

תכנות באינטרנט

המחלקה הסטטית

Helper

גלעד מרקמן

קריית החינוך פארק המדע,
נס ציונה

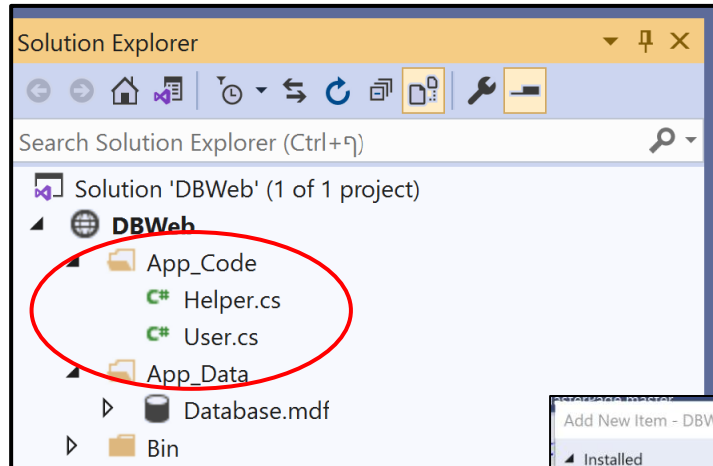


קריית החינוך
השש שנתית
פארק המדע
עמיד של חדשנות

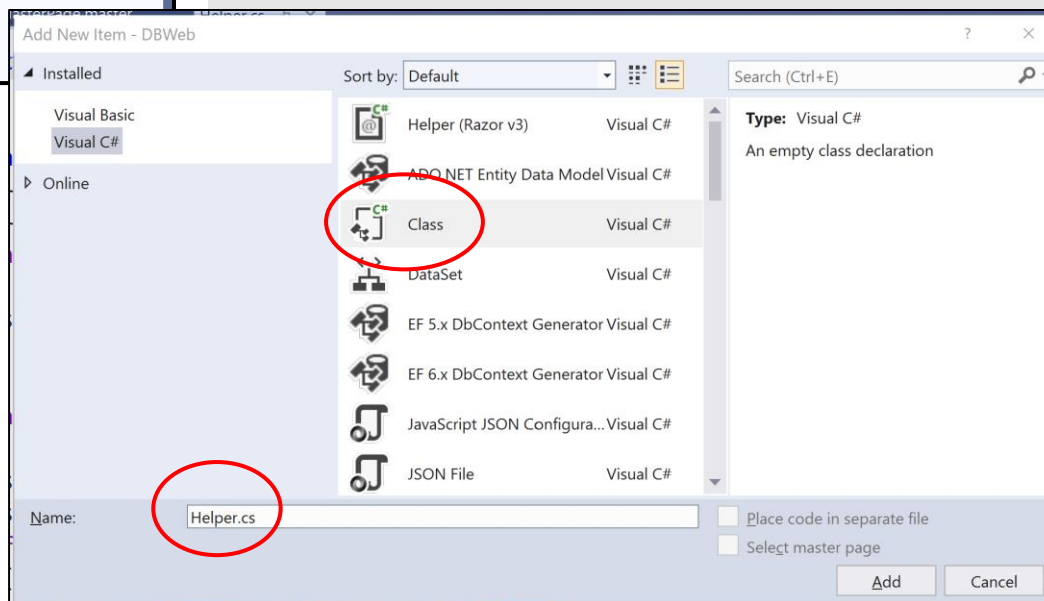


- על מנת לחסוך בכתיבת קוד אנו נקים מחלקה סטטית שתקרא Helper, ובה נכתוב את הפעולות שמתקשרות על בסיס הנתונים שלנו.
- אנו נשתמש בפעולות שמוגדרות במחלקה זו בכל הדפים שלנו בפרויקט.
- המחלקה הינה מחלקה Static כך שניתן להפעיל את הפעולות ללא צורך ביצירת אובייקט מיוחד לצורך כך.
- המחלקה גם תכלול קבועים המשמשים אותנו להתקשרות עם בסיס הנתונים, כמו ה- connection String.
- בנוסף נקים מחלקה נוספת שתכונה User התואמת את השדות בבסיס הנתונים שלנו, ובתוך אובייקט כזה נשמור את הנתונים שהתקבלו מבסיס הנתונים.

המחלקה Helper



- את המחלקה ניצור בספריה מיוחדת שתקרא App_Code בספריה הראשית של הפרויקט.
- יצירת מחלקה נעשית באמצעות Add Item.



```
1 reference
public static class Helper1
{
    0 references
    public Helper1()
    {
        //
        // TODO: Add constructor logic here
        //
    }
}
```

- לאחר שיוצר הקובץ יש להוסיף לכתרת המחלקה את המילה השמורה static, ולמחוק את הבנאי שנוצר אוטומטית.

הגדרת קבועים למחלקה

- בתחילת המחלקה נוסיף שמות של קבועים (const) שימשו אותנו בכל המחלקה:
 - DBName – שם בסיס המידע שיצרנו.
 - tblName – שם הטבלה של המשתמשים ביצרנו.
 - connectionString – הגדרת מחרוזת ההתקשרות עם בסיס הנתונים.
- המחרוזת הוגדרה באמצעות נתיב יחסי (|DataDirectory|), וזאת בהנחה ששמרנו את בסיס הנתונים בספרייה שנקראת App_Data בשורש הפרויקט.

5 references

```
public static class Helper
{
    public const string DBName = "Database.mdf";
    public const string tblName = "tblUsers";
    public const string connectionString = @"Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=|DataDirectory|\\"
        + DBName + ";Integrated Security=True";
}
```

```
1 reference
public static DataSet RetrieveTable(string SQLStr)
{
    // connect to DataBase
    SqlConnection con = new SqlConnection(conString);

    // Build SQL Query
    SqlCommand cmd = new SqlCommand(SQLStr, con);

    // Build DataAdapter
    SqlDataAdapter ad = new SqlDataAdapter(cmd);

    // Build DataSet to store the data
    DataSet ds = new DataSet();

    // Get Data form DataBase into the DataSet
    ad.Fill(ds, tblName);

    return ds;
}
```

- חתימת הפעולה חייבת לכלול את המילה static.
- הפעולה מקבלת משתנה מסוג מחרוזת הכולל פקודת SELECT בשם SQL.
- שם הטבלה ב dataSet יהיה כשם הקבוע tblName שהגדרנו בראש המחלקה.
- הפעולה מחזירה אובייקט מסוג DataSet עם הטבלה שהתקבלה מבסיס הנתונים.

- חתימת הפעולה היא static.
- הפעולה מקבלת אובייקט מסוג DataTable שהתקבל מבסיס הנתונים באמצעות הפעולה RetrieveTable.
- הפעולה מחזירה מחרוזת HTML עם טבלת המשתנים.

0 references

```
public static string BuildSimpleUsersTable(DataTable dt)
{
    string str = "<table class='usersTable' align='center'>";
    str += "<tr>";
    foreach (DataColumn column in dt.Columns) ...

    foreach (DataRow row in dt.Rows) ...
    str += "</tr>";
    str += "</Table>";
    return str;
}
```

- כאמור, כדי להשתמש בפעולות הכתובות במחלקה סטטית אין צורך להגדיר אובייקט של המחלקה עם המילה השמורה `.new`.
- הפניה למחלקה מתבצעת עם שם המחלקה, נקודה ולאחריה שם הפעולה.
- השתמשנו הן בפעולה `RetrieveTable`, והן בפעולה `BuildSimpleUserTable`.

```
if (!IsPostBack)
{
    string SQLStr = "SELECT * FROM " + Helper.tblName;
    DataSet ds = Helper.RetrieveTable(SQLStr);
    DataTable dt = ds.Tables[Helper.tblName];
    string table = Helper.BuildSimpleUsersTable(dt);
    tableDiv.InnerHtml = table;
}
```


0 references

```
public static int ExecuteNonQuery(string SQL)
{
    // התחברות למסד הנתונים
    SqlConnection con = new SqlConnection(conString);

    // SQL בניית פקודת
    SqlCommand cmd = new SqlCommand(SQL, con);

    // ביצוע השאילתא
    con.Open();
    int n = cmd.ExecuteNonQuery();
    con.Close();

    // return the number of rows affected
    return n;
}
```

```
{
    // בניית השורה להוספה
    SQL = $"INSERT INTO tblUsers (firstName, lastName, userName, password, birthday, city) " +
        $"VALUES ('{Request.Form["firstName"]}', '{Request.Form["lastName"]}', " +
        $" '{Request.Form["userName"]}', '{Request.Form["password"]}', " +
        $" '{Request.Form["date"]}', '{Request.Form["city"]}');"
    Helper.ExecuteNonQuery(SQL);
    Response.Redirect("http://localhost:59467/Pages/Main.aspx");
}
```

- הפעולה ExecuteNonQuery מבצעת פעולה בבסיס הנתונים בשיטה "המחוברת" ("Connected").
- הפעולה מקבלת מחרוזת עם פקודת SQL ומבצעת אותה מול בסיס הנתונים.
- דוגמה לזימון של הפעולה לצורך הוספת משתמש לבסיס הנתונים

בניית המחלקה User

```
6 references
public class User
{
    public int userId;
    public string userName;
    public string password;
    public string firstName;
    public string lastName;
    public DateTime birthday;
    public string city;
    public bool Admin;
    2 references
    public User()
    {
        userId = -1;
        userName = "";
        password = "";
        firstName = "";
        lastName = "";
        birthday = DateTime.Today;
        city = "";
        Admin = false;
    }
}
```

- המחלקה User כוללת מאפיינים התואמים את טבלת המשתמשים בבסיס הנתונים שלנו.
- כמו כן, כוללת המחלקה בנאי פשוט.
- המחלקה אינה מסוג static, ולכן כדי להשתמש בה נצירך ליצור אובייקט באמצעות המילה new.
- מחלקה זו תשמש אותנו כדי להעביר ולקבל נתונים לפעולות הסטטיות שלנו.

```
public static void Update(User user)
{
    // HttpRequest Request
    // התחברות למסד הנתונים
    SqlConnection con = new SqlConnection(connectionString);

    // בניית פקודת SQL
    string SQLStr = "SELECT * FROM " + Helper.tblName + $" Where userid={user.userId}";
    SqlCommand cmd = new SqlCommand(SQLStr, con);

    // טעינת הנתונים לתוך DataSet
    DataSet ds = new DataSet();
    SqlDataAdapter adapter = new SqlDataAdapter(cmd);
    adapter.Fill(ds, tblName);

    // בניית השורה להוספה
    DataRow dr = ds.Tables[tblName].Rows[0];
    dr["firstName"] = user.firstName;
    dr["lastName"] = user.lastName;
    dr["userName"] = user.userName;
    dr["password"] = user.password;
    dr["birthday"] = user.birthday;
    dr["city"] = user.city;

    // עדכון הדאטה סט בבסיס הנתונים
    SqlCommandBuilder builder = new SqlCommandBuilder(adapter);
    adapter.UpdateCommand = builder.GetUpdateCommand();
    adapter.Update(ds, tblName);
}
```

- הפעולה מקבלת אובייקט של משתמש ובו פרטים מעודכנים של המשתמש.
- הפעולה מאתרת לפי userId את השורה בטבלה.
- לאחר מכן מעדכנת את שורת הנתונים של המשתמש לפי הנתונים החדש באובייקט user.
- לבסוף, הפעולה מעדכנת את בסיס הנתונים.

שימוש נוסף במחלקה user

```
public static User GetRow(string userName, string password)
{
    // התחברות למסד הנתונים
    SqlConnection con = new SqlConnection(conString);

    // בניית פקודת SQL
    string SQL = $"SELECT firstName, admin FROM " + tblName +
        $" WHERE userName='{userName}' AND password = '{password}'";
    SqlCommand cmd = new SqlCommand(SQL, con);

    // ביצוע השאילתא
    con.Open();
    SqlDataReader reader = cmd.ExecuteReader();

    // שימוש בנתונים שהתקבלו
    User user = new User();
    if (reader.HasRows)
    {
        reader.Read();
        user.userName = reader.GetString(0);
        user.Admin= reader.GetBoolean(1);
    }
    else
    {
        user.userName = "Visitor";
        user.Admin = false;
    }
    reader.Close();
    con.Close();
    return user;
}
```

- הפעולה בודקת בבסיס הנתונים אם שם משתמש וסיסמה קיימים. אם כן, מחזירה את השם הפרטי והשדה .admin

- הפעולה מקבלת שם משתמש וסיסמה ומחזירה אובייקט user ובו מעודכנים רק השם הפרטי והשדה .admin

- יצירת מחלקה סטטית ובה הפעולות המאפשרות התחברות עם בסיס הנתונים.
- הגדרת קבועים במחלקה.
- הגדרת connectionString עם נתיב יחסי ל App_Data.
- שימוש במחלקה סטטית בפרויקט, ללא יצירת אובייקט חדש.
- הגדרת מחלקה רגילה של משתמש.
- שימוש במחלקה משתמש להעברת נותנים בין המחלקה הסטטית לבין התוכנית המזמנת אותה בדף.
- צירפנו לאתר את המחלקה הסטטית Helper ובה יש פעולות נוספות שיעזרו לכם בפרויקט.