccsal tudunk már létező helyre új adatot beírni:

- UPDATE TablazatNeve SET oszlop2 = valami1 WHERE oszlop1 = valami2;

Pl:
UPDATE Neptun SET NeptunKod = 'C3WRGS' WHERE id = 23;

Select

Az utolsó alap parancs a *SELECT* amivel kiolvashatunk adatokat táblázatokból:

- SELECT * FROM TablazatNeve; – minden adatot kiolvas a táblázatból
- SELECT oszlop1, oszlop3 FROM TablazatNeve; – csak adott oszlopok adatait olvassa ki
- SELECT * FROM TablazatNeve LIMIT 4; – csak az első 4-et fogja kiírni
- SELECT * FROM TablazatNeve LIMIT 4 OFFSET 3; – 4-et ír ki, de a 3.-tól kezdve
- SELECT * FROM TablazatNeve ORDER BY oszlop2; – az oszlop2 szerint rendezve adja ki
- SELECT * FROM TablazatNeve ORDER BY oszlop2 DESC; – az oszlop2 szerint csökkenő sorrendben adja ki
- SELECT * FROM TablazatNeve WHERE oszlop2 = valami; – csak azokat írja ki amikre teljesül az adott feltétel

Select példák

```
SELECT * FROM Neptun;

SELECT Nev FROM Neptun;

SELECT Nev, id FROM Neptun;

SELECT * FROM Neptun ORDER BY Nev;

SELECT NeptunKod FROM Neptun ORDER BY Atlag DESC
LIMIT 3;
```

Hasznos dolgok

Ha van olyan oszlopa a táblázatunknak amire egyfajta kulcsként tekinthetünk, talán csak egy számláló, vagy ilyesmi, akkor azt definiálhatjuk *primary key*-nek:

- CREATE TABLE Neptun(id integer primary key, Nev text...);

Ha így adjuk meg az id-t, akkor alapból szerinte fogja rendezni a kiadott adatokat, és ha nem adjuk meg egy *INSERT*-nél az id értékét, akkor lépteti az előző sorból.

Hasznos dolgok

Ha van olyan oszlopa a táblázatunknak amire egyfajta kulcsként tekinthetünk, talán csak egy számláló, vagy ilyesmi, akkor azt definiálhatjuk *primary key*-nek:

- CREATE TABLE Neptun(id integer primary key, Nev text...);

Ha így adjuk meg az id-t, akkor alapból szerinte fogja rendezni a kiadott adatokat, és ha nem adjuk meg egy *INSERT*-nél az id értékét, akkor lépteti az előző sorból.

Ha lesz olyan oszlopunk ahova nem mindig adunk meg értéket, de szeretnénk ha szerepelne ott valami mindig, akkor használhatjuk a *default* értéket:

- CREATE TABLE Neptun(id integer primary key, Nev text, NeptunKod text, Atlag real default 5.0);

Így ha nem adunk meg átlagot egy sorban akkor oda 5.0-t rak.

Mindez javaban

- import java.sql.*; – be kell töltenünk
- connection = DriverManager.getConnection("jdbc:sqlite:hallgatok.db"); – így tudjuk betölteni a hallgatok.db adatbázist
- Statement statement = connection.createStatement(); – kell ahhoz, hogy utasításokat tudjunk adni
- statement.executeQuery("select * from person"); – ezek után így tudunk futtatni sqlite3 parancsokat rajta, amit ez visszaad az egy iterálható dolog, szóval egy for vagy while ciklussal végig tudunk menni rajta
- statement.executeUpdate("insert into hallgatok (kod, nev, szak) values (c2, kk, tt)") – így pedig tudunk új értékeket rakni az adatbázisba

További leírás: https://bitbucket.org/xerial/sqlite-jdbc