

## Asp.net Learn:

2 - 44	<b>ASP Start</b>	<b> כתיבת האפליקציה (*)</b>
		<b>פְּקָדִים</b>
		<b>State</b>
		<b>הציג ועבודה עם נתונים</b>
45 - 76	<b>ASP Advanced</b>	<b>SQL</b>
		<b>פיתוח חכם</b>
		<b>בטחה</b>
77 - 110	<b>ASP Summary</b>	<b>פְּקָדִים ראשיים ^ &amp; ^</b>
		<b>ולידיציות</b>
		<b>מעברי עמודים</b>
		<b>דף { אב }</b>
		<b>מודי נתונים</b>





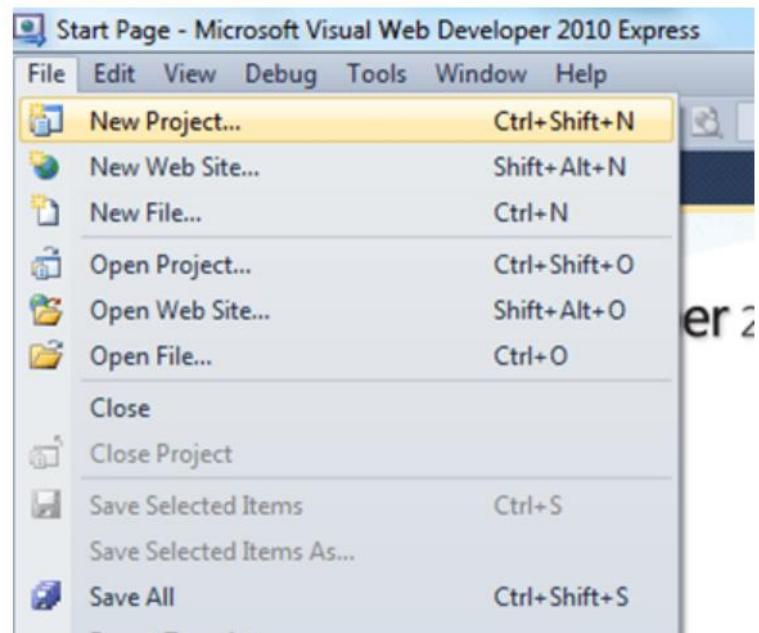
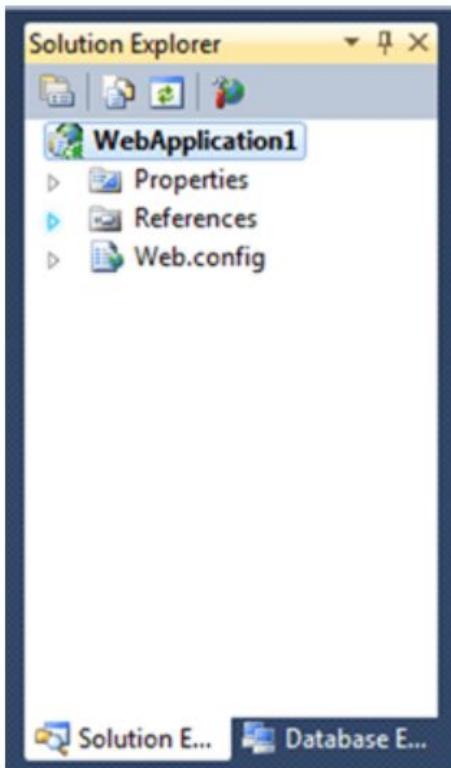
# מדריך ASP.NET – ייצירת פרויקט

Sela • IdoFlatow



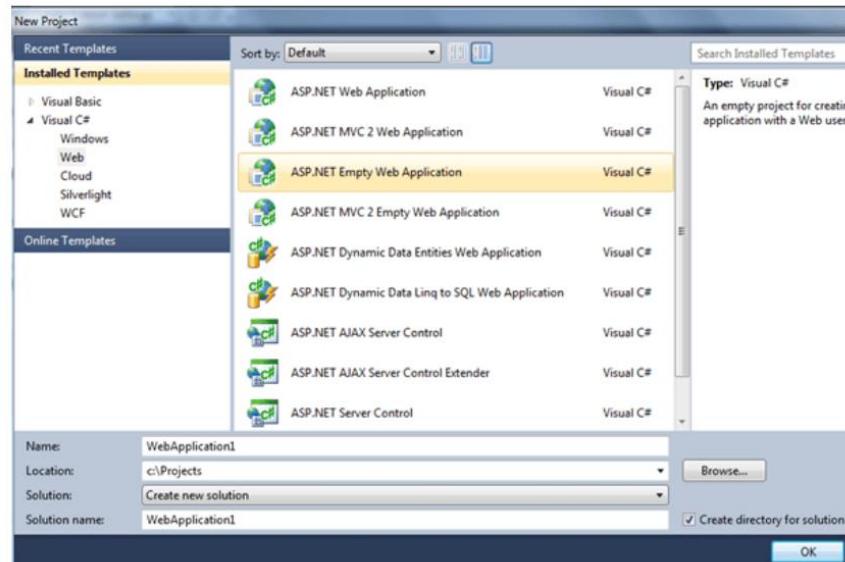
בחלק זה נראה כיצד להשתמש בסביבת העבודה Visual Web Developer על מנת ליצור פרויקט ASP.NET ראשון.

ראשית, נפתח את ה – .File → New Project וنبחר ב – Visual Web Developer



בחלון שיפתח , נבחר בקטgorיה WEB, שם נבחר ב – ASP.NET Empty Web Application – .  
נימן לפרוייקט שלנו את השם הרצוי, את התיקיה בה נרצה שהקבצים ישמרו ואת שם ה – .Solution

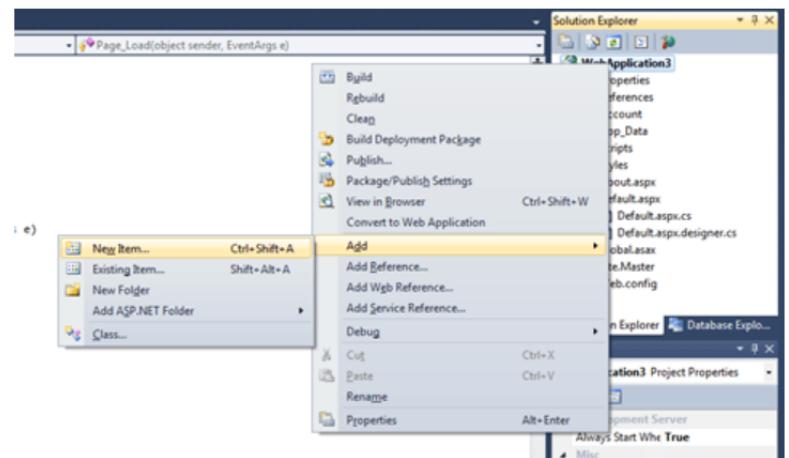
שיםו לב שבאופן אוטומטי ה – Solution מקבל את שם הפרוייקט, אך ניתן כmorן לשנות זאת.



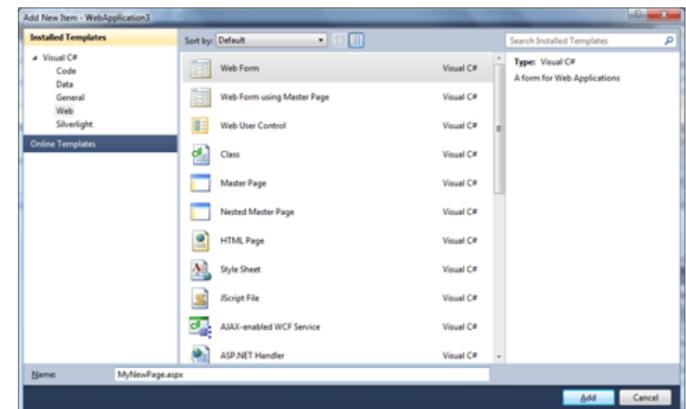
לאחר שיםנו ולחצנו על OK נראה את הפרוייקט שנוצר . נוכל לראות שנוצר לנו פרוייקט ריק ללא דפים ועם קובץ אחד בשם Web.Config שהוא קובץ הגדרות כללי לפרויקט.

במדריך זה נלמד כיצד להויסי דף ASPX חדש לתוכה פרויקט שלנו.

לאחר שיצרתם פרויקט WebDeveloper בתוכה Visual Studio 2010 Web Developer סמנו את הפרויקט בחולון-e, לחזו על הלחצן ימני של העכבר, בחרו באפשרות Add וاذ בחרו באפשרות **.New Item**.

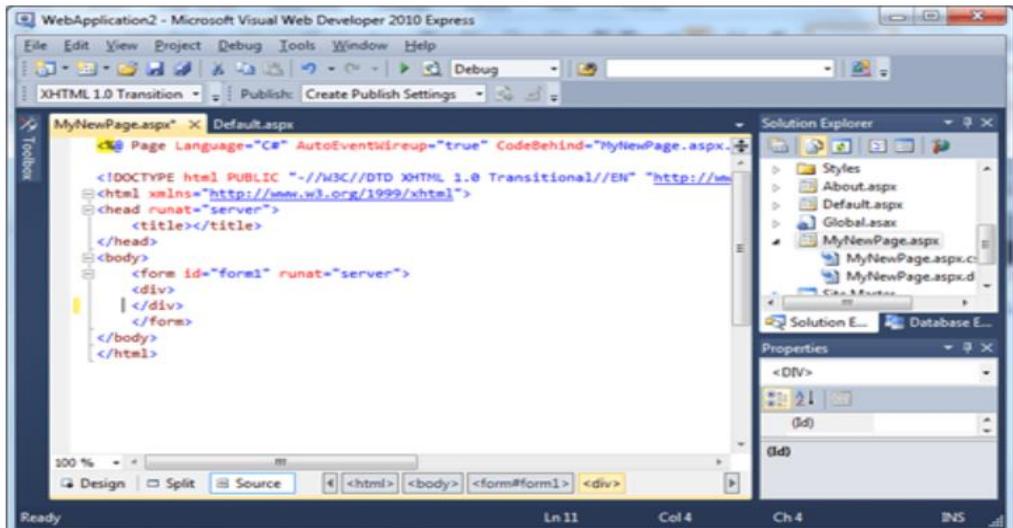


בתפריט שיפתח בחרו באפשרות Web Form (האפשרות הראשונה), תנו לדף את השם שאת חוץם ולחזו על כפתור Add.

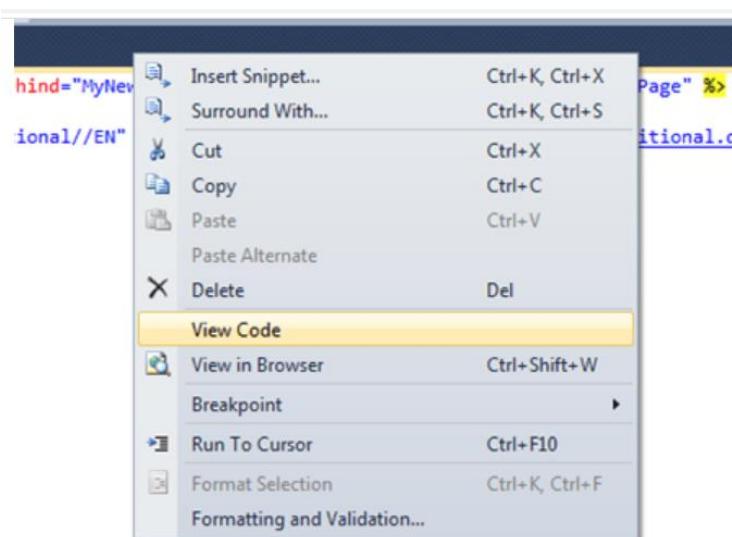


כעת נוכל לראות את הדף שפתחנו בפרויקט שלנו.

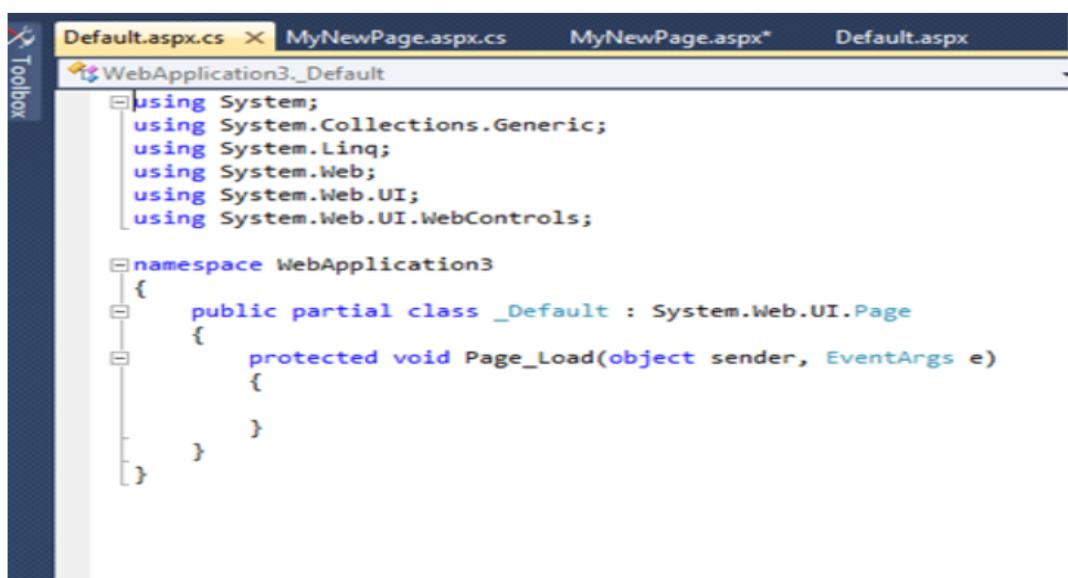
5



בחילון הראשי נראה את תוכן קובץ ה – ASPX. שימוש לב שהוא דומה מאוד ל – HTML . לעומת זאת, ישנו לדף חלק נוסף שנקרא Code-Behind . אל החלק זהה ניתן להגיע על ידי לחיצה בתוך העמוד על הכפתור הימני בעכבר ובחירה באפשרות View Code .



לאחר שנעשה זאת יתקבל המסך הבא:



זהו ה – Code behind , כאן נוכל לכתוב את הפעולות אשר תבוצענה לצד השרת כאשר משתמשים בעמוד זהה .



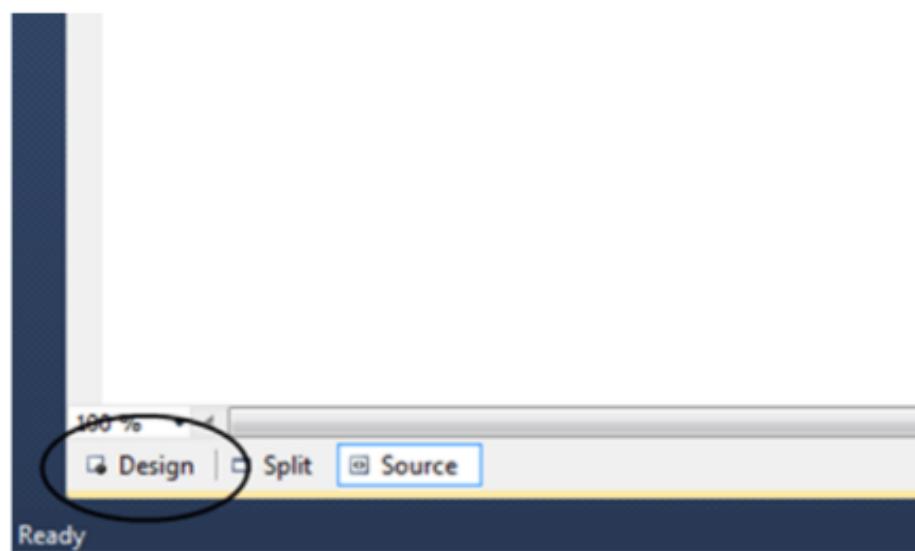
שיתוף

לינק 0



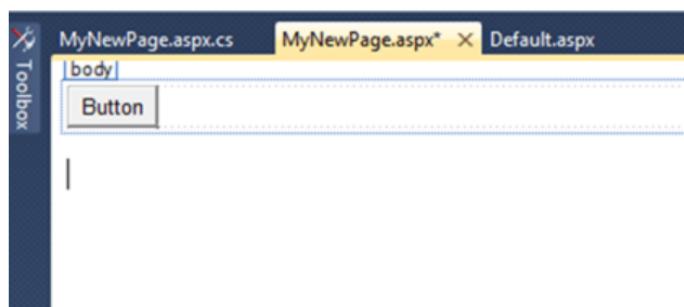
לאחר שיצרנו את העמוד הראשון שלנו נרצה לכתוב קוד עבור התנהוגות הדף בצד השירות. לדוגמה, נרצה שכאשר נלחץ על כפתור – תבוצע פעולה כלשהי בשרת.

על מנת להוסיף כפתור זה יהיה לעבור למצב Design. על מנת לעבור למצב זה נלחץ פעמיים על קובץ ה – ASPX שלנו ולאחר מכן נלחץ על כפתור Design שבתחתית המסך.



כעת, נוסיף כפתור לדף שלנו. נוכל לעשות זאת ע"י בחירה של כפתור מתוך ה – Toolbox

ואגרירתו לדף. ה – Designer אמור להראות לנו:



לאחר מכן, נרצה להוסיף קוד שמתפל באירוע של הקלקה על הכפתור. לאחר הקלקה כפולה על הכפתור ב-Design נראית ב – Code behind את הקוד הבא:

```

public partial class MyFirstPage : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {

    }

    protected void Button1_Click(object sender, EventArgs e)
    {
    }
}

```

בתוך המethode Button1\_Click נוכל לכתוב את הקוד.

בנוסף, נראה ליד המethode הזו עוד Methode ריקה ש – Visual Studio יצר עבורינו בשם Page\_Load. המethode הזו מופעלת כאשר הדף נתען לזכרון השרת.

מכיוון שאין קרייה למethode אחת מהאחרת, כדאי לדעת שהmethode תרוץ אחריו המethode Page\_Load. היא תרוץ Page\_Load.

אחרי כיוון שמethodות המופעלות כתוצאה מארועי פקדים (כמו הקלקה על כפתור) נמצאות אחריו Page\_Load במחזור החיים של הדף.

## מהו מחזור החיים של הדף?

המושג "מחזור החיים של דף" (או "Page LifeCycle" באנגלית) הוא שרשרת האירועים המתרחשים בין הבקשה לדף מהדף, לבין שליחת תוכנו חוזרת לדף ע"י השרת. מחזור החיים של הדף מורכב משלשת השלבים הבאים (קיימיםים עד מספר שלבים, אך אלו הם העיקריים):

1. טיענת הדף לזכרון השרת – Page\_Load.

2. הפעלת אירועים פקדים (methodes כמו Button1\_Click).

3. ייצירת ה – HTML שיוצג לבסוף בדף – Render.

אחר הבקשה הראשונה שלנו וקבלת הדף, אנחנו יכולים לשנות נתונים בעמוד, ללחוץ על כפתור כתובאה מכך תבוצע שליחה של פרטי הדף לשרת ותופעל בשרת מתודה בהתאם לכפתור עליו לחצנו.

מה קורה כאשר לוחצים על הכפתור? הכפתור נלחץ בדף (ולא בשרת) ולעומת זאת השרת "יודע" איזה אירוע קרה באיזה פקד (למשל, שנלחץ כפתור כמו בדוגמה שלפנינו). זהה זה מתאפשר הודות לתהילך שנקרא Postback בו הדף שולח לשרת פרטיים על מצב הפקדים ועל האירוע שגרם לשילחה לשרת, וכך השרת "יודע" איזה אירוע איזו מתודה להפעיל.

בעליה הראשונה של הדף, לא קורה כל אירוע של פקד. אנחנו יכולים גם לדעת האם הפניה לדף היא כתובאה מגלישה לעמוד בפעם הראשונה או כתובאה מאירוע ע"י בדיקת המאפיין IsPostBack. לכן, ב-`Page_Load` ב – `IsPostBack` יראה כך:

```
protected void Page_Load(object sender, EventArgs e)
{
    if (!IsPostBack)
    {
        // בצע פעולות אתחול
    }
    else
    {
        // בצע את שאר הפעולות
    }
}
```

# מדריך ASP.NET – פקדים: פקדי הזנה סטנדרטיים

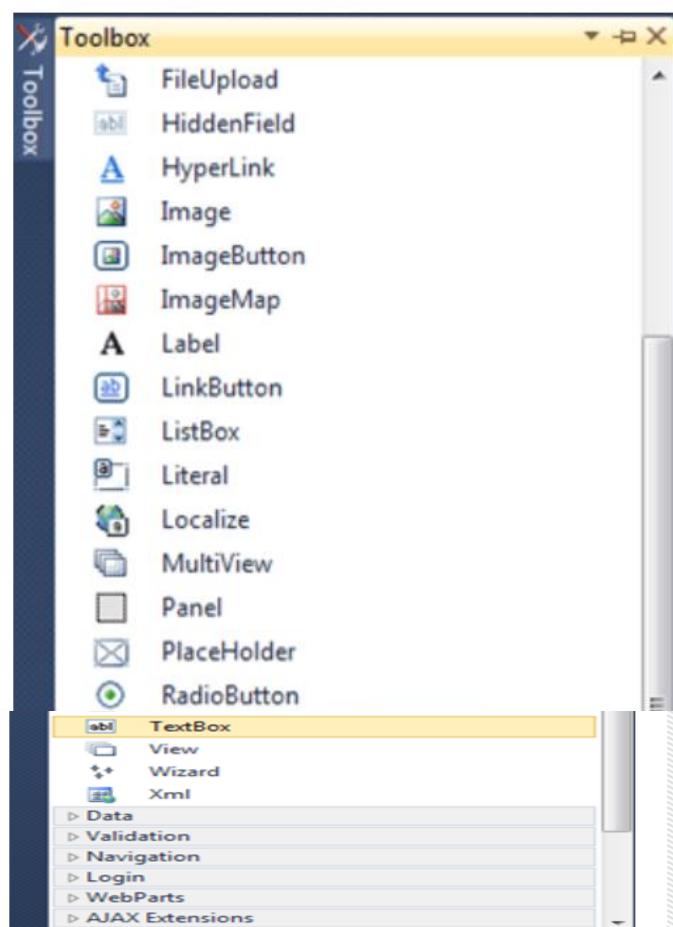
Sela • IdoFlatow



במדריך זה נראה כיצד להוסיף פקדי הזנה סטנדרטיים ואת ה – HTML שנוצר מהם עבור הדף.

## – שדה טקסט שנייתן לכתוב בו TextBox

להוספה פקד TextBox נגרור אותו לדף שלנו מתוך ה-Toolbox:



לאחר אגירת הפקד לדף יכתב הקטע הבא בתוך הדף:

```
<asp:TextBox ID="txtHello" runat="server" Text="Hello ASP.Ne
```

ההגדירה "runat="server" הכרחית לכל פקד שעובדים אליו בצד השרת (Code Behind).

ה – ID משמש כnazha של הפקד. ניתן ב – Code Behind לגשת לפקד כאשר ה – ID הוא שם הפקד. לדוגמה, אם נרצה להגדיר מה שיכתב בתוכו ב – Code Behind נעשה זאת כך:

```
txtHello.Text = "Hello ASP.Net";
```

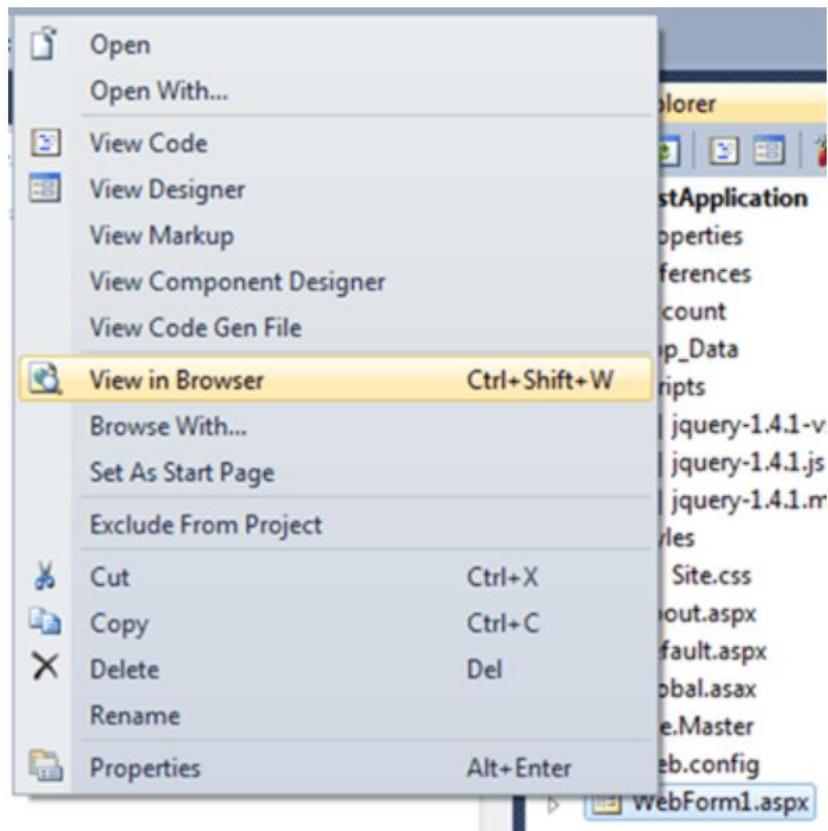
אם נעבור ל – TextBox נראה את ה – TextBox כה:



שיםו לב שם שכתבנו במאפין Text הוא מה שנכתב בתוך תיבת הטקסט בהתחלה. המשמש, כמובן, יכול לשנות זאת.

10

על מנת לראות כיצד הדף נראה בדפדפן, נלחץ על הכפתור הימני של העכבר על הדף שמופיע ב- View in browser – וນבחר ב – Solution Explorer.



אם נתבונן בקטע ה – HTML של תיבת הטקסט, נראה שזו התוצאה:

```
<input name="txtHello" type="text"
       value="Hello ASP.NET" id="Text1" />
```

## – תיבת סימון – CheckBox

הפקד זהה משמש לסימון, נשתמש בו בעיקר כאשר נרצה שימושה שהמשתמש יסמן נתון מסוים ב כן / לא, לדוגמה:

□ אני מעוניין לקבל פרסום

בדומה ל – TextBox – גם אותו ניתן לאגורר מהתוך ה – Toolbox והוא נכתב באופן הבא:

```
<asp:CheckBox ID="Checkbox1"
               Text="אני מעוניין לקבל פרסום" runat="server" />
```

שיםו לב שניתן להגדיר את הכתיבה שהיא ליד ה – Checkbox במאפין Text.

בסוף דבר – ה – HTML שיתקבל הוא :

```
<input id="Checkbox2" type="checkbox" name="Checkbox1" />
<label for="Checkbox1">אני מעוניין לקבל פרסום</label>
```

ניתן לראות מכאן שהוגדרו ב – HTML שני אלמנטים, האחד הוא ה – checkbox עצמו שmagged על-ידי פקד input והשני הוא ה – label (שיםו לב, זהו label של HTML ולא של ASP.NET) המכיל את הכתיבה שלו – CheckBox – ומקשור אליו ע"י המאפיין for שלו.

## לחצן רדיין – RadioButton

11

הפקד הזה מופיע בדרך כלל בקבוצות כאשר המשתמש יכול לסמך אחד מהלחצנים בקבוצה, לדוגמא:

gil 12-16 ● 8-12 ● 4-8 ● 0-4 ●

אפשר לגזרו את ה – RadioButton מתוך ה – Toolbox, וסביר להניח שנרצה בוודאי לגזרו אותו כמה פעמים. בתחום הדף הגדרת הפיקדים תראה כך:

```
<asp:RadioButton ID="RadioButton0" Text="0-4"  
    GroupName="group1" runat="server" />  
<asp:RadioButton ID="RadioButton1" Text="4-8"  
    GroupName="group1" runat="server" />  
<asp:RadioButton ID="RadioButton2" Text="8-12"  
    GroupName="group1" runat="server" />  
<asp:RadioButton ID="RadioButton3" Text="12-16"  
    GroupName="group1" runat="server" />
```

שים לב ש לכל אחד מלוחצני הרדיין יש ID אחר. על מנת שלחיצתו על אחד תבטל את האחרים, علينا לתת להם אותו ערך במאפיין GroupName.

אם נסתכל על ה-HTML שנוצר בעמוד נראה את המבנה הבא:

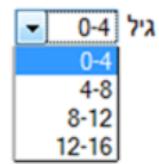
```
<input id="Radio1" type="radio" name="group1"  
    value="RadioButton0" />  
<label for="RadioButton0">0-4</label>  
<input id="Radio2" type="radio" name="group1"  
    value="RadioButton1" />  
<label for="RadioButton1">4-8</label>  
<input id="Radio3" type="radio" name="group1"  
    value="RadioButton2" />  
<label for="RadioButton2">8-12</label>  
<input id="Radio4" type="radio" name="group1"  
    value="RadioButton3" />  
<label for="RadioButton3">12-16</label>
```

שים לב גם כאן ל – label ליד radio, כמו כן שימו לב לכך שהיא שכתבנו ב – .value מגיע ל – ID groupName מה שכתבנו ב – name.

# DropDownList – רשימה נגלחת

12

בדומה ללחצני הרדיו, גם כאן המשתמש צריך לבחור אחד מתוך רשימה, אלא שכאן הרשימה יכולה מסווגת מעט מה שנבחר, ונגלית בלחיצה על חץ.



הגדרת הפקד בדף תראה כך:

```
<asp:DropDownList runat="server" ID="ddlAge">  
</asp:DropDownList>
```

cut הגדרנו רשימה ריקה, על מנת להוסיף לרשימה פריטים נשתמש ב – ListItem עברו כל פריט, כך שהגדרת הפקד תראה כך:

```
<asp:DropDownList runat="server" ID="DropDownList1">  
    <asp:ListItem Text="0-4" Value="0" />  
    <asp:ListItem Text="4-8" Value="1" />  
    <asp:ListItem Text="8-12" Value="2" />  
    <asp:ListItem Text="12-16" Value="3" />  
</asp:DropDownList>
```

כasher ה – Text הוא מה שמוצג למשתמש, וה – Value הוא מה ישלח לשרת במידה והפריט בחר.

תבונן cut ב – HTML שנוצר כתוצאה מה – :DropDownList

```
<select name="ddlAge" id="Select1">  
    <option value="0">0-4</option>  
    <option value="1">4-8</option>  
    <option value="2">8-12</option>  
    <option value="3">12-16</option>  
</select>
```

כאן אנחנו יכולים לראות שאט ה – input מייצגת תגי select (לעומת ה – select) אותו ראיינו בשאר הפקדים) וכל ListItem מייצג ע"י תגי option, כאשר ה – value שלו מקביל ל – Value אותו הגדרנו והוא – Text אותו הגדרנו נמצא בין התג הפתוח והסגור של כל option.

ניתן לקרוא עוד אודות פקדים ב – HTML במאמר [מדריך HTML – טפסים](#).

במדריך זה נראה פקדי תצוגה בסיסיים ב – ASP.NET ו – HTML הנוצר מהם עבור הדף.

בניגוד לפקדי הזרה – בפקדים אלו אין למשתמש יכולת לשנות את תוכן הפקד בדף, כגון שינוי הטקסט או שינוי בחירה בפקד.

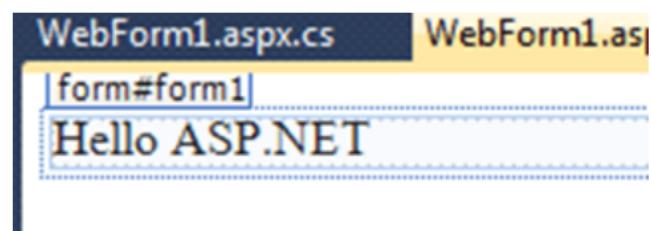
## – תווית – Label

פקד היוצר אזכור עם כתוב שאינו ניתן לעריכה.

בתוך הדף שלנו עליינו Lager מה – Add את פקד ה – Label או לכתוב את הקטע הבא:

```
<asp:Label ID="lblHello" Text="Hello ASP.NET" runat="server">
```

ב – Designer נראה את ה – Label בצורה כזו:



כמו בפקד TextBox, מופיע ה – Text קבוע מה יכתב, אלא שבניגוד לפקד TextBox, למשתמש אין אפשרות לשנות את הכתיבה.

ב – HTML שנוצר עבור הדף נראה את התגית SPAN עבור ה – Label שהגדכנו:

```
<span id="Span1">Hello ASP.NET</span>
```

## – הצגת תמונה – Image

פקד זה מציג תמונה הקיימת באתר שלנו או באתר אחר. בדף שלנו נגדיר אותו לדוגמא כך:

```
<asp:Image runat="server" AlternateText="Webmaster logo" ImageUrl="http://www.webmaster.org.il/images/logo.gif" ToolTip="Webmaster logo" ID="imgLogo" />
```

kpes שלבשה מאפיינים: האחד הוא – `ImageUrl` המגדיר את הכתובת של התמונה, השני – `AlternateText` המגדיר את הכיתוב החלופי לתמונה עבור גולשים עם מוגבלות, מנוע חיפוש וגם מה להציג במידה ומשיבה כלשהי לא מצחחים לטעון את התמונה, השלישי הוא הכיתוב המופיע כשעוביים עם העכבר על התמונה.

ב – יראה הפקד כך:



ה – HTML שיזכר עבור הדף הוא:

```

```

כאשר הפקד יקבל את הtag `img` עם המאפיין `src` עבור הכתובת של התמונה, `title` עבור המאפיין `AlternateText` ו – `alt` עבור `Tooltip`.

## – הצגת טבלה – Table

פקד זה מציג טבלה בדף. על מנת שהטבלה תכיל מידע נוסף נוסף לפקד הזה גם פקד בנים: `Table` ו – `TableRow` אשר יփכו ב – HTML לTAGיות `tr` ו – `td` בהתאם.

הכתיבה בדף ASPX עבור טבלה עם שורה ועמודה אחת תהיה כך:

```
<asp:Table runat="server" ID="myTable">
    <asp:TableRow ID="TblRow1" runat="server">
        <asp:TableCell ID="TblCell1" runat="server"></asp:TableCell>
    </asp:TableRow>
</asp:Table>
```

ניתן גם להוסיף שורות לטבלה ותאים לשורה ב – `Code Behind` באופן הבא:

```
TableRow row = new TableRow();
TableCell cell = new TableCell();
row.Cells.Add(cell);
myTable.Rows.Add(row);
```

ה – HTML המתקיים כתוצאה مما כתבנו ב – ASPX נראה כך:

```
<table id="Table1">
    <tr id="Tr1">
        <td id="Td1"></td>
    </tr>
</table>
```

## – רשימת פריטים BulletedList

פקד זה מציג רשימה כאשר ליד כל פריט מופיע סימן (bullet) כלשהו – אותו סימן ליד כל הפריטים.

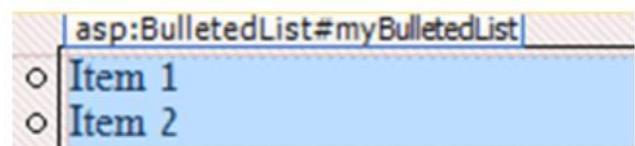
לפקד זהה ישנו פקד בנים, בדומה מאוד לטבלה. הבנים האלה נקראים ListItem, Alla שבניאוד ל – TableRow ו – TableCell אלו אינם פקדים ממש ولكن לא מוסיפים להם "runat="server".

על מנת ליצור רשימה עם שני פריטים נרשום את הקוד הבא ב – ASPX :

```
<asp:BulletedList ID="myBulletedList"
    runat="server" BulletStyle="Circle">
    <asp:ListItem Text="Item 1">
    </asp:ListItem>
    <asp:ListItem Text="Item 2">
    </asp:ListItem>
</asp:BulletedList>
```

המאפיין BulletStyle קובע כיצד יראה הסימן ליד כל פריט, אנחנו בחרנו לצורך ההדגמה ב – Circle המציג עיגול ריק ליד כל פריט, אך ניתן לשנות את מאפיין זה להצגת ספרות, אותיות, ספרות רומיות וכו' מנימן נוספים אחרים. לכל ListItem ישנו מאפיין טקסט הקובע מה יוצג ליד הפריט.

ב – Designer נראה את הרשימה באופן הבא:



ה – HTML שנוצר מתוך מה שכתבנו ב – ASPX הוא:

```
<ul id="Ul1" style="list-style-type: circle;">
    <li>Item 1</li>
    <li>Item 2</li>
</ul>
```

פקד Calendar מציג לוח שנה על המסך שמננו ניתן לבחור תאריך. לוח השנה יראה באופן הבא:

אוקטובר 2009						
שבת יומן יום הדיום ג'יום ב'יום א'						
<u>27</u>	<u>28</u>	<u>29</u>	<u>30</u>	<u>1</u>	<u>2</u>	<u>3</u>
<u>4</u>	<u>5</u>	<u>6</u>	<u>7</u>	<u>8</u>	<u>9</u>	<u>10</u>
<u>11</u>	<u>12</u>	<u>13</u>	<u>14</u>	<u>15</u>	<u>16</u>	<u>17</u>
<u>18</u>	<u>19</u>	<u>20</u>	<u>21</u>	<u>22</u>	<u>23</u>	<u>24</u>
<u>25</u>	<u>26</u>	<u>27</u>	<u>28</u>	<u>29</u>	<u>30</u>	<u>31</u>
<u>1</u>	<u>2</u>	<u>3</u>	<u>4</u>	<u>5</u>	<u>6</u>	<u>7</u>

כל יום בלוח השנה הוא קישור שכשר הוא נלחץ מתעדכן התאריך הנבחר בשורת.

נתבונן כתה במה שנכתב ב – ASPX על מנת ליצור את לוח השנה:

```
<asp:Calendar runat="server" ID="myCalendar"
    SelectedDate="2011-01-17" />
```

התאריך הנבחר מוגדר במאפיין SelectedDate, כן הגדכנו שברירת המחדל של הפקד יהיה התאריך 17 בינואר 2011.

עבור הכתיבה הקצר זהה מיוצר ה – HTML הבא:

```
<table id="Table2" cellspacing="0" cellpadding="2"
    title="Calendar" style="border-width: 1px;
    border-style: solid; border-collapse: collapse;">
    <tr>
        <td colspan="7" style="background-color: silver;">
            <table cellspacing="0" style="width: 100%;
            border-collapse: collapse;">
                <tr>
                    <td style="width: 15%;">
                        <a title="Go to the previous month"
                            href="javascript:_doPostBack('myCalendar','V398')"
                        >
                            &lt;&lt;/a>
                    </td>
                    <td align="center" style="width: 70%;>
                        2011

```

```

<td align="center" style="width: 14%;">
    <a href="javascript:_doPostBack('myCalendar','4017')"
        style="color: Black" title="31">דצמבר 31</a>
</td>
<td align="center" style="width: 14%;">
    <a href="javascript:_doPostBack('myCalendar','4018')"
        style="color: Black" title="01">ינואר 1</a>
</td>
</tr>
***<br/>
</table>

```

\*\*\* – ה – HTML ארוך لكن איננו מובא במלואו – קיימות עוד שורות עבור שאר השבועות, לצורך הדוגמה מוצגת רק השורה הראשונה.

ניתן לראות שלוח השנה מוצג בצורה טבלה, כאשר בשורה הראשונה מוגדר החודש, בשורה לאחר מכן ימי השבוע, ובשורות לאחר מכן התאריכים עצמם, קישוריםיהם. לחיצה על כל קישור תבצע Postback לשרת ותעדכן שם את התאריך.

כמו כן, ניתן לראות בשורה (tr) הראשונה בטבלה, שני קישוריםים עם הסימנים <-> (המייצגים ע"י &gt;& lt;) – ב – HTML בהתאם – אלו מפנים לחודש הבא או הקודם. לחיצה עליהם תביא ליצירת HTML התואם את אותו חודש.

# מדריך ASP.NET – פקדים מתקדמים: FileUpload

Sela • IdoFlatow



פקוד FileUpload מאפשר למשתמש לטעון קובץ לשרת (פעולות Upload). על מנת להשתמש בו – נכתב בדף ה – ASPX את הכתיבה הבא:

```
<asp:FileUpload runat="server" ID="myUpload" />
```

כדי להשתמש בקובץ שהובא לשרת, נוסיף כפטור לשמיירה:

```
<asp:Button Text="Save File" runat="server"
ID="btnSave" onclick="btnSave_Click" />
```

במתודה btnSave\_Click נכתב את הפקוד הבא:

```
protected void btnSave_Click(object sender, EventArgs e)
{
    myUpload.SaveAs(Path.Combine("c:\\temp\\", myUpload.FileName))
}
```

שורט הפקוד שכתבנו תגרום לכך שהקובץ ישמר בשרת, בתיקיה temp\C במכונה בה נמצא האפליקציה שלנו.

שיםו לב, הרצת הפקוד זהה עלולה להיתקל בשגיאות במידה ואין הרשות כתיבה על המיקום בו אתם מבקשים לשמור את הקובץ.

נתבונן כעת ב – HTML שנוצר עבור הפקד זהה:

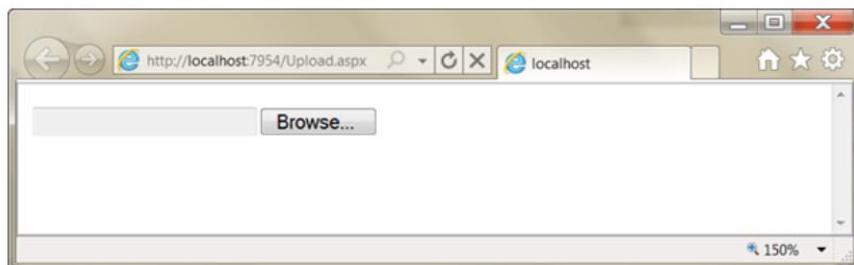
בהגדרת הטופס (form tag) נוספה התכונה enctype עם הערך multipart/form-data. זה נדרש על מנת לאפשר שליחת קבצים לשרת כפי שעושה הפקד.

```
<form method="post" action="WebForm1.aspx"
id="form2" enctype="multipart/form-data">
```

עבור הפקד עצמו יוצר ה – HTML הבא :

```
<input type="file" name="myUpload" id="File1" />
```

זהו תג input מסוג file, הדף נוצר מציר עבורי תיבת טקסט ולחצן לידה עם הכתיבה Browse, ושלוחצים עליו מגעים למערכת הקבצים המקומיות שלנו:



## מדריך ASP.NET – פקדים מתקדמים: ImageMap

Sela • IdoFlatow



פקד ImageMap מאפשר יצירת תמונה המכילה קישורים, כאשר כל אזור בתמונה יכול להיות מוגדר כקישור אחר.

נניח שיש לנו תמונה הנראית כך:



נרצה שלחיצה על כל מלבן טוביל למקום אחר (דף "מי אנחנו", דף "דרושים" ודף "מוצרים"), לצורך כך علينا להגדיר את התמונה כ"מפה" ולהגדיר אזוריים על המפה הזו. לצורך העניין נכתוב כך בדף ה-ASPX:

```

<asp:ImageMap runat="server" ID="myImageMap"
    ImageUrl="images/mapSample.png">
    <asp:RectangleHotSpot NavigateUrl="Pages/Products"
        AlternateText="מוצרים" Right="156" Bottom="110" />
    <asp:RectangleHotSpot NavigateUrl="Pages/Jobs"
        AlternateText="דרושים" Right="312" Bottom="110"
        Left="156" />
    <asp:RectangleHotSpot NavigateUrl="Pages/AboutUs"
        AlternateText="מי אנחנו" Right="468" Left="312"
        Bottom="110" />
</asp:ImageMap>

```

המאפיין imageUrl הוא הכתובת לתמונה (בדומה לפיקד Image). לפיקד ישנו פקד בנים הנקראים HotSpots כאשר לכל אחד שני מאפיינים עיקריים:

- – הכיתוב המופיע כ – tooltip במידה והעכבר עובר על האזור.
- – הכתובת לדף אשר לוחצים על העכבר באזורי.

## איך מוגדר האזור?

האזור מוגדר לפי סוג ה – HotSpot , במקרה שלנו מדובר באזור מלבי (ישנו גם מעגל ומצולע הנקראים PolygonHotSpot ו – CircleHotSpot בהתאם). המלבן מוגדר ע"י 4 קואורדינטות : Left, Right, Top ו – Bottom אשר יחסית למיקום התמונה בדף.

נתבונן ב – HTML הנוצר עבור ה – ImageMap :

```


<map name="ImageMapMainContent_myImageMap"
    id="ImageMapMainContent_myImageMap">
    <area shape="rect" coords="0,0,156,110"
        href="Pages/Products" title="מוצרים" alt="מוצרים" />
    <area shape="rect" coords="156,0,312,110"
        href="Pages/Jobs" title="דרושים" alt="דרושים" />
    <area shape="rect" coords="312,0,468,110"
        href="Pages/AboutUs" title="מי אנחנו" alt="מי אנחנו" />
</map>

```

נוכל לראות כאן את תג של תמונה (img) אשר נוסף לו מאפיין שנקרא usemap – זה גורם לו להציבו על תג אחר בשם map שנוצר אף הוא עבור הפיקד ImageMap והוא מכיל את האזורי.

עבור כל HotSpot נוצר תג בשם area עם מאפייני alt – i shape, cords, href, title

# מדריך ASP.NET – ניהול State בצד שרת

Sela • IdoFlatow



בבואהנו לפתח מערכות WEB, נרצה בוודאי לשמר נתונים בין הדפים השונים ולעתים גם נרצה לשמר נתונים עבור כל המשתמשים במערכת.

לצורך העניין קיימים ב-ASP.NET מספר כלים המאפשרים לנו לנהל מצבים (State בצד השרת). במדריך זה נפרט את הכלים השונים לניהול State.

## Session

נستخدم ב-Session במידה ונרצה לשמר נתונים עבור אותו משתמש גם כאשר הוא עבר בין דפים.

ה-Session תפקידו לשמר נתונים עבור המשתמש בכל זמן שהוא נמצא במערכת (כלומר ה-Session שלו פעיל).

הקוד לשמירת מידע ב-Session הוא כר:

```
Session["IDNumber"] = <User ID number>;
```

נתונים אלו נשמרים עבור המשתמש כל עוד הוא מחובר למערכת שלנו. הנתונים נמחקים עבור המשתמש אם לא ביצע כל פעולה במערכת במשך זמן מסוים שנקבע Session timeout, שערך ברירת המחדל שלו הוא 20 דקות.

## Application

בדומה ל-Session, גם כאן נשמרים נתונים ללא תלות בדף בו נמצא המשתמש, אלא שכאן הנתונים גם אינם תלויים במשתמש עצמו ונשמרים עבור כל המשתמשים במערכת.

למשל – אם נרצה לשמר את הזמן בו התחבר המשתמש האחרון למערכת, נכנו ל – Code Behind של קובץ ה-.Global.asax, שם נמצא מетодה שנקראת Session\_Start המופעלת כאשר משתמש מתחבר למערכת (כasher נוצר ה-Session) וונכתוב בה את הקוד הבא:

```
protected void Session_Start(object sender, EventArgs e)
{
    Application["LastUserEntered"] = DateTime.Now;
}
```

הנתון הזה מתייחס לכל המערכת לא קשר מי המשתמש. הנתונים הקשורים לכל המערכת מתנקים זמן מה לאחר שאחרון המשתמשים עוזב את המערכת.

## מדריך ASP.NET – ניהול State בצד לקוח

Sela • IdoFlatow



לינק

ניהול State בצד לקוח הוא שם כולל לכל הפעולות שאנו עושים כדי לשמר נתונים על מחשב (ודףפן) הלקוח.

במדריך זה נסקור את הכלים שברשותוינו על מנת לשמר נתונים בדףן.

## ViewState

אם נתבונן ב – HTML הנוצר עבור הדף ASPX נראה את הדבר הבא:

```
<input type="hidden" name="__VIEWSTATE" id="__VIEWSTATE"
value="/wEPDwUKLTE0ODM1NDQ20Q9kFgJmD2QWA
gIDD2QWAgIFD2QWAgIBDxQrAAJkEBYAFgAWAGRkC
RKuJ6UkvoiqWwXj4Zp9pCeIFA8mZv3XY3fwOJNCr4k=" />
```

זהו שדה מיוחד שנוצר ע"י ה-ASPX שומר בתוכו נתונים על הדף, על הפקדים שבתוכו, ומידע לנווה שהמפתח יכול לשמר לעמוד.

לכל פקד ישנו מאפיין בוליани שנקרא EnableViewState. אם הערך שלו יהיה True (ערך ברירת המחדל של רוב הפקדים) אז כל שינוי שייה בפקד אצלינו בדף, ישמור בשרת ונקבל אותו גם ב – Postback הבא שנעשה. לדוגמה: אם בפקד תיבת טקסט (TextBox) נשנה בקוד Postbacks את מאפיין BackColor של תיבת הטקסט ל – Blue, הוא ישאר במצב הזה גם ב – הבאים, גם אם לא נשנה שוב בקוד את המאפיין הזה.

## ואיך זה עובד?

יחד עם הדף יורדים נתונים על מצב הפקדים בשדה נסתר שנקרא \_\_VIEWSTATE, שם נשמרים כל הנתונים אודות הפקדים. תוכן השדה הזה יכול או קטן בהתאם במספר הפקדים בדף בהם המאפיין EnableViewState הוא True.

ב – ViewState ניתן לשמור גם נתונים שאינם קשורים לפקדים. לדוגמה, אם נרצה לשמר את הפעם האחרונה בה نطען הדף ללא Postback נוכל לנכון את הקוד הבא:

```
protected void Page_Load(object sender, EventArgs e)
{
    if (!IsPostBack)
    {
        ViewState["LastInitialLoad"] = DateTime.Now;
    }
}
```

ה – ViewState נשמר כאשר אנחנו באוטו הדף, ובמידה ועובריםמן הדף לדף אחר, ה – ViewState מתנתקה.

## Hiddden Field

שדות נסתרים הם כל' המאפשר לנו לשמר מידע שלא נראה למשתמש. המידע הזה נשלח לשרת בכל Submit כיון שהוא נשמר בשדות. דוגמא טובה לכך אנו רואים בחלק המדובר על ViewState שכן הוא נשמר בשדה צזה. ניתן כמובן גם ליצור שדות נוספים ולשמור בהם נתונים שונים.

בצד השרת קיים פקד המיציר Hidden field, שנקרא Hidden field עבורי נכתב לדוגמה בדף ה – ASPX את הקוד הבא:

```
<asp:HiddenField runat="server" ID="myHiddenField"
Value="My Hidden Value" />
```

זהו ה – HTML הנוצר ממנו:

```
<input type="hidden" name="myHiddenField"
id="Hidden1" value="My Hidden Value" />
```

ניתן לשמור מידע בתוך הקישורים ע"י הוספת פרמטרים. הפרמטרים האלו ישלו לשרת חדשות לכל דבר (כמו Hidden fields או תיבות טקסט)

ובשרת ניתן יהיה להשתמש בנתון שנשלח. לדוגמה: בקישור <http://www.webmaster.org.il/article.asp?id=646> שנו פרמטר בשם id עם הערך 646. כאן השרת יודע לשЛОפ מדריך מסוים באתר לפי הפרמטר.

## Cookie

העוגיות (Cookies) הם קבצים קטנים הנמצאים במחשב הלקוח ותפקידן לשמר מידע, לעיתים גם לאחר שהלקוח סגור את הדף. ה – Cookies נשלחים לשרת כל פעם כחלק מה – Request ולעיתים השרת עושה בהן שימוש. לדוגמה – אם נרצה לשמר את תאריך ההתחברות האחרון של המשתמש יוכל לעשות זאת כך:

```
Response.Cookies["LastLoginDate"].Value = DateTime.Now.ToString();
```

אם נרצה לעשות שימוש בנתון יוכל לפנות אליו כך:

```
string strLastLoginDate = Request.Cookies["LastLoginDate"].Value;
```

Value

## מדריך ASP.NET – הציג נתונים בדף

Sela • IdoFlatow



שיתוף 0 לijk 0

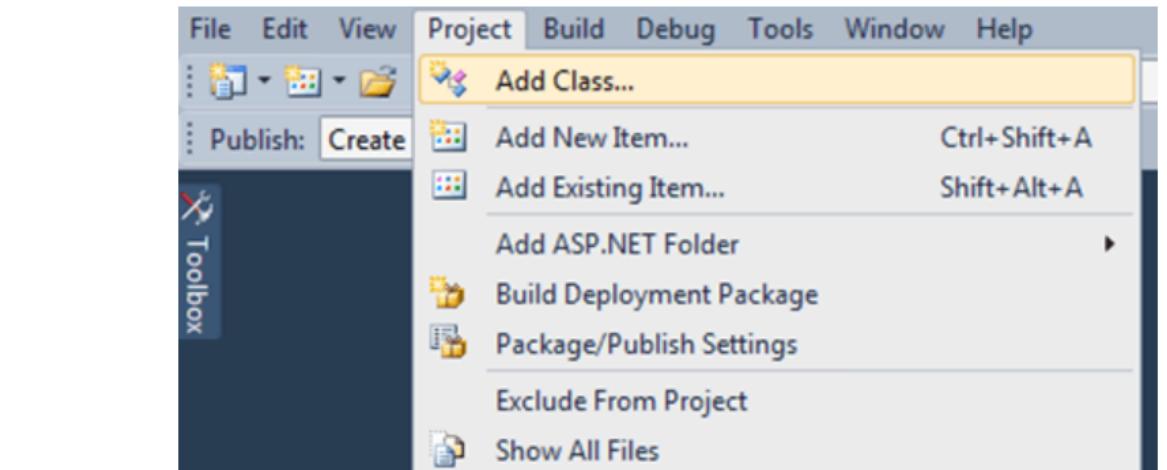
במדריך זה נראה איך ניתן להציג נתונים ב – GridView. אנו ניצור נתונים דמה ונחבר אותם ל – GridView.

### הכנה

נוסף דף ASPX למערכת שלנו. לדף נוסף פקד מסווג GridView ונקרא לו gvPeople. הכיתוב עבורי יהיה כך:

```
<asp:GridView runat="server" ID="gvPeople" />
```

נוסף מחלקה חדשה בשם Person לפרויקט שלנו (בחירה ב-Project Add Class מftpriet



נרשום את מבנה המחלקה באופן הבא:

```
public class Person
{
    public string FirstName { get; set; }
    public string LastName { get; set; }
    public string FullName
    {
        get
        {
            return string.Format("{0} {1}", FirstName, LastName);
        }
    }
    public DateTime BirthDate { get; set; }
}
```

```

private List<Person> CreatePeopleList()
{
    List<Person> people = new List<Person>();
    for (int i = 0; i < 5; i++)
    {
        people.Add(new Person())
        {
            FirstName = string.Format("First{0}", i),
            LastName = string.Format("Last{0}", i),
            BirthDate = new DateTime(1980, 10, i + 1),
        });
    }
    return people;
}

```

שיםו לב שב – `BirthDate` אנחנו משתמשים ב – `i+1` זאת כדי להמנע מכך שיוצג תאריך עם ים בחודש 0 שכן החודש תמיד מתחילה מ-1.

כעת, ב – `Page_Load` נחבר את רשימת ה – `GridView` `People` באופן הבא:

```

protected void Page_Load(object sender, EventArgs e)
{
    List<Person> people = CreatePeopleList();
    gvPeople.DataSource = people;
    gvPeople.DataBind();
}

```

את הרשימה שזרה מ – `CreatePeopleList` אנחנו מחברים ל – `gvPeople` דרך המאפיין `DataSource`. לאחר מכן אנחנו קוראים ל – `DataBind`. `gvPeople` מתקבל במסגר הטבלה הבאה:

ID	FirstName	LastName	FullName	BirthDate
0	First0	Last0	First0 Last0	01/10/1980 00:00:00
0	First1	Last1	First1 Last1	02/10/1980 00:00:00
0	First2	Last2	First2 Last2	03/10/1980 00:00:00
0	First3	Last3	First3 Last3	04/10/1980 00:00:00
0	First4	Last4	First4 Last4	05/10/1980 00:00:00

 The browser address bar shows 'http://localhost/Demos/UsingGridView.aspx'."/>

זהו למעשה ייצוג של הנתונים שייצרנו ב – `CreatePeopleList` בטבלה.

ונכל לשנות את הכיתוב בעמודות ע"י:

א. השמת ערך False במאפיין AutoGenerateColumns של פקד ה-GridView.

ב. עריכת העמודות. הדוגמה הבאה מציגה GridView עם עמודות מוגדרות:

```
<asp:GridView runat="server" ID="gvPeople"
    AutoGenerateColumns="false">
    <Columns>
        <asp:BoundField HeaderText="First Name"
            DataField="FirstName" />
        <asp:BoundField HeaderText="Last Name"
            DataField="LastName" />
        <asp:BoundField HeaderText="Full Name"
            DataField="FullName" />
        <asp:BoundField HeaderText="Birth Date"
            DataField="BirthDate" />
    </Columns>
</asp:GridView>
```

עבור GridView כזה יתקבל המסר הבא (שיםו לב לשוני בריווח המילים בכותרות העמודות):

First Name	Last Name	Full Name	Birth Date
First0	Last0	First0 Last0	01/10/1980 00:00:00
First1	Last1	First1 Last1	02/10/1980 00:00:00
First2	Last2	First2 Last2	03/10/1980 00:00:00
First3	Last3	First3 Last3	04/10/1980 00:00:00
First4	Last4	First4 Last4	05/10/1980 00:00:00

אם נרצה ליצג את הנתונים בצורה אחרת נוכל להשתמש בפקדים נוספים כגון Repeater, FormView, DataList וכו'.

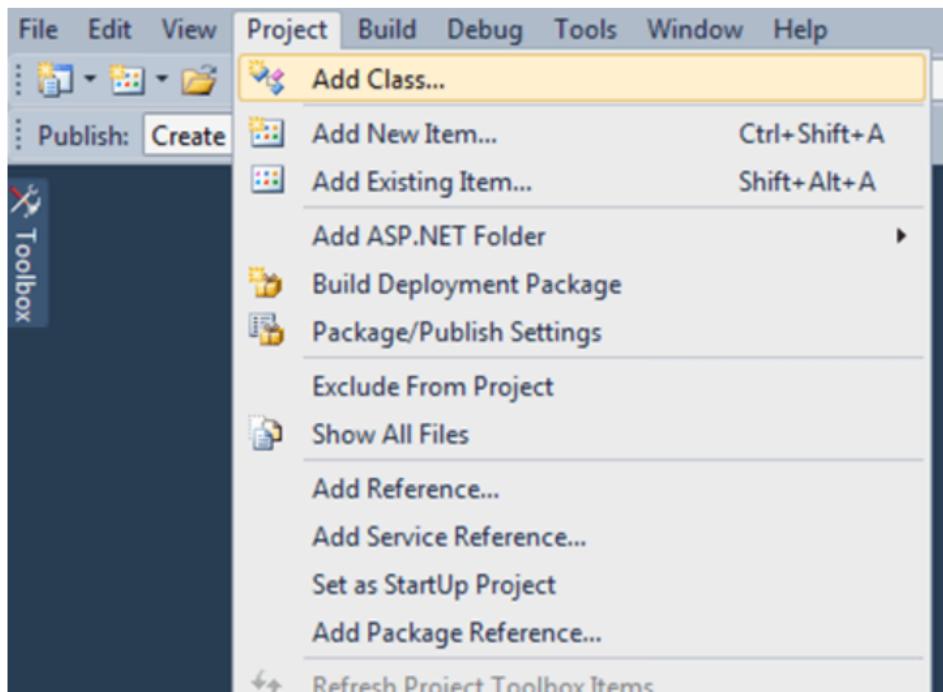
# מדריך ASP.NET – עבודה עם נתונים: שימוש ב- ObjectDataSource

Sela • IdoFlatow



במדריך זה נראה כיצד משתמשים ב- ObjectDataSource על מנת להציג נתונים באמצעות פקד GridView ובאמצעות פקד Repeater.

נוסף לפרויקט שלנו מחלקה חדשה בשם Person (בחירה ב-Project Add Class מתפרקת בבחירה Person).



נרשום את קוד המחלקה הבא:

```
public class Person
{
    public string FirstName { get; set; }
    public string LastName { get; set; }
    public string FullName
    {
        get
        {
            return string.Format("{0} {1}", FirstName, LastName);
        }
    }
    public DateTime BirthDate { get; set; }
}
```

נוסף עוד מחלקה בשם PersonLogic שתכיל את הפעולות שאנו רוצים לבצע. נרשום את קוד המחלקה הבא:

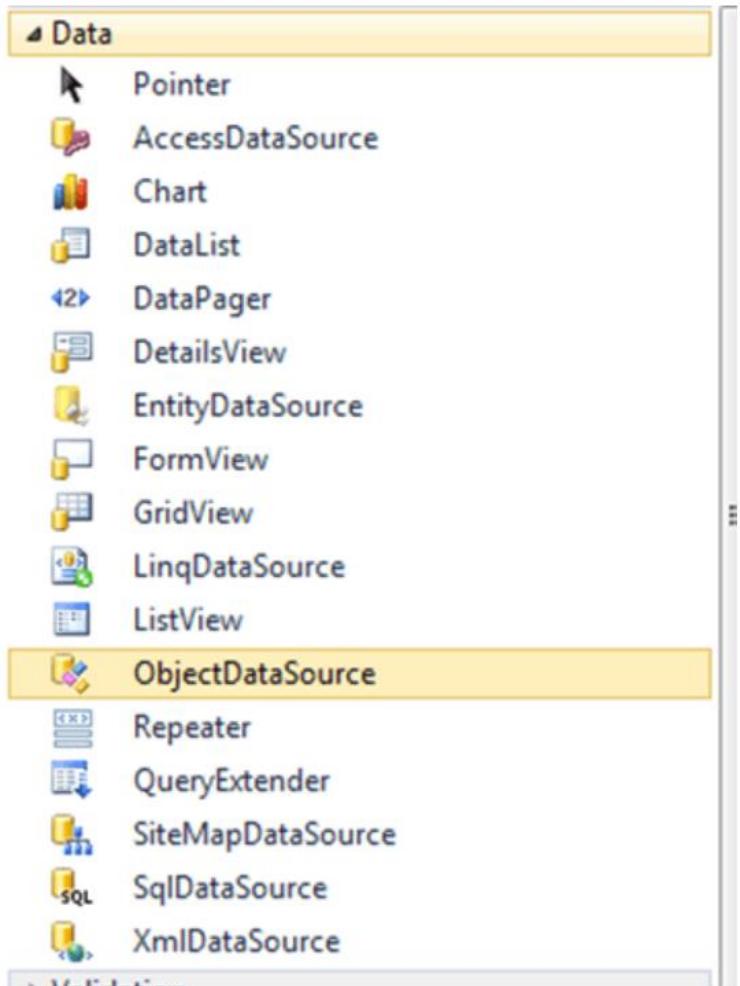
```

public class PersonLogic
{
    public static List<Person> CreatePeopleList()
    {
        List<Person> people = new List<Person>();
        for (int i = 0; i < 5; i++)
        {
            people.Add(new Person()
            {
                FirstName = string.Format("First{0}", i),
                LastName = string.Format("Last{0}", i),
                BirthDate = new DateTime(1980, 10, i + 1),
            });
        }
        return people;
    }
}

```

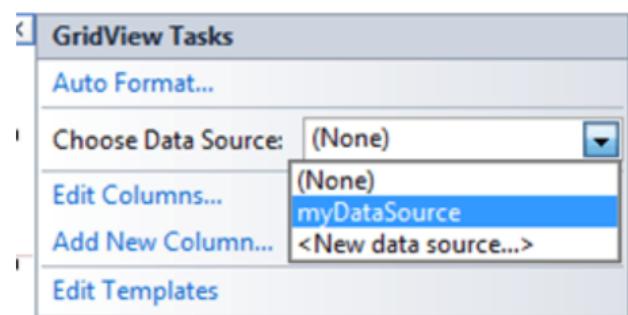
נוסף דף ASPX למערכת שלנו ומתחילה – **Toolbox**, תחת הקטגוריה **Data**, נמצאת:

ונגרור אותו לתוך הדף:



נשנה ל – ID את ה – myDataSource – .myDataSource. לאחר מכן נמצאת פקדת – GridView – ב – Toolbox וngror אותו לדף. נלחץ על החץ המסומן בעיגול:

בתיבה שנפתחה – נסמן את myDataSource כמקור הנתונים של ה – GridView:



נסמן את פקד ה-ObjectDataSource, נלחץ עליו עם הכפטור ימני של העבר ונבחר בו – Choose Data Source. בחלון ה-Properties נגדיר ערכים לשני מאפיינים חשובים: TypeName ו-SelectMethod.

1. TypeName – זהו שם המחלקה ממנה אנחנו מקבלים את הנתונים (זהו איננו סוג הנתון עצמו), כולל ה-Namespace המלא, לדוגמה `Namespace.DataDemo.PersonLogic`.

(במרבית המקרים Namespace של המחלקה יהיה על-פ' שם פרויקט ה-Web שנוצר).

2. SelectMethod – זהו שם המתודה בתוך המחלקה מהמאפיין הקודם, המחזירה את הנתונים. המתודה צריכה להיות `public`, לא-גנרטית (לא `<%>` בשמה) ולא פרמטרים, לדוגמה: `CreatePeopleList`.

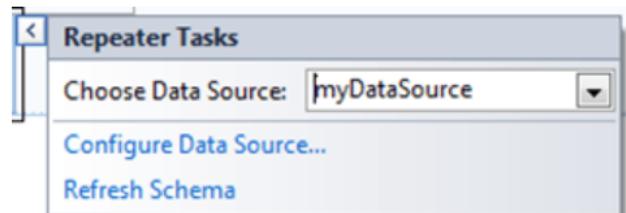
כעת, כל שנותר לנו הוא לראות את התוצאה בדף – נסמן את קובץ ASPX-ה – Solution Explorer, נלחץ על הכפטור ימני ונבחר ב-View in Browser:

FirstName	LastName	FullName	BirthDate
First0	Last0	First0 Last0	01/10/1980 00:00:00
First1	Last1	First1 Last1	02/10/1980 00:00:00
First2	Last2	First2 Last2	03/10/1980 00:00:00
First3	Last3	First3 Last3	04/10/1980 00:00:00
First4	Last4	First4 Last4	05/10/1980 00:00:00

# שימוש ב-Repeater

31

נוסף לדף שלנו פקק Repeater ונבחר עליו את ה – DataSource באותה הצורה:

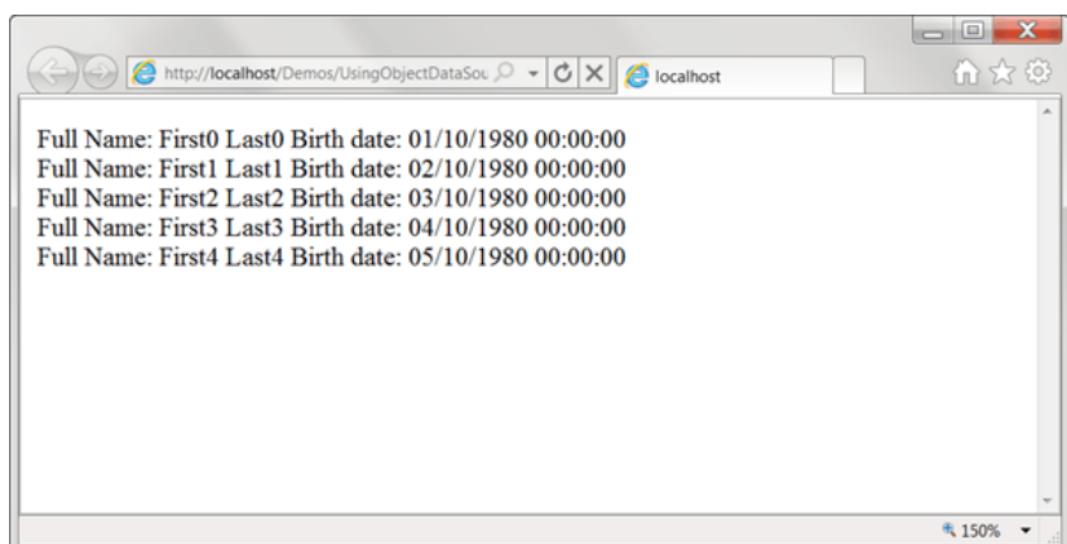


כעת علينا לעורר את התבנית של ה – Repeater, לצורך כך נחזיר ל – Source tab ונקתוב את הכתוב הבא:

```
<asp:Repeater ID="Repeater1" runat="server"
    DataSourceID="myDataSource">
    <ItemTemplate>
        <asp:Label ID="Label1"
            Text="Full Name:" runat="server" />
        <asp:Label ID="Label2" runat="server"
            Text='<%# DataBinder.Eval(Container.DataItem, "FullName")%>'>
        <asp:Label ID="Label3"
            Text="Birth date: " runat="server" />
        <asp:Label ID="Label4" runat="server"
            Text='<%# DataBinder.Eval(Container.DataItem, "BirthDate")%>'>
        <br />
    </ItemTemplate>
'
```

שים לב לאוטם Labels המציגים את הנתונים, התחברו שלהם הוא קבוע כאשר מה שמשתנה הוא מה שכתוב במרקאות. אלו הם שמות המאפיינים של ה – class אליו אנחנו עושים ..DataBind

כעת, נבדוק שוב את התוצאה בדף. תוצאה מה – Repeater, התקבלה התוצאה זו:





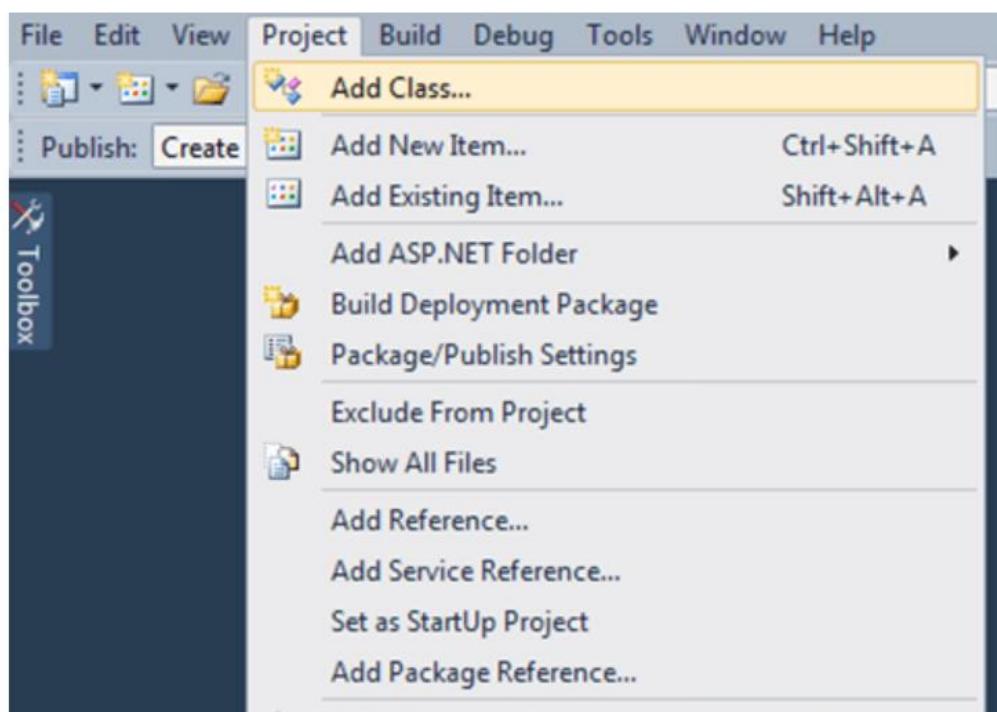
שתף

לijk 0

במדריך זה נראה איך עורכים נתונים בפקד GridView.

## שלב 1 – הכנה

נוסף מחלקה חדשה בשם Person לפרויקט שלנו (בחירה ב-**Add Class** מתפריט :



נרשום את מבנה המחלקה הבא:

```
public class Person
{
    public int ID { get; set; }
    public string FirstName { get; set; }
    public string LastName { get; set; }
    public string FullName
    {
        get
        {
            return string.Format("{0} {1}", FirstName, LastName);
        }
    }
    public DateTime BirthDate { get; set; }
}
```

נוסף עוד מחלקה בשם PersonLogic שתכיל את הפעולות שאנו רוצים לבצע. אנו נרצה להוסיף למחלקה זו שתי פעולות:

2. פעולה Update. פועלה שתעדכן פרטיים של אדם.

את רשימת האנשים נוכל לאחזר מבסיס הנתונים ואת העדכנים לבצע מול בסיס הנתונים, אך לשם הדגמה אנחנו נשמר אותם ב-Session, שכןኖסיף גם מופיע למחלקה החדשה שיחזיר רשימה של אנשים.

הקוד של המחלקה PersonLogic יראה כך:

```
public class PersonLogic
{
    private List<Person> PersonList
    {
        get
        {
            return HttpContext.Current.Session["People"]
                as List<Person>;
        }
        set
        {
            HttpContext.Current.Session["People"] = value;
        }
    }

    public List<Person> SelectPeople()
    {
        if (PersonList == null)
        {
            List<Person> persons = new List<Person>();
            for (int i = 0; i < 5; i++)
            {
                persons.Add(new Person()
                {
                    ID = i,
                    FirstName = string.Format("First{0}", i),
                    LastName = string.Format("Last{0}", i),
                    BirthDate = new DateTime(1980, 10, i + 1),
                });
            }
            PersonList = persons;
        }

        return PersonList;
    }
}
```

```

public void UpdatePerson(int id, string FirstName,
    string LastName, DateTime BirthDate)
{
    if (PersonList == null)
        return;
    Person personToUpdate = PersonList.Where(
        person => person.ID == id).FirstOrDefault();
    if (personToUpdate == null)
        return;
    personToUpdate.FirstName = FirstName;
    personToUpdate.LastName = LastName;
    personToUpdate.BirthDate = BirthDate;

}
}

```

נוסף דף ASPX למערכת שלנו ונוסף לדף פקד ObjectDataSource. נעדכן את הפקד שיצביע על המתודות שיצרנו במחלקה PersonLogic (שיםו לב שהשדה TypeName מכיל את שם ה- Namespace וייתכן שתצטרכו לתקן אותו בהתאם ל- Namespace שליכם):

```

<asp:ObjectDataSource runat="server" ID="myDataSource"
    TypeName="DataDemo.PersonLogic"
    SelectMethod="SelectPeople"
    UpdateMethod="UpdatePerson"></asp:ObjectDataSource>

```

לדענו אף גם פקח מסוג GridView ונקרא לו gvPeople. נקבע מתחם ה – GridView על ה – ObjectDataSource ונגידר אליו עמודות יוצאו ב-GridView. זהו הכתיבה המלא של ה – GridView

35

```
<asp:GridView ID="GridView1" EnableViewState=true  
    runat="server" DataSourceID="myDataSource"  
    AutoGenerateColumns="False">  
    <Columns>  
        <asp:BoundField DataField="ID" HeaderText="ID"/>  
        <asp:BoundField DataField="FirstName" HeaderText="FirstNa  
        <asp:BoundField DataField="LastName" HeaderText="LastNa  
        <asp:BoundField DataField="FullName" HeaderText="FullNa  
            ReadOnly="True"/>  
        <asp:BoundField DataField="BirthDate" HeaderText="BirthDa  
    </Columns>  
</asp:GridView>
```

בשלב זה יצרנו פקח GridView ל视ichage בלבד אשר אינו מאפשר עריכה של הנתונים שבתוכו.  
בשלב הבא נאפשר עריכה של תוכן הפקח.

## שלב 2 – הוספה עמודת פקודה Grid – ל (CommandColumn)

נוסף בתוך הגדרות ה – GridView את העמודה הבאה:

```
<asp:CommandField ButtonType="Link" EditText="Edit"
    HeaderText="Edit" ShowEditButton="true" />
```

כך שהכיתוב של ה – Grid יראה כך:

```
<asp:GridView ID="GridView2" EnableViewState=true runat="server"
    DataSourceID="myDataSource" AutoGenerateColumns="False">
    <Columns>
        <asp:CommandField ButtonType="Link" EditText="Edit"
            HeaderText="Edit" ShowEditButton="true" />
        <asp:BoundField DataField="ID" HeaderText="ID" />
        <asp:BoundField DataField="FirstName" HeaderText="FirstName" />
        <asp:BoundField DataField="LastName" HeaderText="LastName" />
        <asp:BoundField DataField="FullName" HeaderText="FullName" />
        <asp:BoundField DataField="BirthDate" HeaderText="BirthDate" />
    </Columns>
</asp:GridView>
```

Edit	ID	FirstName	LastName	FullName	BirthDate
<a href="#">Edit</a>	0	First0	Last0	First0 Last0	01/10/1980 00:00:00
<a href="#">Edit</a>	1	First1	Last1	First1 Last1	02/10/1980 00:00:00
<a href="#">Edit</a>	2	First2	Last2	First2 Last2	03/10/1980 00:00:00
<a href="#">Edit</a>	3	First3	Last3	First3 Last3	04/10/1980 00:00:00
<a href="#">Edit</a>	4	First4	Last4	First4 Last4	05/10/1980 00:00:00

נראה שנוספו לנו קישורים עם המלה "Edit". אם נלחץ על אחד מהם (למשל על השורה השלישי) נקבל את המסר הבא:

Edit	ID	FirstName	LastName	FullName	BirthDate
<a href="#">Edit</a>	0	First0	Last0	First0 Last0	01/10/1980 00:00:00
<a href="#">Edit</a>	1	First1	Last1	First1 Last1	02/10/1980 00:00:00
<a href="#">Update</a> <a href="#">Cancel</a>	2	First2	Last2	First2 Last2	03/10/1980 00:00:00
<a href="#">Edit</a>	3	First3	Last3	First3 Last3	04/10/1980 00:00:00
<a href="#">Edit</a>	4	First4	Last4	First4 Last4	05/10/1980 00:00:00

נראה כי קיבלנו את השורה עלייה לחצנו לעריכה, ובמקום ה קישור ל – "Edit" קיבלנו שני קישורים עבור "Update" ו – "Cancel" כאשר לחיצה על "Update" תבצע את המתודה אותה כתבנו קודם לכן.

נשנה את השם הפרטி ל – Web ואת שם המשפחה ל – Master ולחץ על Update Master:

Edit	ID	FirstName	LastName	FullName	BirthDate
<a href="#">Edit</a>	0	First0	Last0	First0 Last0	01/10/1980 00:00:00
<a href="#">Edit</a>	1	First1	Last1	First1 Last1	02/10/1980 00:00:00
<a href="#">Edit</a>	2	Web	Master	Web Master	10/03/1980 00:00:00
<a href="#">Edit</a>	3	First3	Last3	First3 Last3	04/10/1980 00:00:00
<a href="#">Edit</a>	4	First4	Last4	First4 Last4	05/10/1980 00:00:00

cut ניתן לראות את השינוי.

# מדריך ASP.NET – עבודה עם נתונים: תצוגת Master/Detail

38

Sela • IdoFlatow



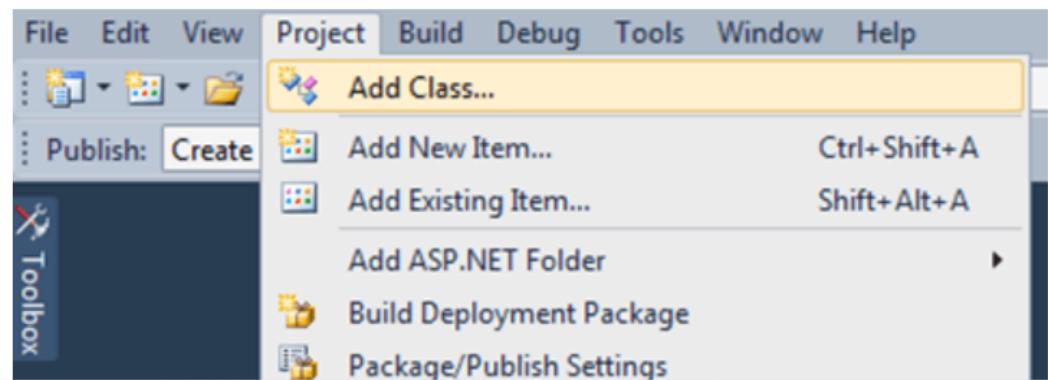
לעתים, כאשר יש לנו הרבה מאפיינים בחלוקת, לא נרצה להציג את כלם בפקד ה-GridView. לאחר זה יגרום לutableה המוצגת להיות מאוד רחבה ולא יהיה נוח לעבוד איתה. לכן, נציג בutableה רק את המאפיינים החשובים.

במקרה זה שבו לא מוצגים כל מאפייני האובייקט, נרצהאפשרות לסמן שורה בutableה ולקבל פירוט של כל המאפיינים של האובייקט הנבחר, כולל המאפיינים שלא הוצגו בutableה. מבנה זה של .”Master/Detail + פירוט נוסף מכונה “תצוגת GridView

במדריך זה נראה כיצד ליצור מסך עם תצוגת Master/Details.

## שלב 1 – הכנה

נוסף מחלוקת חדשה בשם Person לפרויקט שלנו (בחירה ב-Project Add Class מתפרק ב-Project Person).



```

public class Person
{
    public int ID { get; set; }
    public string FirstName { get; set; }
    public string LastName { get; set; }
    public string FullName
    {
        get
        {
            return string.Format("{0} {1}", FirstName, LastName);
        }
    }
    public DateTime BirthDate { get; set; }
}

```

נוסף עוד מחלקה בשם PersonLogic שתכיל פעולה Select שמחזירה את רשימת האנשים שיוצגו ב-GridView. את רשימת האנשים נוכל לאחזר מבוסיס הנתונים, אך לשם הדגמה אנחנו נשמר אותו ב-Session, שכן נוסיף גם מאפיין למחלקה חדשה שיחזיר רשימה של אנשים.

הקוד של המחלקה PersonLogic יראה כך:

```

public class PersonLogic
{
    private List<Person> PersonList
    {
        get
        {
            return HttpContext.Current.Session["People"]
                as List<Person>;
        }
        set
        {
            HttpContext.Current.Session["People"] = value;
        }
    }
}

```

```

public List<Person> SelectPeople()
{
    if (PersonList == null)
    {
        List<Person> people = new List<Person>();
        for (int i = 0; i < 5; i++)
        {
            people.Add(new Person()
            {
                ID = i,
                FirstName = string.Format("First{0}", i),
                LastName = string.Format("Last{0}", i),
                BirthDate = new DateTime(1980, 10, i + 1),
            });
        }
        PersonList = people;
    }

    return PersonList;
}
}

```

נוסף דף ASPX למערכת שלנו ונוסף לדף פקד ObjectDataSource. נעדכן את הפקד שיצביע על המתודה שיצרנו במחלקה PersonLogic (שים לב שהשדה TypeName מכיל את שם ה- Namespace ויתכן שתצטרכו לתקן אותו בהתאם ל-Namespace שלכם):

```

<asp:ObjectDataSource runat="server" ID="ObjectDataSource1"
    TypeName="DataDemo.PersonLogic"
    SelectMethod="SelectPeople"></asp:ObjectDataSource>

```

נוסיף לדף פיקד GridView ונכבייע מותך ה – ObjectDataSource על ה GridView – .זהו GridView – הכתיב המלא של ה :

```
<asp:GridView ID="GridView1" EnableViewState=true runat="server"
    DataSourceID="myDataSource" AutoGenerateColumns=False>
    <Columns>
        <asp:BoundField DataField="ID" HeaderText="ID" />
        <asp:BoundField DataField="FirstName"
            HeaderText="FirstName"/>
        <asp:BoundField DataField="LastName"
            HeaderText="LastName"/>
        <asp:BoundField DataField="FullName"
            HeaderText="FullName" ReadOnly=True/>
        <asp:BoundField DataField="BirthDate"
            HeaderText="BirthDate"/>
    </Columns>
</asp:GridView>
```

## שלב 2 – הוספה עמודת פקודה Grid – ל (CommandColumn)

נוסיף בתוך הגדרות ה – Grid את העמודה הבאה:

```
<asp:CommandField ShowSelectButton="true" />
```

נוסיף גם עיצוב לשורה הנבחרת על מנת להבדיל אותה משאר השורות:

```
<SelectedRowStyle BackColor="#0099FF" />
```

כך שהכיתוב של ה – Grid יראה כך:

```
<asp:GridView ID="GridView2" runat="server"
    DataSourceID="myDataSource" AutoGenerateColumns=False"
    DataKeyNames="ID" EnablePersistedSelection=True>
    <Columns>
        <asp:BoundField DataField="ID" HeaderText="ID"
            HeaderStyle-Width="0px" ItemStyle-Width="0"
            Visible=true SortExpression="ID" />
        <asp:BoundField DataField="FirstName" HeaderText="FirstName"
            SortExpression="FirstName" />
        <asp:BoundField DataField="LastName" HeaderText="LastName"
            SortExpression="LastName" />
```

```

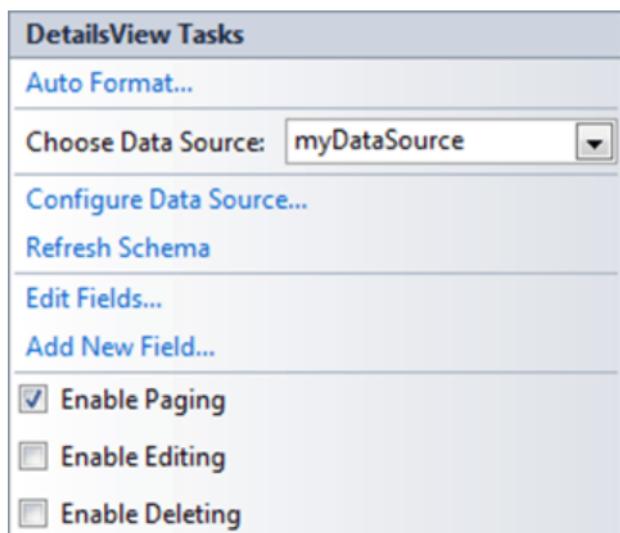
<asp:BoundField DataField="FullName" HeaderText="FullName"
    ReadOnly="True" SortExpression="FullName" />
<asp:BoundField DataField="BirthDate" HeaderText="BirthDate"
    SortExpression="BirthDate" />
    <asp:CommandField ShowSelectButton="true" />
</Columns>
<SelectedRowStyle BackColor="#0099FF" />
</asp:GridView>

```

## שלב 3 – הוספת DetailsView

נפתח את דף ASPX במצב Design, נסיף לדף פקד DetailsView (מה-Toolbox, תחת קטגורית Data), ונבחר לפקד את אותו DataSourceID GridView' כשל ה – :DataSource

לראות כיצד לבחור את ה –



ע"י לחיצה על Refresh Schema קיבל את אותו השודוט כשל ה – :GridView

```

<asp:DetailsView ID="DetailsView1" runat="server"
    AllowPaging="True" AutoGenerateRows="False"
    DataSourceID="myDataSource" Height="50px" Width="125px">
    <Fields>
        <asp:BoundField DataField="ID" HeaderText="ID"
            SortExpression="ID" />
        <asp:BoundField DataField="FirstName" HeaderText="FirstName"
            SortExpression="FirstName" />
        <asp:BoundField DataField="LastName" HeaderText="LastName" />
        <asp:BoundField DataField="FullName" HeaderText="FullName"
            ReadOnly="True" SortExpression="FullName" />
        <asp:BoundField DataField="BirthDate" HeaderText="BirthDate"
            SortExpression="BirthDate" />
    </Fields>
</asp:DetailsView>

```

## שלב 4 – סינכרון השורה הנבחרת ב-Grid עם ה- DetailsView

43

נוסף שני מאדים אחד עבור השורה הנבחרת (SelectedIndexChanged) -EventHandlers :GridView

```
<asp:GridView ID="GridView2" runat="server"
    DataSourceID="myDataSource" AutoGenerateColumns="False"
    DataKeyNames="ID" EnablePersistedSelection="True"
    OnSelectedIndexChanged="GridView1_SelectedIndexChanged">
    <Columns>
        <asp:BoundField DataField="ID" HeaderText="ID"
            HeaderStyle-Width="0px" ItemStyle-Width="0"
            Visible="true" SortExpression="ID" />
        <asp:BoundField DataField="FirstName" HeaderText="FirstName"
            SortExpression="FirstName" />
        <asp:BoundField DataField="LastName" HeaderText="LastName"
            SortExpression="LastName" />
        <asp:BoundField DataField="FullName" HeaderText="FullName"
            ReadOnly="True" SortExpression="FullName" />
        <asp:BoundField DataField="BirthDate" HeaderText="BirthDate"
            SortExpression="BirthDate" />
        <asp:CommandField ShowSelectButton="true" />
    </Columns>
    <SelectedRowStyle BackColor="#0099FF" />
</asp:GridView>
```

והשני עבור שינוי העמוד ב – :DetailsView

```
<asp:DetailsView ID="DetailsView1" runat="server"
    AllowPaging="True" AutoGenerateRows="False"
    DataSourceID="myDataSource" Height="50px" Width="125px"
    OnPageIndexChanged="DetailsView1_PageIndexChanged">
    <Fields>
        <asp:BoundField DataField="ID" HeaderText="ID"
            SortExpression="ID" />
        <asp:BoundField DataField="FirstName" HeaderText="FirstName"
            SortExpression="FirstName" />
        <asp:BoundField DataField="LastName" HeaderText="LastName"
            SortExpression="LastName" />
        <asp:BoundField DataField="FullName" HeaderText="FullName"
            ReadOnly="True" SortExpression="FullName" />
        <asp:BoundField DataField="BirthDate" HeaderText="BirthDate"
            SortExpression="BirthDate" />
    </Fields>
</asp:DetailsView>
```

```

protected void GridView1_SelectedIndexChanged(
    object sender, EventArgs e)
{
    DetailsView1.SetPageIndex(GridView1.SelectedIndex);
}

protected void DetailsView1_PageIndexChanged(
    object sender, EventArgs e)
{
    GridView1.SelectedIndex = DetailsView1.PageIndex;
}

```

cut נראה את התוצאות (ע"י הקלקה ימנית על הדף ו – :

ID	FirstName	LastName	FullName	BirthDate	
0	First0	Last0	First0 Last0	01/10/1980 00:00:00	Select
1	First1	Last1	First1 Last1	02/10/1980 00:00:00	Select
2	First2	Last2	First2 Last2	03/10/1980 00:00:00	Select
3	First3	Last3	First3 Last3	04/10/1980 00:00:00	Select
4	First4	Last4	First4 Last4	05/10/1980 00:00:00	Select

ID	0
FirstName	First0
LastName	Last0

נלחץ על המספר 3 למטה ונראה כי גם השורה ב – GridView השתנתה:

ID	FirstName	LastName	FullName	BirthDate	
0	First0	Last0	First0 Last0	01/10/1980 00:00:00	Select
1	First1	Last1	First1 Last1	02/10/1980 00:00:00	Select
2	First2	Last2	First2 Last2	03/10/1980 00:00:00	Select
3	First3	Last3	First3 Last3	04/10/1980 00:00:00	Select
4	First4	Last4	First4 Last4	05/10/1980 00:00:00	Select

ID	2
FirstName	First2
LastName	Last2
FullName	First2 Last2
BirthDate	03/10/1980 00:00:00

1	2	3	4	5
---	---	---	---	---

תצוגת Master/Detail יכולה לשמש אותנו להרחבת המידע המוצג בטבלה, אך היא גם יכולה לשמש אותנו עבור תצוגת מידע נוסף, לדוגמה נוכל להציג טבלה של קטגוריות מוצרים וכשהמשתמש מסמן קטgorיה, להציג לו טבלה נוספת של רשימת המוצרים בקטgorיה. גם תצוגה כזו מכונה **תצוגת Master/Detail**.

## מדריך ASP.NET – שימוש ב-SqlDataSource

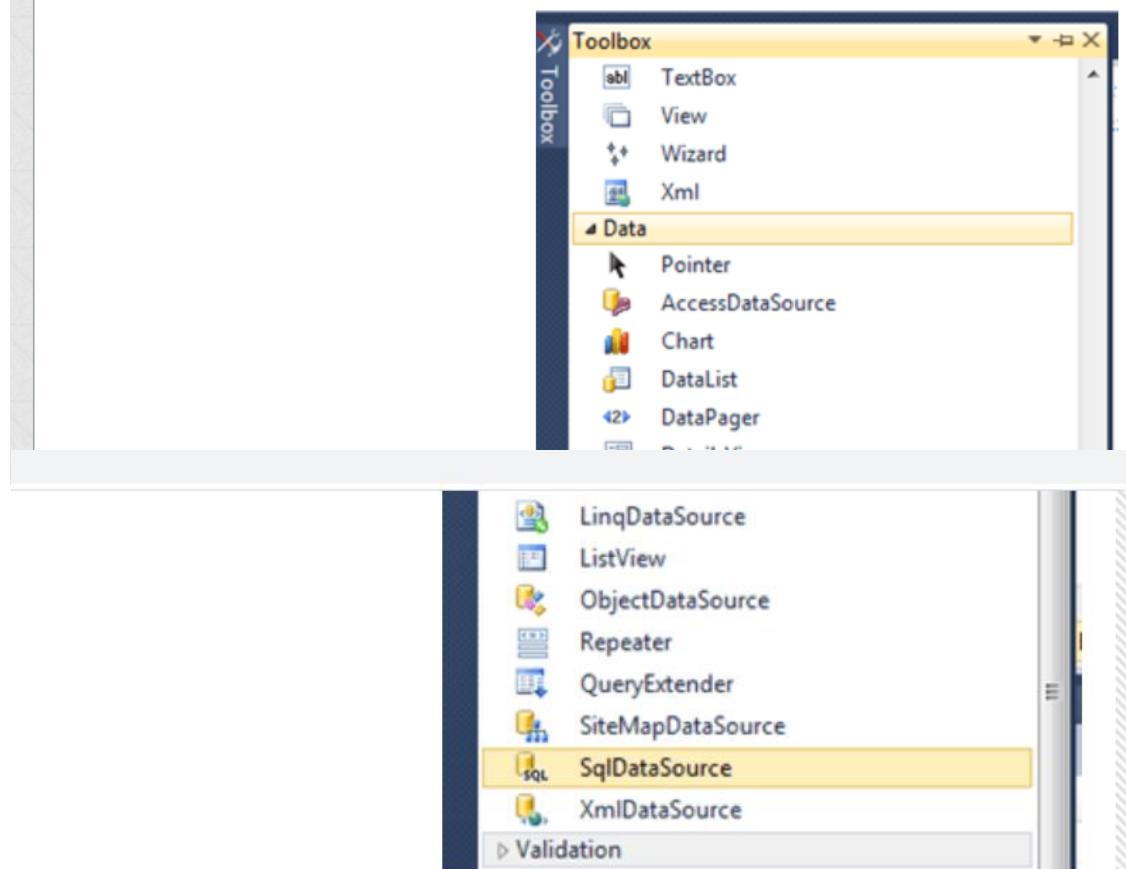
Sela • IdoFlatow



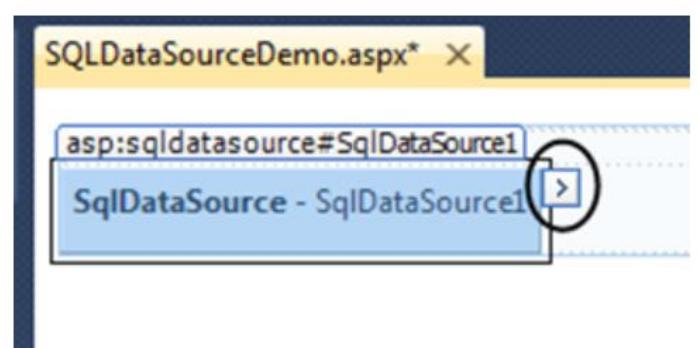
במדריך זה נלמד כיצד לטעון נתונים באמצעות SqlDataSource ולחזקם על המסך.

מדריך זה משתמש בבסיס הנתונים Northwind אשר ניתן להורדה [כאן](#).

בדף שלנו נווסף SqlDataSource ע"י בחירה מה – Toolbox . האובייקט הזה נמצא בקטגורית :Data



נבחר אותו לדף שלנו, ונלחץ על החץ המסומן בעיגול:



:Configure data sourceicha shaiā

asp:SqlDataSource#SqlDataSource1

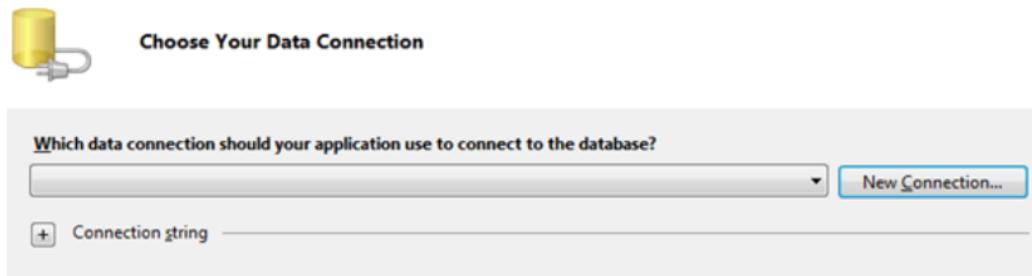
SqlDataSource - SqlDataSource1

SqlDataSource Tasks

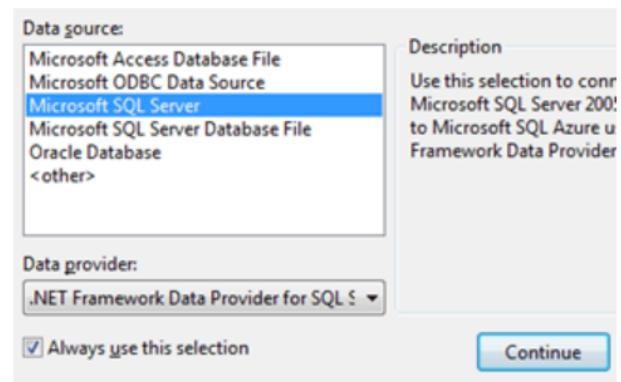
[Configure Data Source...](#)

Configure your data source's settings.

לאחר שיפתח החלון של האשפ, נלחץ על כפתור ה – New Connection –



בחלון שייפתח נבחר ב – Continue Microsoft SQL Server – ונלחץ על



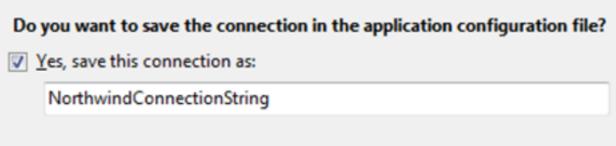
בחלון שייפתח כתת נכתוב את שם השרת:

Server name:  
.\sqlexpress

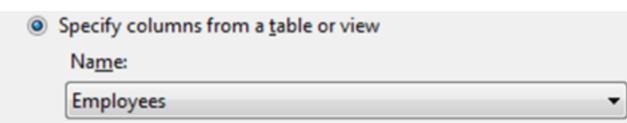
לאחר שנמצא מסודה שם השרת נוכל לבחור מהרשימה למטה את שם בסיס הנתונים, נוכל לעשות זאת ע"י בחירה או ע"י כתיבת השם:



לאחר שבחרנו את השם, נלחץ OK. לאחר שבחזר לאשפ, נלחץ על Next, כאן נשאל האם לשומר את מחרוזת פרטי ההתקשרות (Connection string) בערך קבוע הגדרות:



דף לענות על שאלה זאת בחיבור ولكن נשאיר את תיבת הסימון מסומנת. ניתן לשנות את השם של ה – Connection string, אנחנו נשאיר אותו כמו שהוא ונלחץ על Next. בשלב הבא, נבחר את הטבלה הרצiosa לנו:



בחלק התיכון של המסר נראה את העמודות של אותה הטבלה. נוכל לבחור עמודות מסוימות או בבחירה ב – \* ולהביא את כל העמודות.

47

נבחר את העמודות EmployeeID, LastName, FirstName ו-Title. למטה נוכל לראות את (SELECT Statement SQL שנבנית כתוצאה מהבחירה שלנו) (תחת הcotreta שאלתת ה – SQL

Columns:

*	Region
<input checked="" type="checkbox"/> EmployeeID	PostalCode
<input checked="" type="checkbox"/> LastName	Country
<input checked="" type="checkbox"/> FirstName	HomePhone
<input checked="" type="checkbox"/> Title	Extension
<input type="checkbox"/> TitleOfCourtesy	Photo
<input type="checkbox"/> BirthDate	Notes
<input type="checkbox"/> HireDate	ReportsTo
<input type="checkbox"/> Address	PhotoPath
<input type="checkbox"/> City	

SELECT statement:

```
SELECT [EmployeeID], [LastName], [FirstName], [Title] FROM [Employees]
```

לאחר שבחרנו נלחץ על Next וגיעו לשלב של בדיקת השאלתת:

EmployeeID	LastName	FirstName	Title
1	Davolio	Nancy	Sales Representative
2	Fuller	Andrew	Vice President, Sales
3	Leverling	Janet	Sales Representative
4	Peacock	Margaret	Sales Representative
5	Buchanan	Steven	Sales Manager
6	Suyama	Michael	Sales Representative
7	King	Robert	Sales Representative
8	Callahan	Laura	Inside Sales Coordinator
9	Dodsworth	Anne	Sales Representative

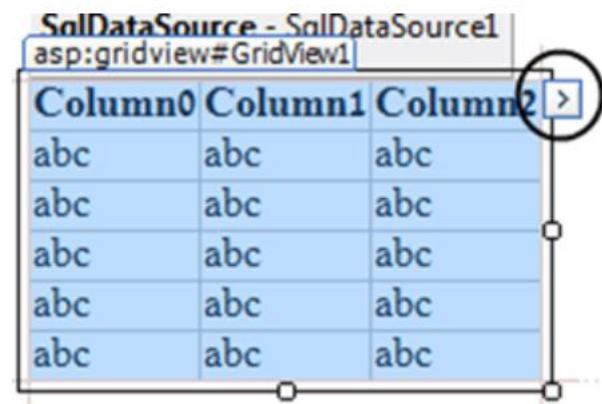
Test Query

SELECT statement:

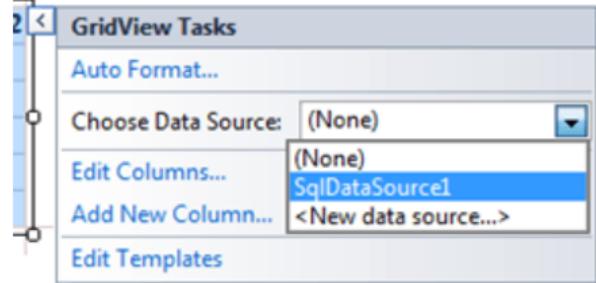
```
SELECT [EmployeeID], [LastName], [FirstName], [Title] FROM [Employees]
```

לחיצה על TestQuery תביא את תוצאות השאלתת, עצת נותר לנו רק ללחוץ על Finish.

עת, נזרף לדף שלנו GridView. לאחר שנגירור אותו לדף, נלחץ על החץ שלו (מוסמן בעיגול):



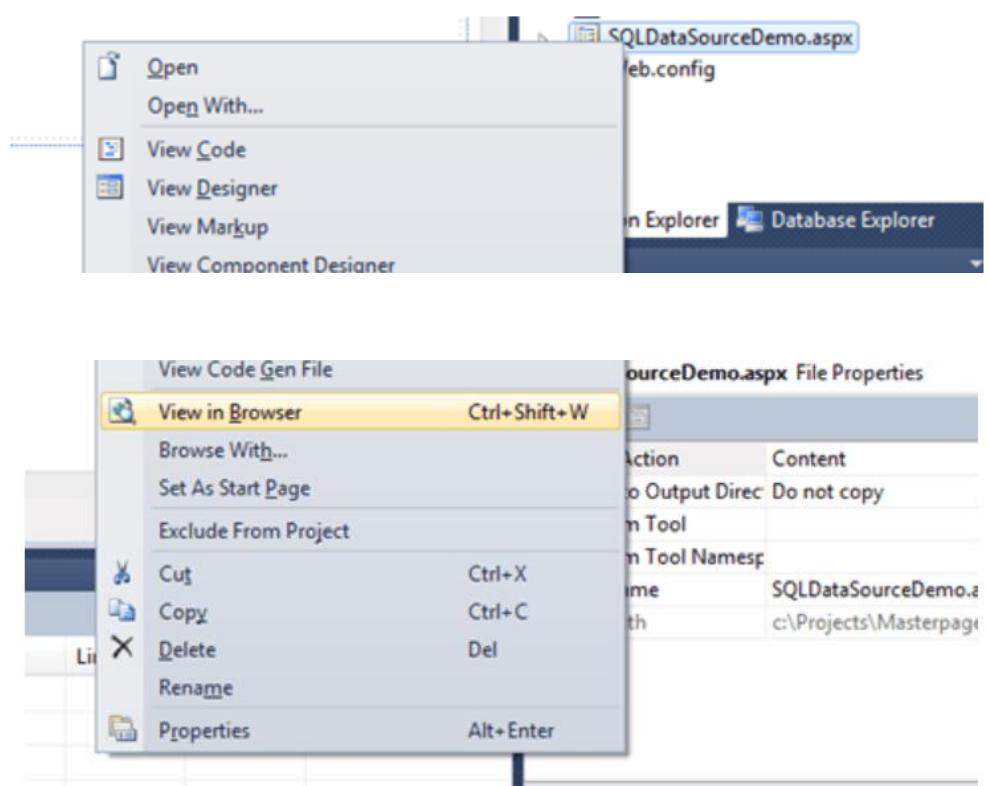
בתפריט שנפתח, נחבר ל – GridView את ה – SqlDataSource שהעטת סיימנו להגדיר:



נודא שה – Gridview קיבל שדות בהתאם לשאלתך:

EmployeeID	LastName	FirstName	Title
0	abc	abc	abc
1	abc	abc	abc
2	abc	abc	abc
3	abc	abc	abc
4	abc	abc	abc

כל מה שנותר לנו כעת הוא לבדוק איך זה נראה בדף. נעשה זאת ע"י הקלקה ימנית על קובץ ה – ASPX ובחירה ב-View in Browser :



וזו תראה התוצאה בדף:

EmployeeID	LastName	FirstName	Title
1	Davolio	Nancy	Sales Representative
2	Fuller	Andrew	Vice President, Sales
3	Leverling	Janet	Sales Representative
4	Peacock	Margaret	Sales Representative

# מדריך – ASP.NET – בניית Master Page

Sela • IdoFlatow



לijk 1

שתף

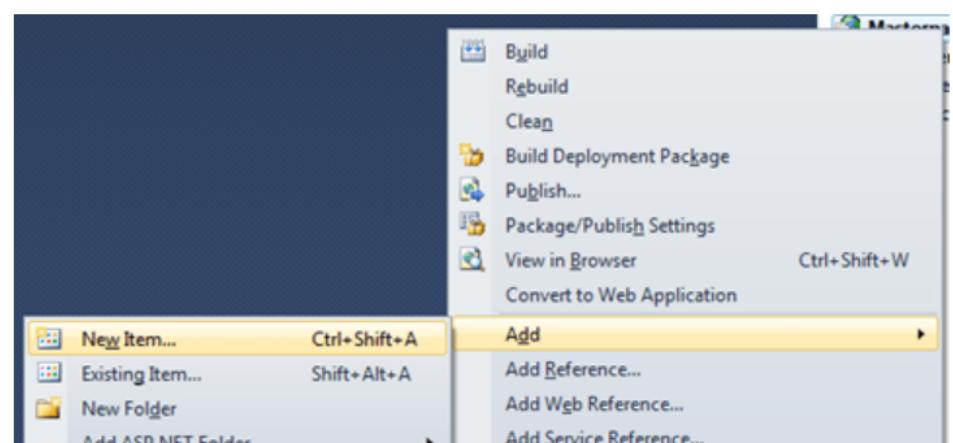
כאשר נבנה מערכת Web, נרצה שהעמודים השונים יהיו בעלי אותה תבנית, לעיתים נרצה לשלב בכל העמודים את אותו אזור לתפריט, כותרות ופרטים כלליים על האתר.

לשם בניית התבנית זו נוכל להעזר ב – Master page. Master page הוא קובץ בעל סימט Master המהווה את התבנית לעמודים השונים.

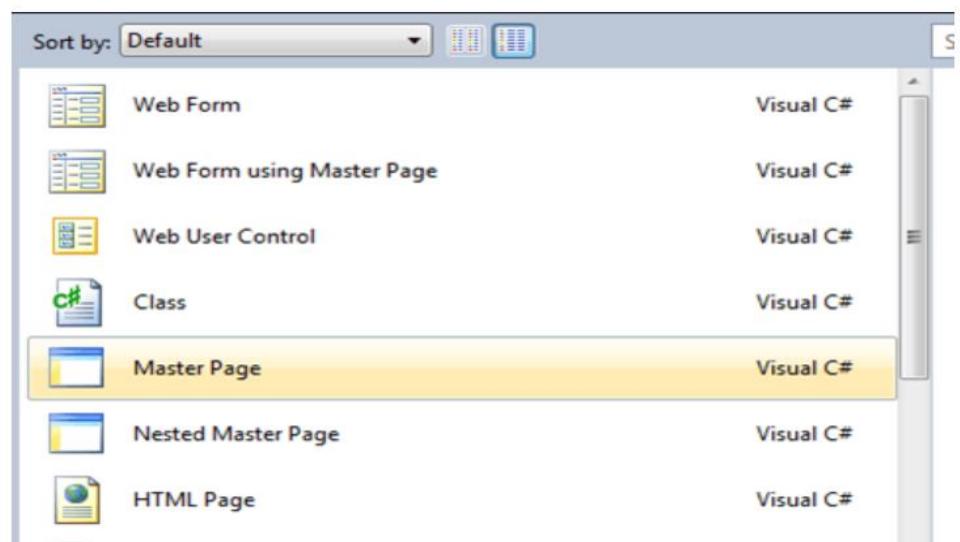
במדריך זה נראה כיצד בונים Master page וכי怎 משלבים אותו בדף השונים.

## יצירת Master Page

בפרויקט שלנו נבחר ב – Add New Item



בתפריט שנפתח נבחר ב – Master Page



ניתן לקובץ שם, לצורך הדגמה קראנו לקובץ Site.Master. נתבונן בקוד שנוצר לנו באופן אוטומטי:

```

<%@ Master Language="C#" AutoEventWireup="true"
CodeBehind="Site.master.cs" Inherits="Demos.Site" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html >="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
    <asp:ContentPlaceHolder ID="head" runat="server">
        </asp:ContentPlaceHolder>
    </head>
    <body>
        <form id="form1" runat="server">
            <div>
                <asp:ContentPlaceHolder
                    ID="ContentPlaceHolder1" runat="server">
                    </asp:ContentPlaceHolder>
            </div>
        </form>
    </body>
</html>

```

שימוש לב לתגים `asp:ContentPlaceHolder` , נשוב אליהם בהמשך. בימתיים נראה שנותר לנו קובץ עם תוכן דומה למה שנוצר ב – ASPX ואנחנו יכולים להוסיף לו פקדים כראות עינינו.

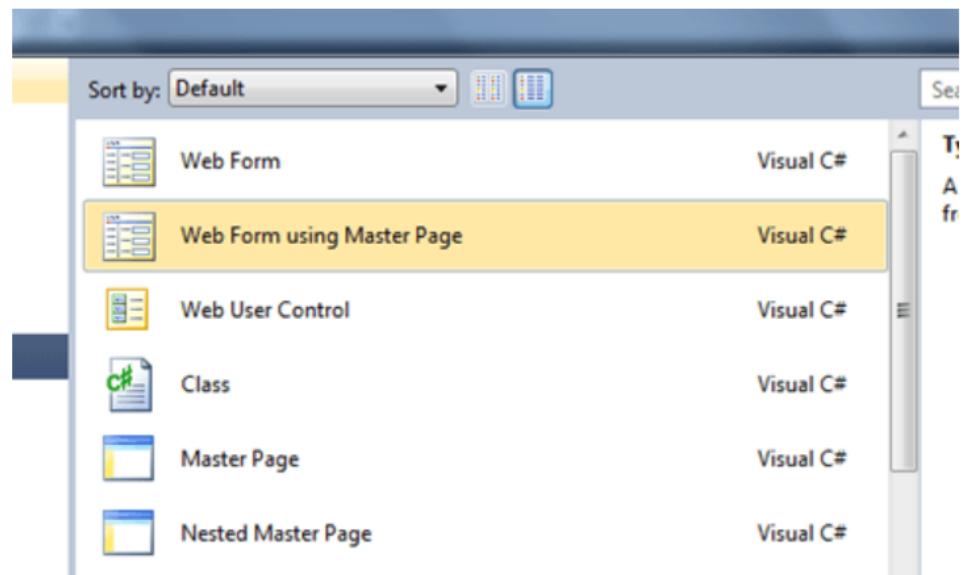
לצורך הדוגמא נוסיף `Label` לפני ה `ContentPlaceHolder` – והוא יראה כך :

```
<%@ Master Language="C#" AutoEventWireup="true"
CodeBehind="Site.master.cs" Inherits="Demos.Site" %>

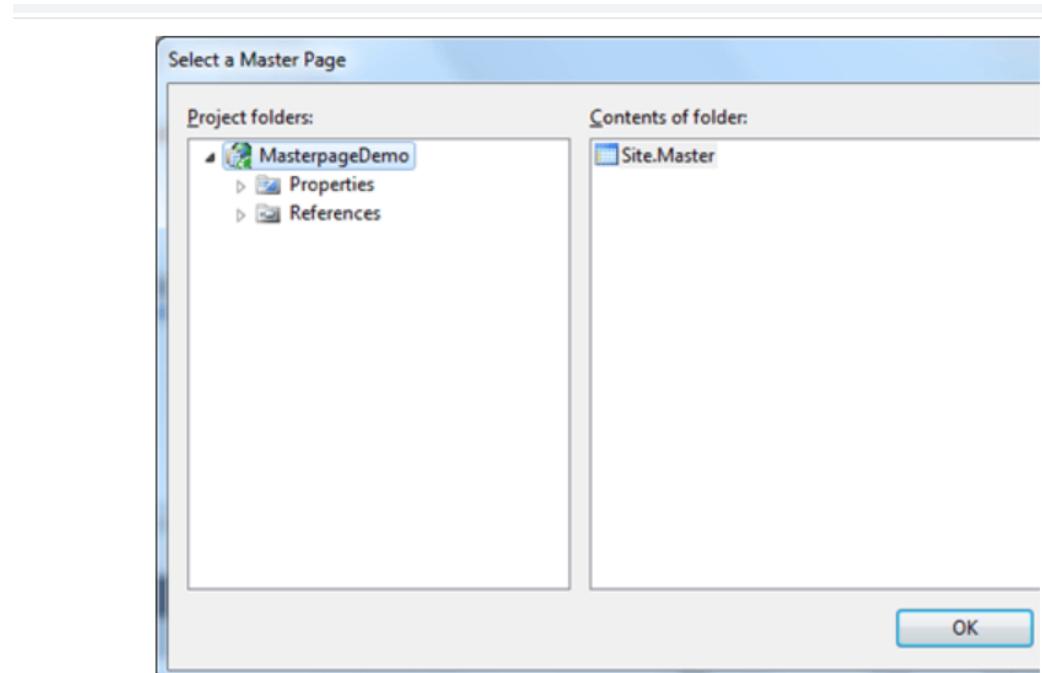
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html >="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
    <asp:ContentPlaceHolder ID="head" runat="server">
        </asp:ContentPlaceHolder>
</head>
<body>
    <form id="form1" runat="server">
        <div>
            <asp:Label ID="Label1" runat="server"
                Text="Your Content Here:" />
            <asp:ContentPlaceHolder
                ID="ContentPlaceHolder1" runat="server">
                </asp:ContentPlaceHolder>
            </div>
        </form>
    </body>
</html>
```

נבחר שוב ב - Add new item - Web Form using Master Page



ניתן לו שם, אנחנו צריכים להציג הדוגמה קראנו לו כתה, עלינו לבחור מהו ה Master Page בו הדף החדש שלנו משתמש, היה וקיים לנו בפרויקט רק Master Page אחד, נבחר בו, ונלחץ OK.



נתבונן כתה בקוד שנוצר לנו באופן אוטומטי בדף החדש:

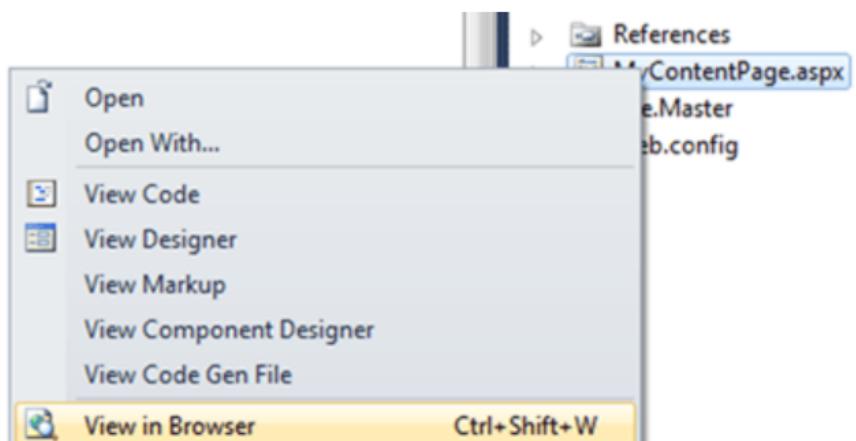
```
<%@ Page Title="" Language="C#" MasterPageFile("~/Site.Master")
   AutoEventWireup="true" CodeBehind="myContentPage.aspx.cs"
   Inherits="Demos.myContentPage" %>
<asp:Content
   ID="Content1" ContentPlaceHolderID="head" runat="server">
</asp:Content>
<asp:Content ID="Content2"
   ContentPlaceHolderID="ContentPlaceHolder1" runat="server">
</asp:Content>
```

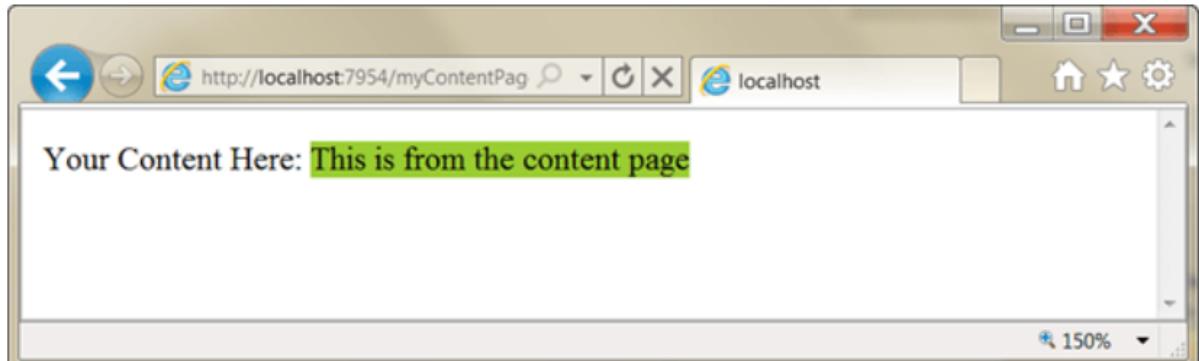
נראה בשורה הראתה את המאפיין המצביע על ה - **Master** **MasterPageFile** המצביע על ה - **Page** ושני פקדי תוכן (**asp:Content**). (**asp:Content**) לפקד **asp:Content** ישנו מאפיין בשם **ContentPlaceHolderID**, תפקיד המאפיין זהה הוא להצביע על הפקד **.asp:ContentPlaceHolder**

נוסיף **Label** בתוך פקד התוכן השני וניתן לו גם צבע רקע על מנת להבדילו. הקוד בדף שלנו יראה כך:

```
<%@ Page Title="" Language="C#" MasterPageFile "~/Site.Master"
   AutoEventWireup="true" CodeBehind="myContentPage.aspx.cs"
   Inherits="Demos.myContentPage" %>
<asp:Content
   ID="Content1" ContentPlaceHolderID="head" runat="server">
</asp:Content>
<asp:Content ID="Content2"
   ContentPlaceHolderID="ContentPlaceHolder1" runat="server">
<asp:Label ID="Label1" Text="This is from the content page"
   BackColor="YellowGreen" runat="server" />
</asp:Content>
```

כעת, נראה את הדף ע"י הקלקה ימנית ובחירה ב - **:View In Browser** -





וב - :HTML

```

<html >="http://www.w3.org/1999/xhtml">
<head><title>

</title>
</head>
<body>
    <form method="post" action="MyContentPage.aspx" id="form1">

        <div class="aspNetHidden">
            <input type="hidden" name="__VIEWSTATE" id="__VIEWSTATE"
                value="/wEPDwULLTE1MjQ5ODA0NjlkZAtf8DceGtYv0R7C9YhfJH
                sSVzW/gHz3gnws5Rw9DL" />
        </div>
        <div>
            <span>Your Content Here:</span>
            <span style="background-color:YellowGreen;">
                This is from the content page</span>
        </div>
    </form>
</body>
</html>

```

ניתן לראות כיצד CAN שחלק מה-HTML הגיע מה-**Master Page** – והחלק מהדף עצמו. ניתן להוסיפה דפים נוספים המשמשים באותו **Master Page** ולשנות את תוכנם.

ניתן לשנות את התוכן של ה-**Master Page** ולראות איך כל הדפים שהוספנו ומשמשים בו, תוכנם משתנה בהתאם.



שיתוף

לайק 0



פעמים רבות אנחנו צריכים לבצעolidציה על קלט שנשלח מהמשתמשמערכות שלנו. במדריך זה נראה את השימוש בolidטורים ב – ASP.NET, נכיר את הסוגים השונים ונראה את השימוש בהם.

## מהיolidציה ב – ASP.NET?

olidציה היא בדיקת הקלט לפני שאנו מבצעים אותו פעולה.olidציה ישנה שני תפקידים עיקריים:

1. נוחות למשתמש – במידה והמשתמש טעה ניתן להציג לו את השגיאה בצורה נוחה ומהירה
2. אבטחה – במידה ובכונה המשתמש הכניס קלט לא נכון, יש לבצעolidציה ולמנוע פעולה העוללה לגרום לנזק.

במערכות Web ישנה הפרדה שלolidציה לצד הלוקוחolidציה לצד השרת.olidציה לצד הלוקוח באחצורה נוחות המשתמש – אם הנתונים שגויים חבל לשלחם לשרת, אולם אין היא יכולה להיות בלבד וכן אנו צריכים גםlolidציה לצד השרת.

מה עושים ה – Validators ב – ASP.NET? את שנייהם! היתרונו הגדול שלהם הינו שאיןנו צריכים לנכתב קוד עבור השרת ועבור הלוקוח, ה – Validator יעשה העבודה לשניהם.

לצורך הדוגמה ניצור טופס רישום המכיל שם פרטי ושם משפחה. אנחנו צריכים להשתמש למלא את שנייהם ונציג הودעה במקרה אחד לא מולא.

## שלב 1 – ייצירת הטופס

56

ניצור את הטופס עם שתי תיבות טקסט וכפתור. ה – ASPX יראה כך:

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="ValidatorsDemo.aspx.cs"
Inherits="DemosValidatorsDemo" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html >="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
        <div>
            <asp:Label ID="Label1" Text="First Name"
                AssociatedControlID="txtFirstName" runat="server" />
            <asp:TextBox runat="server" ID="txtFirstName" />
            <br />
            <asp:Label ID="Label2" Text="Last Name" runat="server"
                AssociatedControlID="txtLastName" />
            <asp:TextBox runat="server" ID="txtLastName" /><br />
            <asp:Button Text="Submit" runat="server" ID="btnSubmit" />
        </div>
    </form>
</body>
</html>
```

כעת יש לנו טופס פשוט – ללאolidציה

## שלב 2 – הוספת Validators

אם נתבונן ב – Toolbox נראה שישנו איזור של Validation המכיל את ה – Validators של ASP.NET ופקד נוסף שנקרא ValidationSummary, תפקיד הפקד זהה הוא להציג את כל הודעות השגיאה, אם ישן, באזור אחד.

נosiפ' כעת את ה – Validators הנדרשים לנו – במקרה זהה אנו נדרשים ל – RequiredFieldValidator הגדרתו ב – ASPX תראה כך:

```
<asp:RequiredFieldValidator ID="firstNameValidator"
    ControlToValidate="txtFirstName" runat="server"
    ErrorMessage="Please enter First Name" Display="Dynamic"
    ForeColor="Red" SetFocusOnError="True"
    EnableClientScript="true"/>
```

נעבור על המאפיינים העיקריים:

1. ID של הפקד עליו עורכים את הולידציה, המאפיין החשוב ביותר.

2. ErrorMessage – הודעה השגיאה של ה-Validator.

3. SetFocusOnError – האם להביא את הסמן לפקד שהולידציה שלו נכשלה.

4. Display – אופן הציגות של השגיאה. CAN ישן שלוש אפשרויות:

None – אל תציג כל (אם משתמש ב-ValidationSummary הודעה השגיאה תוצג בו).

Dynamic – הציג אם יש שגיאה.

Static – הציג אם יש שגיאה, אם לא שומר מקום ריק.

– מה שהוא מציג במקרה של הודעה שגיאה, אם לא הוגדר מוצג – ErrorMessage

שימוש נפוץ הוא הצגת כוכבית בשדה זהה כאשר הודעה השגיאה (ErrorMessage) מוצגת ב-ValidationSummary.

6. האם לאפשר ולידציה מצד הלקוח. כבירית מחדל ערך שדה זה הוא true.

כעת, לאחר שעברנו על המאפיינים, נראה את הטופס שלנו לאחר הוספת ה-Validators –

```

<%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="ValidatorsDemo.aspx.cs"
Inherits="WebApplication17.ValidatorsDemo" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html >="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
        <div>
            <asp:Label ID="Label1" Text="First Name"
                AssociatedControlID="txtFirstName" runat="server" />
            <asp:TextBox runat="server" ID="txtFirstName" />
            <asp:RequiredFieldValidator ID="firstNameValidator"
                ControlToValidate="txtFirstName" runat="server"
                ErrorMessage="Please enter First Name" Text="*"
                Display="Dynamic" ForeColor="Red" SetFocusOnError="True"
                EnableClientScript="true"></asp:RequiredFieldValidator>
            <br />
            <asp:Label ID="Label2" Text="Last Name" runat="server"
                AssociatedControlID="txtLastName" />
            <asp:TextBox runat="server" ID="txtLastName" />
            <asp:RequiredFieldValidator ID="RequiredFieldValidator1"
                ControlToValidate="txtLastName" runat="server"
                ErrorMessage="Please enter Last Name" Text="*"
                Display="Dynamic" ForeColor="Red" SetFocusOnError="True"
                EnableClientScript="true"></asp:RequiredFieldValidator>
            <br />
            <asp:Button Text="Submit" runat="server"
                ID="btnSubmit" OnClick="btnSubmit_Click" />
            <asp:ValidationSummary ID="ValidationSummary"
                runat="server"/>
        </div>
    </form>
</body>
</html>

```

אם משנים את מאפייני EnableClientScripts בכל הValidators ל – false, הדף ישלח לשרת גם אם הליידציה נכשלה, כך שפעולות כגון שמירה לבסיס הנתונים יבוצעו למרות השגיאות, אם כי כשהעמוד יחזיר לדף, יוצגו הודעות השגיאה.

לכן, נצטרך בצד השרת להשתמש בבדיקה האם הדף עבר וליידציה. הקוד שלנו בצד השרת נראה כך:

```
protected void btnSubmit_Click(object sender, EventArgs e)
{
    if (Page.IsValid)
    {
        RegisterUser();
    }
}
```

כאשר בmethodה RegisterUser נכתב את הקוד שיבצע את פעולה הרישום, ונפנה לשם רק אם הדף עבר וליידציה.

## Validators

**מודוא שבקוד הנבדק קיים ערך :** RequiredFieldValidator 

**מודוא שהערך בפקד הנבדק הוא בין שני ערכים.** מאפיינים עיקריים:

( ערכי **מינימום ומקסימום** **MaximumValue, MinimumValue**

- **סוג הנתון** שיש בפקד (**מספר, תאריך ...**)

**משווה את הערך בפקד הנבדק לעומת הערך בפקד אחר (או ערך קבוע) וקובע אם הוא תקין או לא בהתאם להשוואה המוגדר.** מאפיינים עיקריים:

- **הפקד בו נמצא הערך אליו משווים את הערך בפקד הנבדק**

, **ControlToCompare - הערך אליו משווים את הערך בפקד הנבדק (אם גם Mogdrim אזי הבדיקה תהיה מול ValueToCompare)**

- **סוג הנתון** שיש בפקד **Type**

, **הבדיקה אותה עושים.** **Operator**

- **הערך תקין אם שווה ל...** **Equal**

- **הערך תקין אם גדול מ...** **GreaterThan**

- **הערך תקין אם שווה או גדול מ...** **GreaterThanOrEqual**

- **הערך תקין אם קטן מ...** **LessThan**

- **הערך תקין אם שווה או קטן מ...** **LessThanOrEqual**

- **הערך תקין אם שונה מ...** **NotEqual**

- **הערך תקין אם הוא מאותו סוג** **DataTypeCheck**

4. **RegularExpression –** בדיקה האם הערך תואם ל-RegularExpressionValidator. טוב לבודיקה עבור שדות עם פורמט מוגדר כמו דואר אלקטרוני, מספר טלפון, כתובות אטר (וכו').

5. **CustomValidator –** אם נרצה לפתח ולידציה משל עצמנו נוכל להשתמש ב- Validator – **CustomValidator**. כאן נצטרך לכתוב מתודה עבור OnServerValidate שתופעל כאשר

ה

יבצע ולידציה. אם נרצה להפעיל גםolidציה מצד לקוח נגדיר את המאפיין ClientSideValidationFunction לשם פונקציה מצד הלקוח שתופעל בזמן ולידציה.

# מדריך ASP.NET – אבטחה: חיבור ל- Membership Provider

Sela • IdoFlatow



הדרישה מחלוקת גדול ממערכות Web בכלל, ומאפייניות ASP.NET בפרט הוא שהן יהיו מאובטחות ויכללו הרשותות שונות למשתמשים שונים.

ניהול המשתמשים יכול להיות בתוך המערכת יוכל להיות מחוץ לה. במדריך זה נדגים כיצד ניתן להשתמש במודול ניהול המשתמשים של ASP.NET שנקרא גם ASP.NET Membership Provider.

## שלב 1 – הגדרת בסיס הנתונים

המודול הזה שומר בסיס נתונים את נתוני המשתמשים. על מנת להקים את בסיס הנתונים זהה יש להשתמש בתכנית aspnet\_regsql.

ניתן להפעיל את התוכנית מתוך תיקיית ה-.Framework (מתוך C:\Windows\Microsoft.net\framework\v4.0.30319).

ב�行 פעולה התוכנית יפתח לנו אשף אשר יבקש מייתנו את שם שרת ה – SQL ואת שם בסיס הנתונים.

המסמך הראשון הוא מסך פתיחה, כל שיש לנו לעשות הוא לחוץ על Next.

במסך השני נתקבש לבחור האם להגדיר בסיס נתונים על שרת SQL עבור Membership provider או להסיר, נבחר להגדיר ונלחץ על Next

What database task do you want to perform?

- Configure SQL Server for application services

This option runs a script that creates a new database or configures an existing database to store information for ASP.NET membership, profiles, role management, personalization and SQL Web event provider.

במסך השלישי נגדיר את השרת ואת שם בסיס הנתונים

Server: .\sqlexpress  
 Windows authentication  
 SQL Server authentication  
User name:  
Password:  
Database: <default>

במסך הרביעי נראה את סיכום ההגדרות, במידה וההגדרות נכונות יש ללחוץ על Next

#### Settings summary:

Server name: .\sqlexpress  
Database name: aspnetdb

Click **Next** to continue or **Previous** to change your settings.

ולסיום במסך האחרון יש ללחוץ על Finish

## שלב 2 – חיבור בסיס הנתונים לאפליקציה

ראשית נגדיר Connection string מתאים לבסיס הנתונים שזה עתה הוגדר. ה – `<connectionStrings>` יוגדר בתוך הקובץ `web.config` בתוך האלמנט `string`

```
<connectionStrings>
  <add
    name="MembershipProviderDB"
    connectionString="Data Source =.\sqlexpress;
    initial catalog=aspnetdb;Integrated Security=True"
    providerName="System.Data.SqlClient" />
</connectionStrings>
```

[connecti](#)

```

<system.web>
    <membership defaultProvider="SqlProvider"
        userIsOnlineTimeWindow="15">
        <providers>
            <clear />
            <add
                name="SqlProvider"
                type="System.Web.Security.SqlMembershipProvider"
                connectionStringName="MembershipProviderDB"
                applicationName="MyApplication"
                enablePasswordRetrieval="false"
                enablePasswordReset="true"
                requiresQuestionAndAnswer="true"
                requiresUniqueEmail="true"
                passwordFormat="Hashed" />
        </providers>
    </membership>
    <compilation debug="true" targetFramework="4.0" />
</system.web>

```

שים לב שהמאפיין `defaultProvider` נדרש להיות זהה לשם ה Provider והמאפיין `ConnectionStringName` נדרש להיות זהה לשם ה Provider – שנותנו על מנת להתחבר לבסיס הנתונים.

cutת המערכת שלנו מחוברת לבסיס נתונים של ניהול משתמשים. כל הקוד של שליפה מבסיס הנתונים כתובה לשם נמצא כבר ואנחנו רק צריכים להפעילו.



שתף

לינק



ה – Membership Provider מאפשר לנו גם להגדיר משתמשים וקבוצות במערכת. במדריך זה נכיר את ממשך המשתמש המאפשר לעשות זאת.

על מנת להגיע למשתמש נפתח את תפריט Project ומתוכו נבחר ב-ASP.NET Configuration. כתוצאה מבחירה אפשרית זו בתפריט "פתח לנו דף" עם התצוגה הבאה:

**Welcome to the Web Site Administration Tool**

**Application:**/Demos  
**Current User Name:**DELL-IDOF\IDOF

<a href="#">Security</a>	Enables you to set up and edit users, roles, and access permissions for your site. Existing users: 0
<a href="#">Application Configuration</a>	Enables you to manage your application's configuration settings.
<a href="#">Provider Configuration</a>	Enables you to specify where and how to store administration data used by your Web site.

## הגדרת קבוצות משתמשים (Roles)

על מנת להגדיר Roles נלך לשונית של Security או שנלחץ על הקישור Security.

במסך הבא נבחר בקישור **Create or Manage roles**

You can use the Web Site Administration Tool to manage all the security settings for your application. You can set up users and passwords (authentication), create roles (groups of users), and create permissions (rules for controlling access to parts of your application).

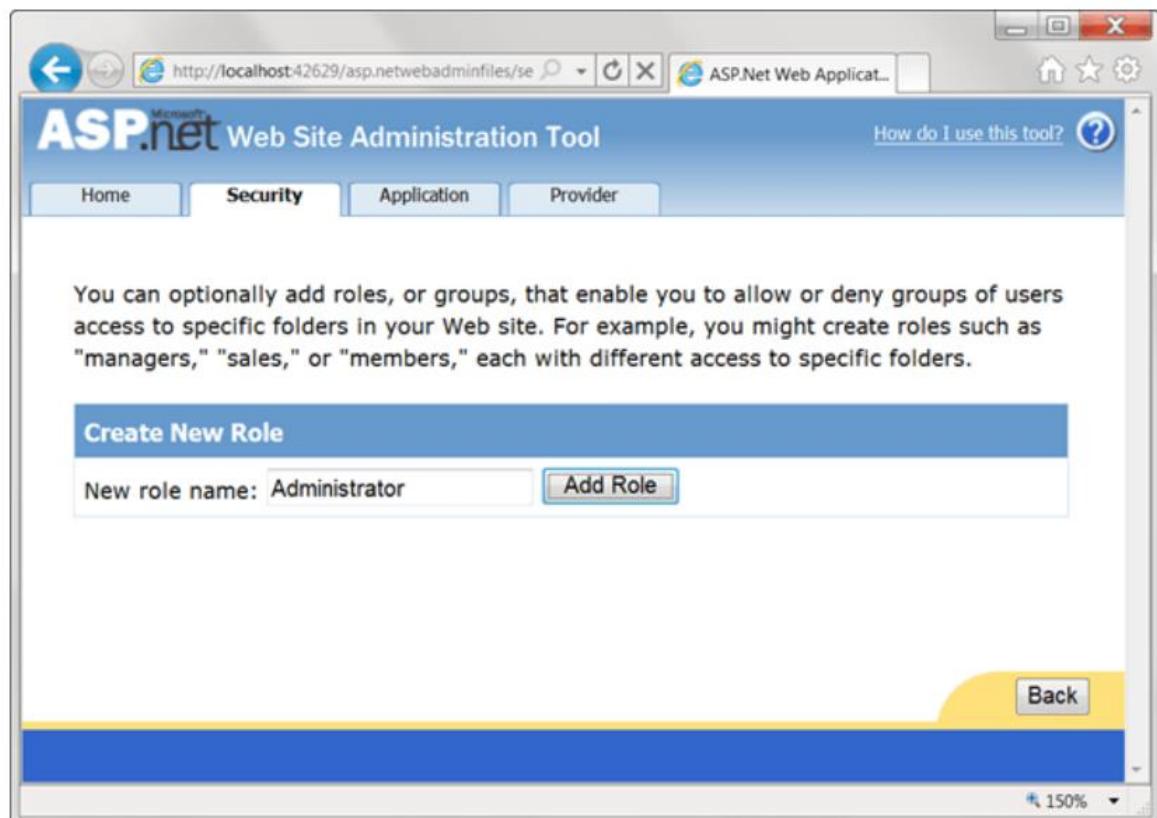
By default, user information is stored in a Microsoft SQL Server Express database in the Data folder of your Web site. If you want to store user information in a different database, use the Provider tab to select a different provider.

[Use the security Setup Wizard to configure security step by step.](#)

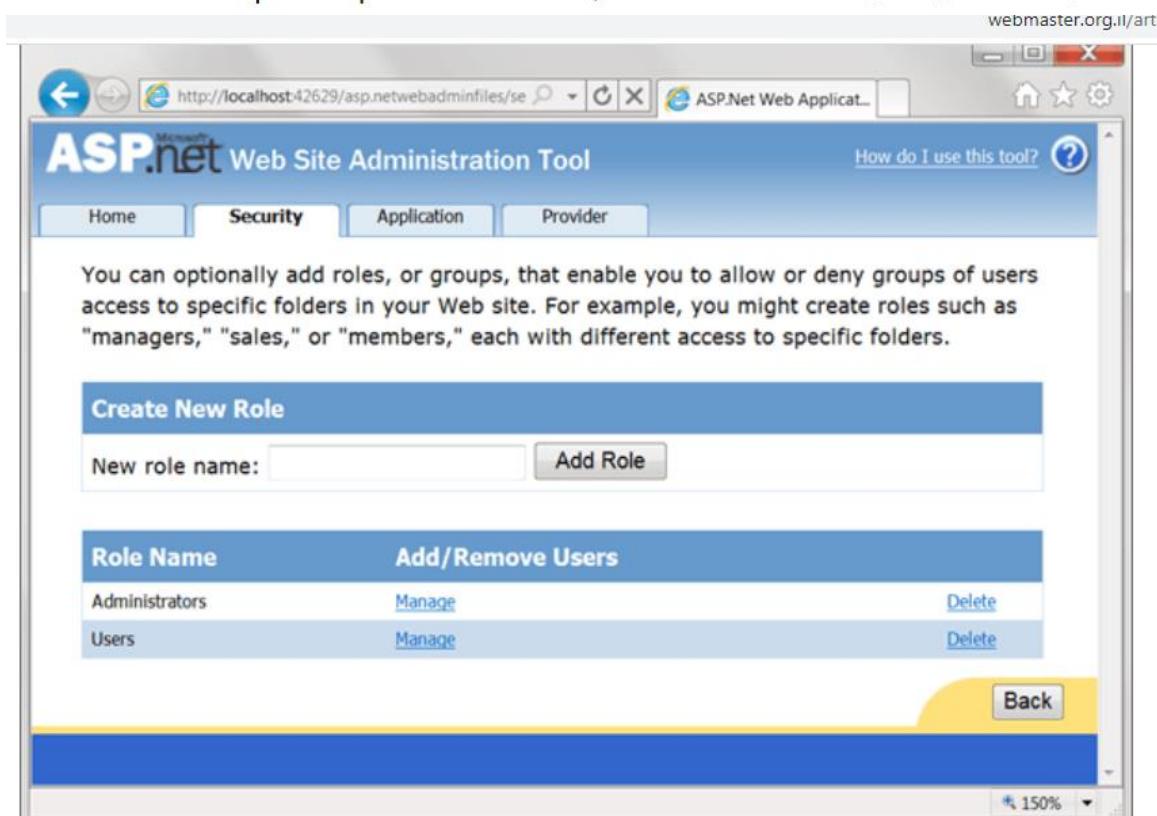
Click the links in the table to manage the settings for your application.

Users	Roles	Access Rules
Existing users: 0 <a href="#">Create user</a> <a href="#">Manage users</a>  <a href="#">Select authentication type</a>	Existing roles: 0 <a href="#">Disable Roles</a> <a href="#">Create or Manage roles</a>	<a href="#">Create access rules</a> <a href="#">Manage access rules</a>

65



נחזיר על אותה פעולה עבור Role בשם Users, בסופה של דבר יתקבל המסר הבא:



## הגדרת משתמשים (Users)

נלחץ על לשונית Security לחזרה למסך הראשי של Security:

You can use the Web Site Administration Tool to manage all the security settings for your application. You can set up users and passwords (authentication), create roles (groups of users), and create permissions (rules for controlling access to parts of your application).

By default, user information is stored in a Microsoft SQL Server Express database in the Data folder of your Web site. If you want to store user information in a different database, use the Provider tab to select a different provider.

[Use the security Setup Wizard to configure security step by step.](#)

Click the links in the table to manage the settings for your application.

Users	Roles	Access Rules
<b>Existing users:</b> 0 <a href="#">Create user</a> <a href="#">Manage users</a>  <a href="#">Select authentication type</a>	<b>Existing roles:</b> 2 <a href="#">Disable Roles</a> <a href="#">Create or Manage roles</a>	<a href="#">Create access rules</a> <a href="#">Manage access rules</a>

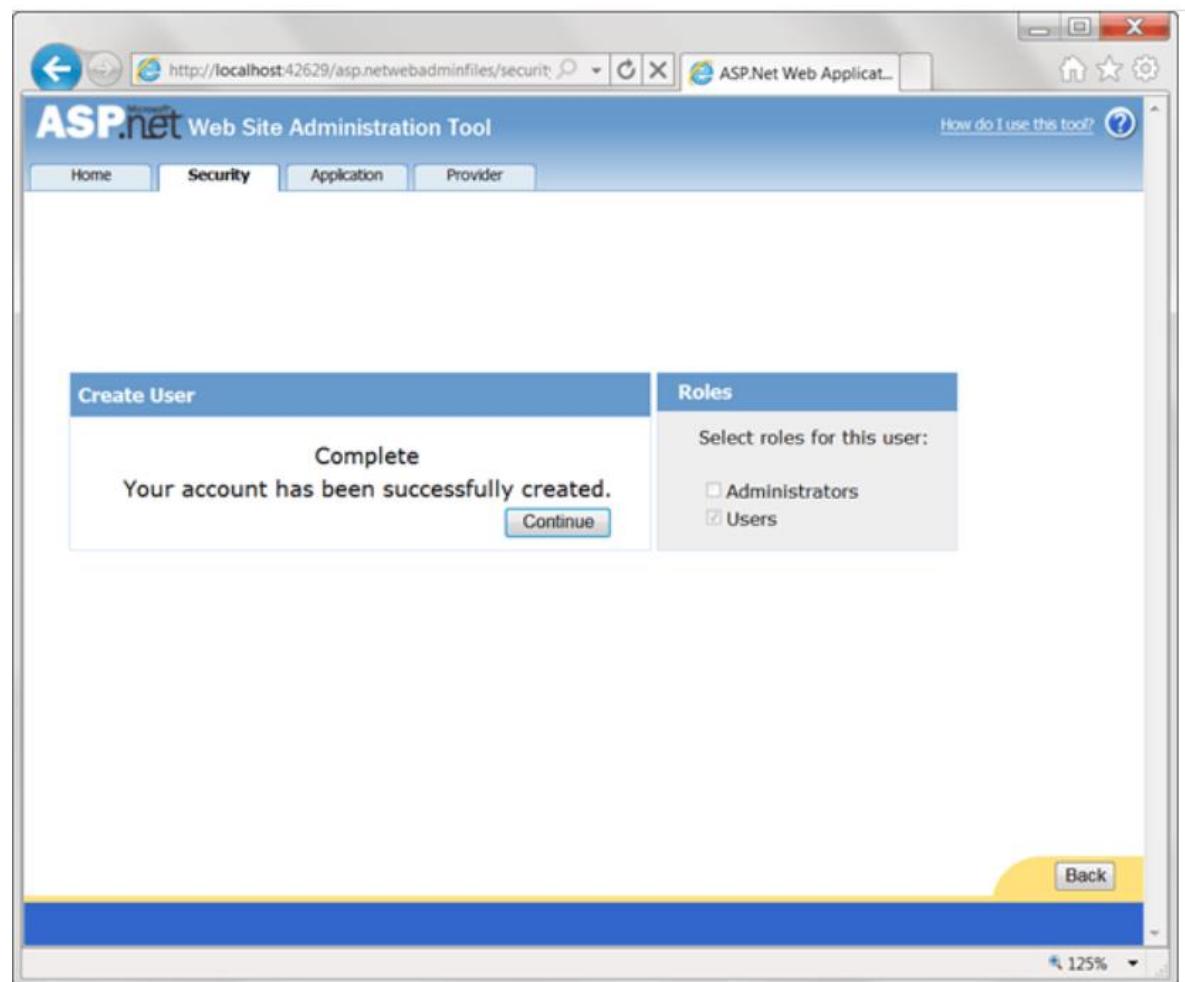
כעת נלחץ על ה קישור `Create user` ו נמלא את הפרטים בדף פרטי המשתמש שנפתח לנו:

Add a user by entering the user's ID, password, and e-mail address on this page.

Create User	Roles
<b>Sign Up for Your New Account</b> User Name: <input type="text" value="user1"/> Password: <input type="password" value="*****"/> Confirm Password: <input type="password" value="*****"/> E-mail: <input type="text" value="user1@webmaster.org.il"/> Security Question: What is your favorite site? Security Answer: <input type="text" value="webmaster.org.il"/>	Select roles for this user: <input type="checkbox"/> Administrators <input checked="" type="checkbox"/> Users

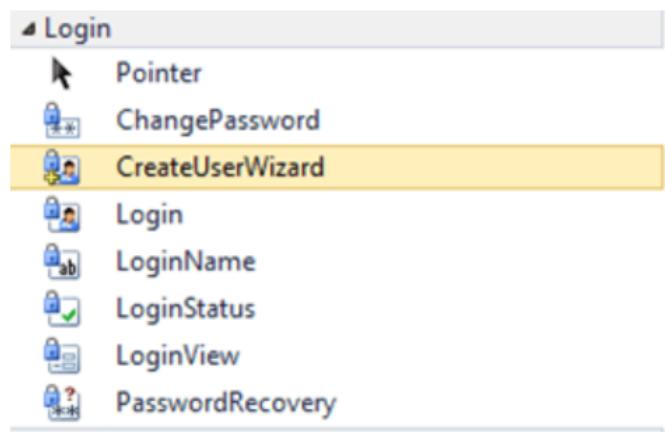
Active User

בסוף יצירת המשתמש תתקבל הודעה שהמשתמש נוצר בהצלחה:



לחיצה על Continue תחזיר אותנו לאותו המסך ונוכל לחזור על הפעולה בעבר מספר משתמשים.

ב – קיימים פקדים מיוחדים עבור כל הקשור ל – Login, רישום למערכת וכו'. במדריך זה נכיר את הפקדים השונים אשר נמצאים ב-Toolbox תחת קטגורית Login, מהו תפקידו ואיך לחברים אותו למערכת.



## הכנה

על מנת לעבוד עם פקדי Login, נוצרה להגדר את ה-Membership Provider כפ' שמוסבר במאמר [מדריך ASP.NET – אבטחה: הגדרת משתמשים וקבוצות](#).

## פקד Login

פקד זה משמש אותנו ליצור מסךהתחברות למערכת.

נוסף דף חדש למערכת בשם login.aspx, ונגרור את פקד ה – Login לתוך הדף.

ב – web.config נוסיף תחת ה-<system.web><authentication mode="Forms"> אלמנט בשם authentication שבאמצעותו נגדיר את אופן ההזדהות באפליקציה.

마וחר ואנו נרצה להזדהות באמצעות שם וסיסמה, נגדיר את ה-mode ל-Forms ונצין שיש להציג את דף ה-Login שבנו בכניסה למערכת, באופן הבא:

```
<system.web>
  <authentication mode="Forms">
    <forms loginUrl="login.aspx" name=".ASPFORMSAUTH" />
  </authentication>
</system.web>
```

על מנת למנוע מצב של משתמשים לא מזוהים (Anonymous) יכנסו למערכת נוסף תחת ה-`:authorization` במאוץ האלמנט (Authorization) במאוץ ה-`system.web` גם הגדרת הרשות (system.web).

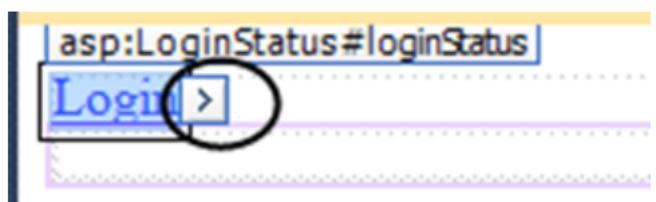
```
<authorization>
  <deny users="?" />
</authorization>
```

הגדרת deny מאפשרת לנו להגדיר אילו משתמשים אינם מורשים להכנס למערכת. סימן השאלה (?) מגדיר את המשתמש "אנונימי".

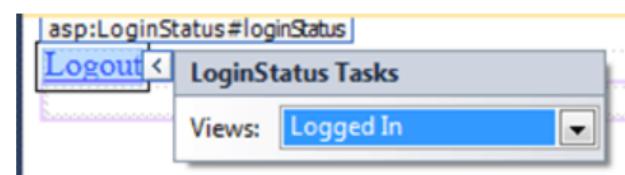
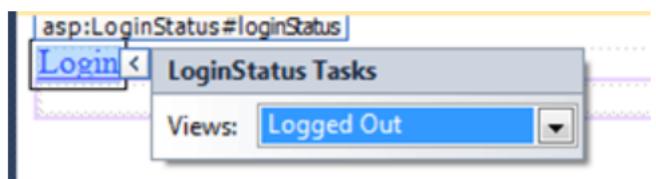
## פקד LoginStatus

הפקד זהה מציג קישור ל – Log in או Log out בהתאם להאם המשתמש מחובר או לא.

אפשר לראות את השינוי בכתיבה כבר ב – Designer כאשר לוחצים על החץ המסומן בעיגול:



כאשר אנחנו נמצאים במצב של Logged out הכתוב יהיה Log in , ואשר אנחנו במצב של in Logged : Logout הכתוב יהיה Log out .



את הכתובים האלה ניתן לשנות ע"י שינוי המאפיינים LogInText ו – LogOutText של הפקד . מאפיין חשוב נוסף הוא Url – LogOutPage – הדף אליו מופנה המשתמש לאחר Log out .

## פקד LoginView

פקד זה מאפשר לנו להציג כתוב שונה עבור משתמש מזוהה (Logged In) ועבור משתמש בלתי מזוהה (Anonymous) .

לצורך כך קיימים שני מאפיינים – LoggedInTemplate ו – AnonymousTemplate בהתקמה . נראה דוגמה לשימוש ב – LoginView :

```
<asp:LoginView ID="LoginView1" runat="server">
  <AnonymousTemplate>
    <asp:Label Text="Hello anonymous user"
      runat="server" ID="lblAnonymous" />
  </AnonymousTemplate>
  <LoggedInTemplate>
    <asp:Label ID="Label1" Text="Hello " runat="server" />
    <asp:LoginName runat="server" ID="ctrlLoginName" />
  </LoggedInTemplate>
</asp:LoginView>
```

במידה והמשתמש אינו מחובר נראה את הכתוב "Hello So". במידה והמשתמש מחובר (לצורך ההדגמה משתמש בשם "user1") נראה את הכתוב "Hello user1".

## פקד LoginName

פקד זה מראה לנו את שם המשתמש. ניתן לראות את השימוש בפקד בדוגמה של פקד ה- `UserName` לעיל. בכל מקום בו שמים את הפוך הוא מציג את שם המשתמש.

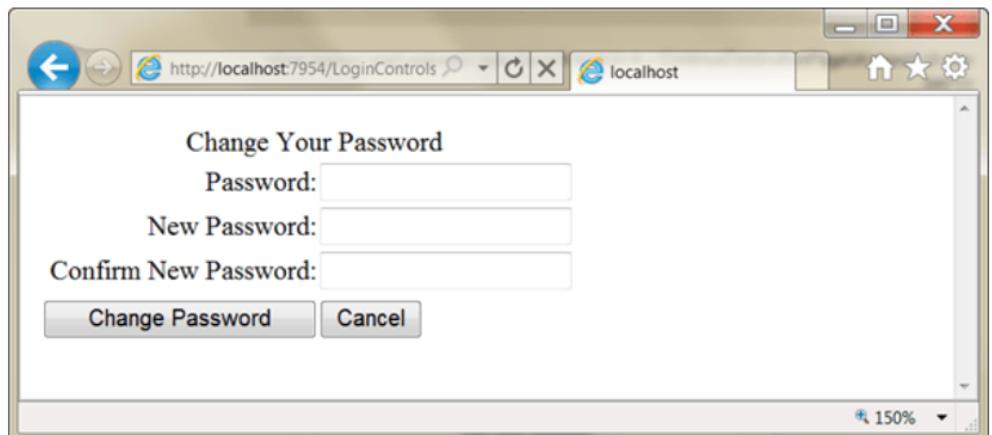
## פקד ChangePassword

פקד זה מאפשר לנו ליצור בקשות טופס לשינוי סיסמה. כל שעליינו לעשות הוא להוסיף דף לאתר `ChangePassword.aspx`

לדף זהו נוסיף את הפוך כך שיופיע לנו הכתוב הבא:

```
<asp:ChangePassword ID="ChangePassword1" runat="server">
</asp:ChangePassword>
```

כאשר הדף יעלה – יוצג למשתמש המסר הבא:



נוקור חלק מן המאפיינים:

שםבצע את פעולה של שינוי הסיסמה).

לשנות את המאפיין `ChangeButtonType` – ניתן לשיטים במקום הceptor קישור או תמונה – אך ניתן `Link` או `Image` – `ChangeButtonType`.

במקרה של Image יש לתת ק'ישור לתמונה במאפיין ChangePasswordButtonImageUrl, במקרה זה הערך המופיע במאפיין ChangePasswordButtonText ישמש כ – alt text של ConfirmNewPasswordLabelText.

71

. – הכיתוב ליד השדה לאישור הסיסמה החדשה ConfirmNewPasswordLabelText.

. – הכיתוב ליד השדה לכתיבה הסיסמה החדשה newPasswordLabelText.

. – הכיתוב ליד שדה הסיסמה (הישנה) PasswordLabelText.

– הודעה השגיאה אשר תופיע במקרה של שגיאה בשינוי הסיסמה ChangePasswordFailureText.

## פקד CreateUserWizard

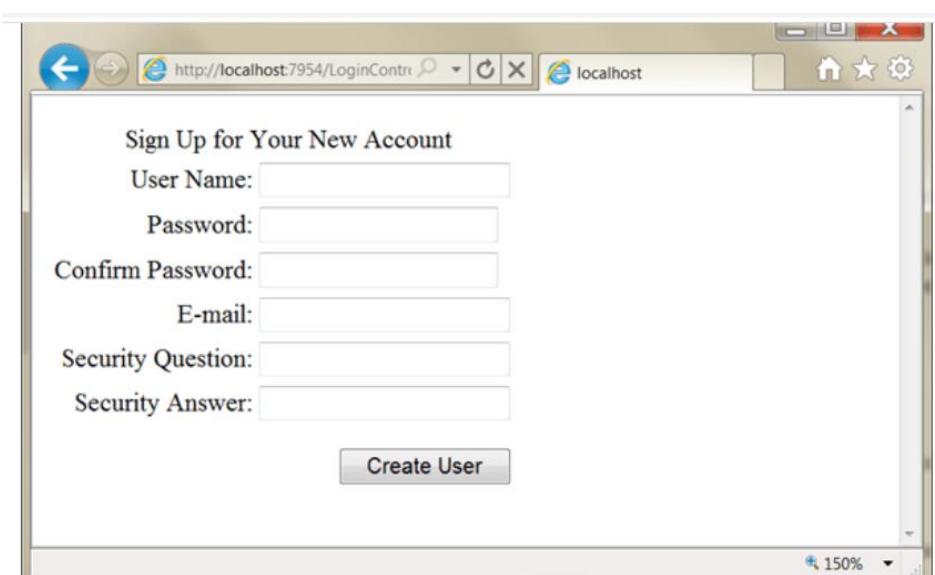
פקד זה מאפשר לנו להויסף דף רישום למערכת. הכיתוב עבור הפקד הוא:

```
<asp:CreateUserWizard ID="CreateUserWizard1" runat="server"
    ContinueDestinationPageUrl="~/home.aspx">
    <WizardSteps>
        <asp:CreateUserWizardStep ID="CreateUserWizardStep1"
            runat="server">
        </asp:CreateUserWizardStep>
        <asp:CompleteWizardStep ID="CompleteWizardStep1"
            runat="server">
        </asp:CompleteWizardStep>
    </WizardSteps>
</asp:CreateUserWizard>
```

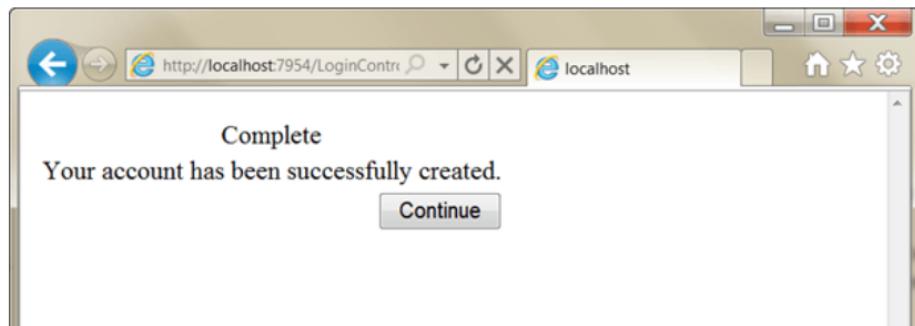
שימוש לב למאפיין ContinueDestinationPageUrl – זו הכתובת לדף אליו יפתח המשתמש לאחר השלמת תהליך הרישום.

לבד ממנו נראה כי האשפ (Wizard) זהה מחולק לשני שלבים (Steps) כאשר האחד הוא הרישום והשני הוא שלב ההשלמה. נוכל לראות כי מתקבלים שני מסכים (אחד עבור כל שלב).

שלב ההרשמה:



72



לחיצה על Continue תפנה אותנו לכתובת אותה הגדרנו במאפיין `ContinueDestinationPageUrl`.

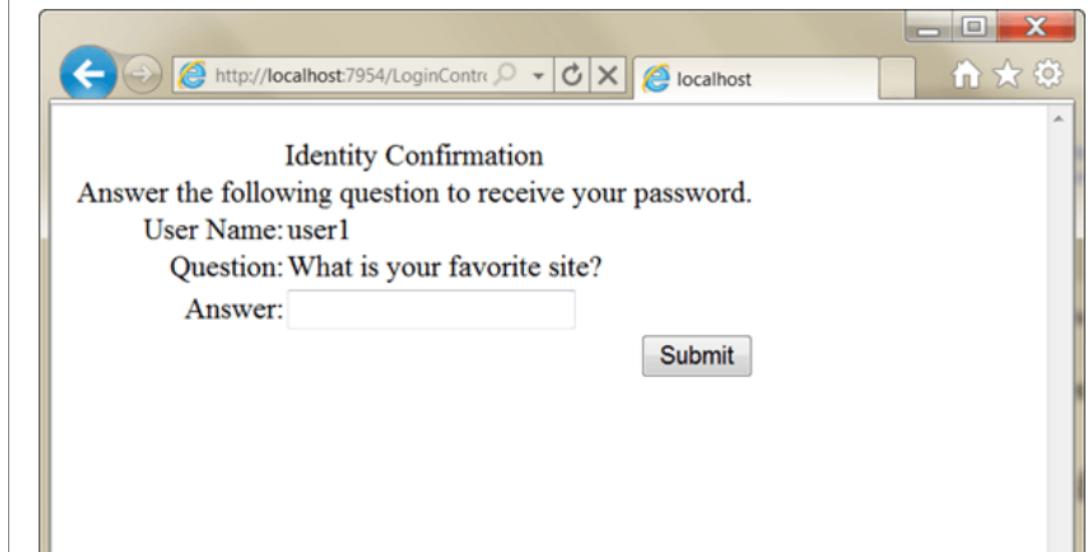
## פקד PasswordRecovery

שימוש בפקד זה מאפשר לנו ליצור דף, בו המשתמש נדרש לענות על שאלה בה בחר בעת שנרשם למערכת, והסיסמה תואפס ותשלח אליו למייל.

הכיתוב עבור הפקד זהה:

```
<asp:PasswordRecovery ID="Passwordrecovery1" runat="server" ,
```

כאשר משתמשים בפקד מתקין המסר הבא:



1. שרת SMTP – אותו יש להגדיר ב-:web.config

```
<system.net>
  <mailSettings>
    <smtp deliveryMethod="Network">
      <network host="yourSmtpServer"/>
    </smtp>
  </mailSettings>
</system.net>
```

2. כתובת השולח, אותה ניתן להגדיר בתוך הפקד במאפיין .MailDefinition.From



לינק 0

שיתוף

במדריך זה נראה כיצד אנחנו בודקים הרשאות של משתמש בקוד.

## הכנה

על מנת לעבוד עם הרשאות, נצטרך להגדיר את ה-Provider Membership כפ' שמוסבר במאמר [מדריך ASP.NET – אבטחה: הגדרת משתמשים וקבוצות](#).

נוסף למערכת שלנו דרך כל הניהול Role בשם Admin, כפ' שמוסבר במאמר [מדריך ASP.NET – אבטחה: הגדרת משתמשים וקבוצות](#).

נוסף דף למערכת שלנו ונקרא לו `PermissionCheck.aspx`. נוסף לדף זהה שני `Labels`, `lblIsAuthenticated` ולשני נקרא `lblIsAdmin`.

הכיתוב של הדף יהיה כר:

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="PermissionCheck.aspx.cs"
Inherits="Demos.PermissionCheck" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html >="http://www.w3.org/1999/xhtml">
<head id="Head1" runat="server">
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
        <div>
            <asp:Label runat="server" ID="lblIsAuthenticated" />
            <br />
            <asp:Label runat="server" ID="lblIsAdmin" />
        </div>
    </form>
</body>
</html>
```

הקוד הבא בודק האם המשתמש מחובר:

75

```
bool isAuthenticated = Page.User.Identity.IsAuthenticated;
```

בדיקה האם המשתמש הוא ב – Admin Role

הקוד הבא בודק האם המשתמש הוא ב – Admin Role

```
bool isAdmin = Page.User.IsInRole("Admin");
```

שיקוף התוצאות בדף

נכתב ב – Labels את תוצאות הבדיקות. ניקח את שני המשתנים הבוליאניים ולפיהם נקבע מהו הטקסט שיופיע ב – Labels.

```
if (isAdmin)
    lblIsAdmin.Text = "User is admin";
else
    lblIsAdmin.Text = "User is not admin";

if (isAuthenticated)
    lblIsAuthenticated.Text = "User is authenticated";
else
    lblIsAuthenticated.Text = "User is not authenticated";
```

נציג כעת את הקוד במלואו.

```

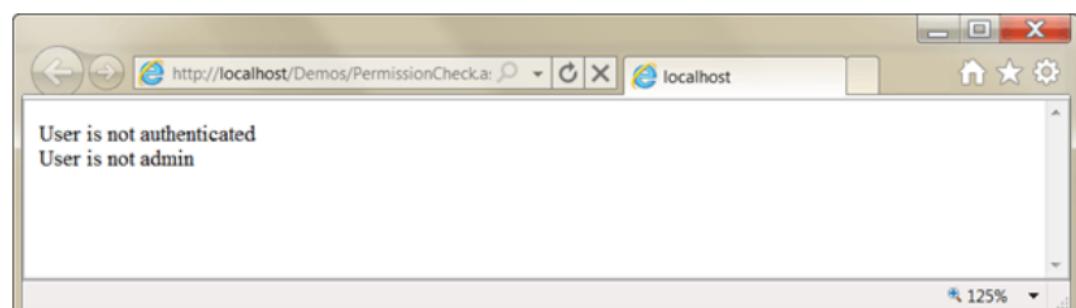
public partial class PermissionCheck : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        bool isAuthenticated = Page.User.Identity.IsAuthenticated;
        bool isAdmin = Page.User.IsInRole("Admin");

        if (isAdmin)
            lblIsAdmin.Text = "User is admin";
        else
            lblIsAdmin.Text = "User is not admin";

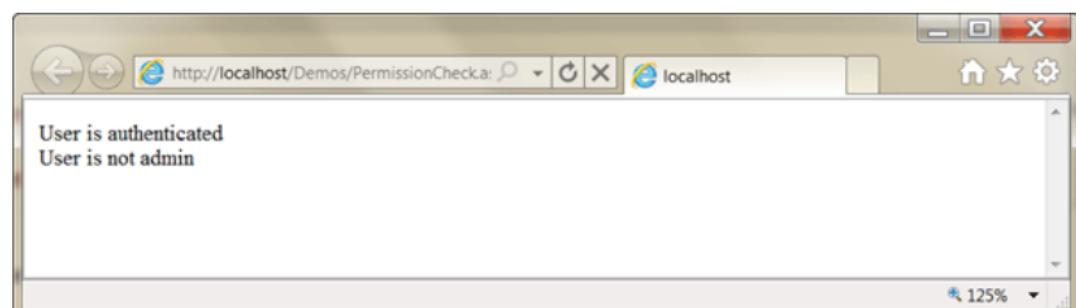
        if (isAuthenticated)
            lblIsAuthenticated.Text = "User is authenticated";
        else
            lblIsAuthenticated.Text = "User is not authenticated";
    }
}

```

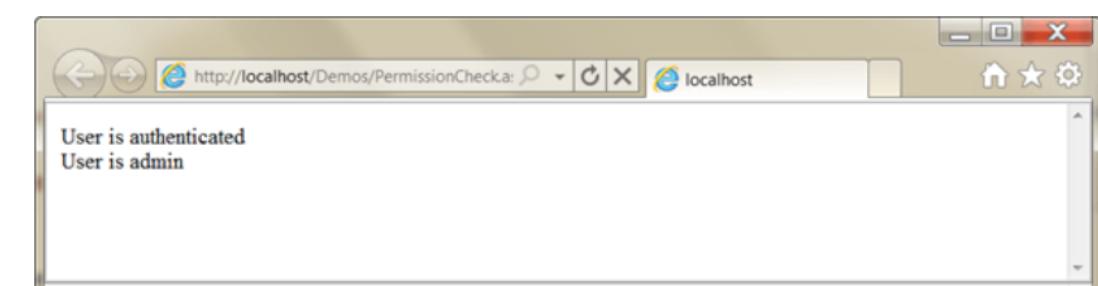
עבור משתמש שאינו מחובר יתקבל המסר הבא:



עבור משתמש מחובר שאינו Admin יתקבל המסר הבא:



עבור משתמש Admin יתקבל המסר הבא:



```

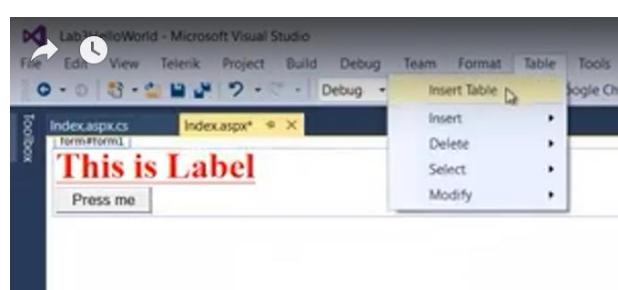
public partial class Index : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {

    }

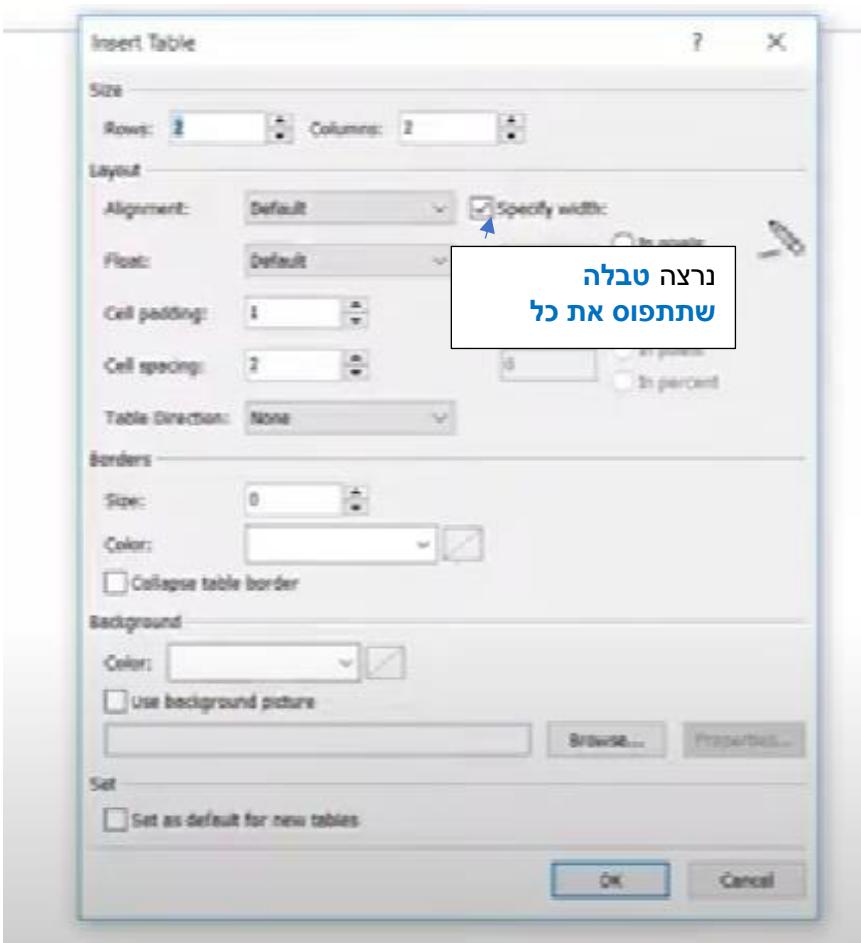
    protected void btnPressMe_Click(object sender, EventArgs e)
    {
        lblResult.Text = "This is a new message that I put in the Label!";
        lblResult.ForeColor = System.Drawing.Color.Blue;
    }
}

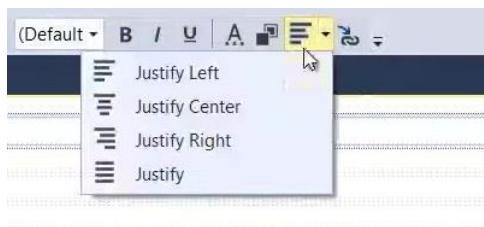
```

שינוי תכונות פקדים  
בלחיצת כפתור



הגדרת טבלה





מרכז אלמנטי הטבלה

```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
    <style type="text/css">
        .auto-style1 {
            width: 100%;
        }
        .auto-style2 {
            text-align: center;
        }
    </style>
</head>
<body>

<form id="form1" runat="server">

    <table class="auto-style1">
        <tr>
            <td class="auto-style2">
                <asp:Label ID="Label1" runat="server" Text="Label"></asp:Label>
            </td>
        </tr>
        <tr>
            <td>&ampnbsp</td>
        </tr>
        <tr>
            <td>&ampnbsp</td>
        </tr>
    </table>
</form>

```

**Calculator**

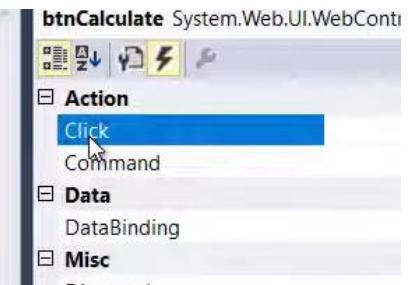
Enter first number

Enter second number

**asp:button#btnCalculate**

= **[lblResult]**

הוסף אירוע לחיצה  
1 פעמים על הפקד  
2 בלשונית Event פעמים על הפקד



```
public partial class Index : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {

    }

    protected void btnCalculate_Click(object sender, EventArgs e)
    {
        string num1 = txtNumber1.Text;
        string num2 = txtNumber2.Text;

        int n1 = int.Parse(num1);
        int n2 = int.Parse(num2);

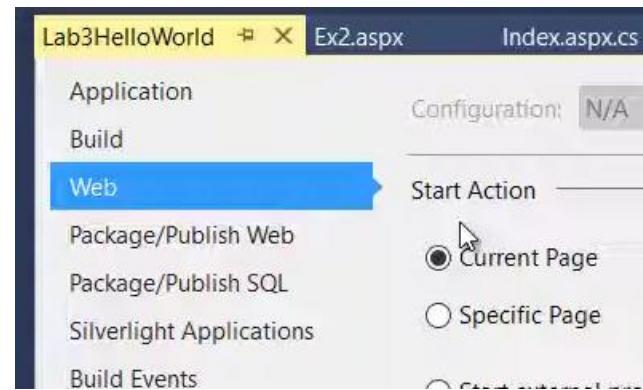
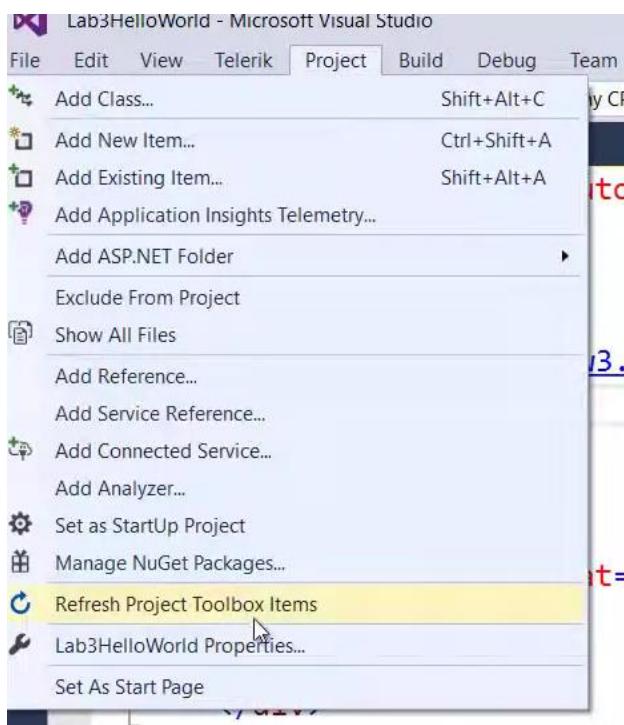
        int res = n1 + n2;

        lblResult.Text = res.ToString();
    }
}

lblResult.Text = (int.Parse(txtNumber1.Text) + int.Parse(txtNumber2.Text)).ToString();
```

חיבור  
מספרים

הגדרת איקן הריצה  
תהייה

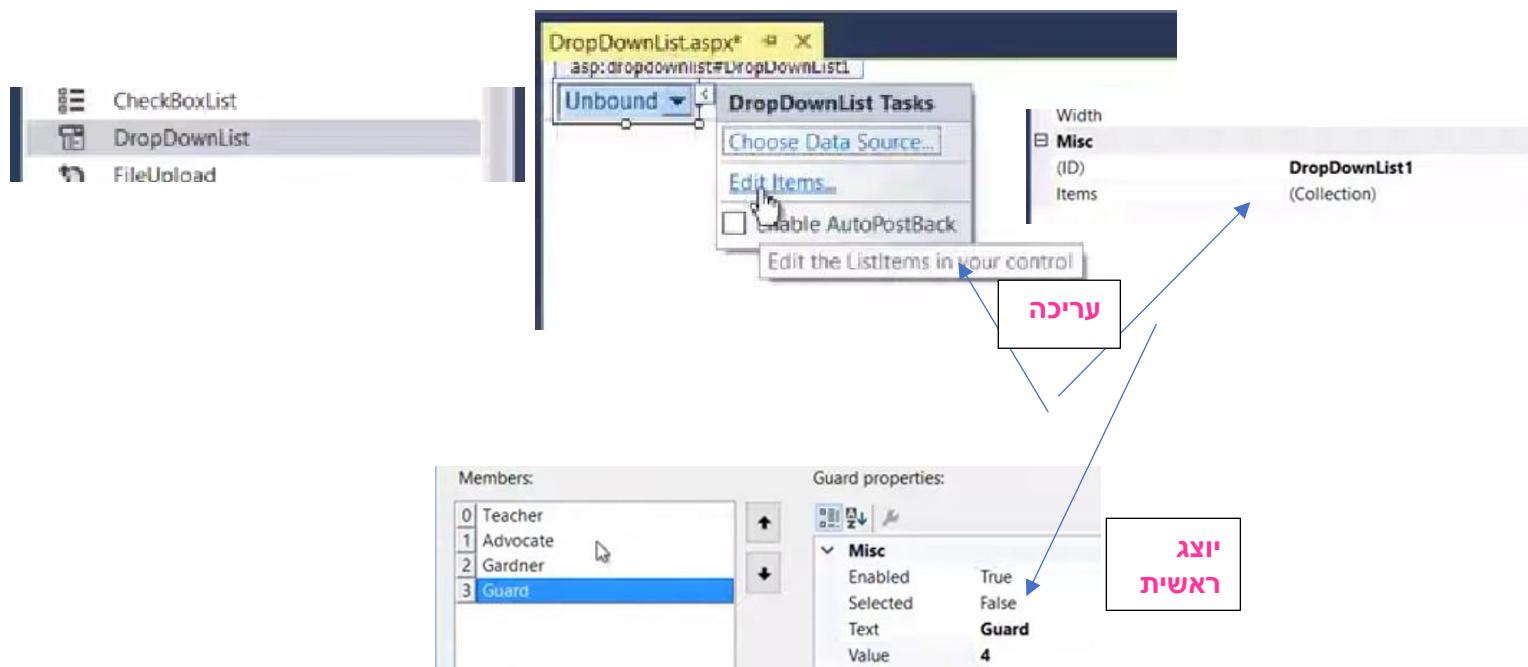


```
private void Calculate(int operand)
{
    int num1 = int.Parse(txtNum1.Text);
    int num2 = int.Parse(txtNum2.Text);
    switch (operand)
    {
        case 1: //+
            txtResult.Text = (num1 + num2).ToString();
            break;
        case 2://-
            txtResult.Text = (num1 - num2).ToString();
            break;
        case 3://*
            txtResult.Text = (num1 * num2).ToString();
            break;
        case 4:///
            txtResult.Text = ((double)num1 / (double)num2).ToString();
            break;
    }
}

protected void btnPlus_Click(object sender, EventArgs e)
{
    Calculate(1);
}
```

הגדרת פונקציית לחיצה  
שתקרא לפונקציה שנגידיר

הגדרת **ילדי** רשימה נפתח



```

protected void btnShow_Click(object sender, EventArgs e)
{
    lblJobTitle.Text = "Selected Text: " + ddlJobTitles.SelectedItem.Text +
        " Selected Value: " + ddlJobTitles.SelectedItem.Value;

    ddlJobTitles.SelectedIndex = 0;
}

```

גישה לתוכנות

```

namespace Lab3HelloWorld
{
    public partial class Ex2 : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {
            if (!Page.IsPostBack) < 13,314ms elapsed
            {
                txtNum1.Text = "";
                txtNum2.Text = "";
                txtResult.Text = "";
                dd1Operand.SelectedIndex = 0;
            }
        }
    }
}

```

איפוס הדף בפעם  
הראשונה

GroupName  
SkinID

**קיבוץ כפטורים**  
**לסוג**

**Operands**

```
protected void btnCalculate_Click(object sender, EventArgs e)
{
    if (rbSum.Checked) [
        Calculate(1);
    else if (rbSub.Checked)
        Calculate(2);
    else if (rbMul.Checked)
        Calculate(3);
    else
        Calculate(4);
}
```

**בדיקה לחוץ**

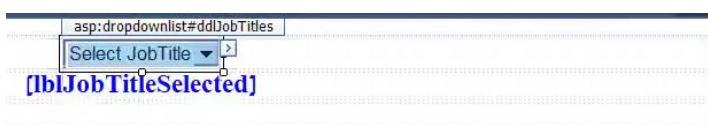
```
protected void btnCalculate_Click(object sender, EventArgs e)
{
    double basicArnona = double.Parse(txtArnona.Text);
    double discount = 0;

    if (cbSeniority.Checked)
        discount += 0.1;
    if (cbStudent.Checked)[
        discount += 0.2;
    if (cbSoldieer.Checked)
        discount += 0.3;

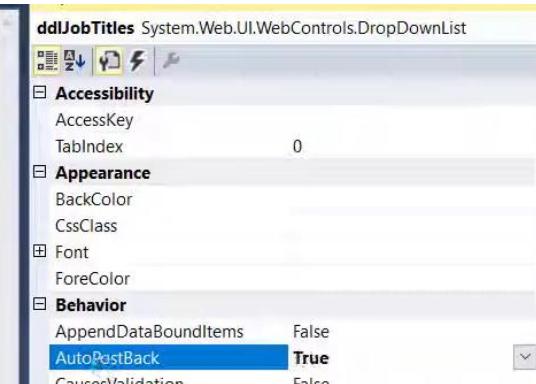
    if (discount == 0)
        lblResult.Text = basicArnona.ToString();
    else
        basicArnona *= (1 - discount);

    lblResult.Text = basicArnona.ToString();
}
```

**סימון כפטורים**  
**והנחה בהתאם**



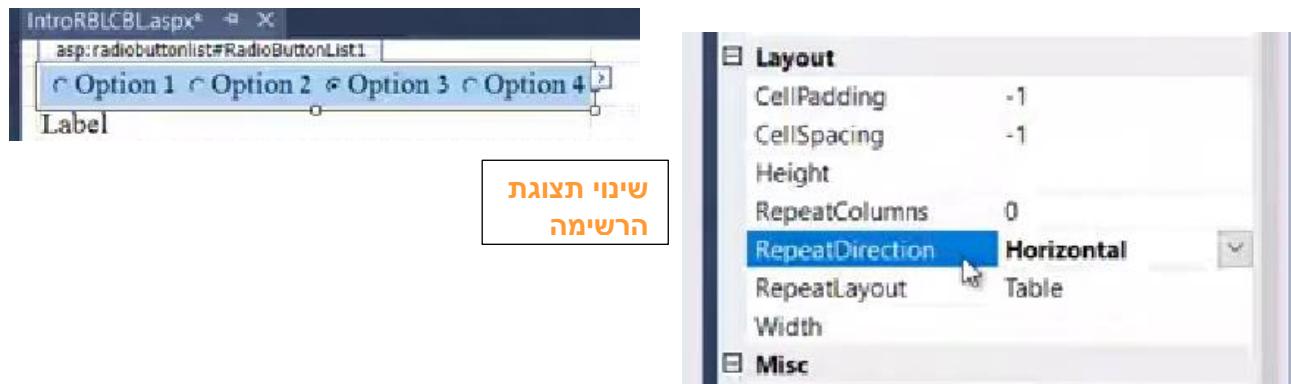
**נזהיר שבעת שינוי / בחירת ערך רשימה**  
**תהייה פניה לשרת**



```
protected void ddlJobTitles_SelectedIndexChanged(object sender, EventArgs e)
{
    lblJobTitleSelected.Text = ddlJobTitles.SelectedItem.Text;
}
```

## Web Controls -CheckBoxList, RadioButtonList

- ▶ DropDownList (RadioButtonList) RBL ו (CheckBoxList) CBL דומים מאוד ל CBL
- ▶ ב RBL אנו בונים מערך של Items שמתוכם נוכל לבחור רק אחד. לכל Item קיים .Value ו Text
- ▶ ב CBL אנו בונים מערך של Items שמתוכם נוכל לבחור מספר איברים (0 או יותר). לכל Item קיים Value ו Text
- ▶ כל RBL עומד בפני עצמו, ככלומר אם קיימים לנו 2 RBL נוכל לבחור איבר אחד מכל RBL



```

protected void btnShow_Click(object sender, EventArgs e)
{
    lblSelected.Text = "Text:" + rblOptions.SelectedItem.Text + " Value:" + rblOptions
}

protected void btnShow_Click(object sender, EventArgs e)
{
    lblSelected.Text = "RadioButtonList - Text:" + rblOptions.SelectedItem.Text + " Val

    string items = "";
    for (int i = 0; i < cb1Classes.Items.Count; i++)
        if (cb1Classes.Items[i].Selected)
            items += " Text" + cb1Classes.Items[i].Text + " Value" + cb1Classes.Items[i]

    foreach(ListItem li in cb1Classes.Items)
        if(li.Selected)
            items += " CheckBoxList - Text:" + li.Text + " Value: " + li.Value;

    lblSelected.Text += items;
}

```

מעבר על איברי הרשימה כפטור  
הבחירה

```

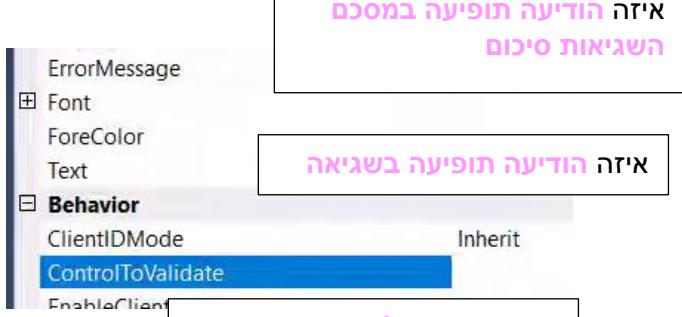
protected void btnAddWorker_Click(object sender, EventArgs e)
{
    if(txtFirstName.Text.Trim().Length==0)
    {
        lblErrors.Text = "Please enter first name";
        return;
    }
    if (txtLastName.Text.Trim().Length == 0)
    {
        lblErrors.Text = "Please enter last name";
        return;
    }
    if (txtIdNum.Text.Trim().Length == 0)
    {
        lblErrors.Text = "Please enter id number";
        return;
    }
    if(ddlStatus.SelectedIndex==0)
    {
        lblErrors.Text = "Please select marital status";
        return;
    }
    if(txtIdNum.Text.Trim().Length>9)
    {
        lblErrors.Text = "Please enter 9 digits in id number";
        return;
    }
}

```

ולידיציות

## RequiredFieldValidator

- ▶ פקד וולידציה זה מודיא כי החן ערך בשדה (טקסט או תיבת גלילה).
- ▶ במידה ולא החן ערך (או נבחר ערך מסוים בתיבת הגלילה) ניתן להציג הודעה שגיאה למשתמש ללא צורך בפנינה לשרת.
- ▶ מאפיינים חשובים שיש להתייחס בפקד זה:
  - ▶ ControlToValidate - על איזה פקד בדף (תיבת טקסט/תיבת גלילה) על פקד הוולידציה "לשמור"
  - ▶ Text - הודעה השגיאה שתופיע
  - ▶ ValidationSummary - הודעה השגיאה שתופיע בפקד ה ErrorMessage



איזה הודעה תופיע במסכים  
השגיאות סיכום

איזה הודעה תופיע בשגיאה

איזה פקד יפעיל הבדיקה

EnableViewState	True
InitialValue	-1
SetFocusOnError	False
SkinID	

איזה ערך לא תקין

DisplayMode  
Font

## ValidationSummary

EnableTheming	True	יצג חלון קופץ
EnableViewState	True	
ShowMessageBox	True	יצג באתר
ShowModelStateErrors	True	
ShowSummary	False	
ShowValidationErrors	True	

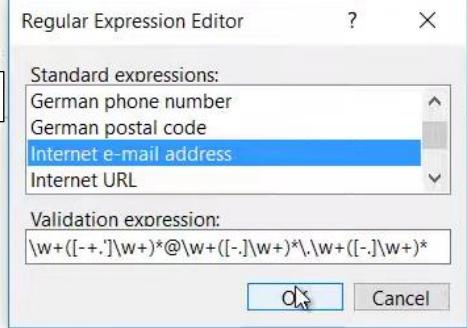
בדיקה ביתוי

מופעל רק אם יש על מה לבדוק

## RegularExpressionValidator

ValidateRequestMode	Inherit
ValidationExpression	[0-9]{9}

רישומת ביתויים רגולריים



## CompareValidator

Behavior	
ClientIDMode	Inherit
ControlToCompare	
ControlToValidate	
CultureInvariantValues	False
EnableClientScript	True
Enabled	True
EnableTheming	True
EnableViewState	True
Operator	Equal
SetFocusOnError	False
SkinID	
ToolTip	
Type	String

איזה פקד מקור

איזה פקד יעד

מה סוג ההשוואה

מה סוג הנתונים

## CompareValidator

- ▶ פקק וולידציה זה מבצע השוואת השווות בין שני ערכים (בדרך כלל ב TextBox)
- ▶ ההשוואה מתבצעת רק אם יש ערך במאפיין .ControlToValidate
- ▶ ההשוואה מתבצעת עם הפוך שהוגדר במאפיין ControlToCompare
- ▶ ניתן לבחור את סוג ההשוואה (שווה, גדול, קטן...) במאפיין Operator
- ▶ יש להגדיר את סוג הנתון שמחוץ בתיבות הטקסט להשוואה (ברירת מחדל - String) במאפיין Type

EnableTheming	True
EnableViewState	True
InitialValue	
SetFocusOnError	True
Text	

הבא ת סמן  
לפקק השגוי

Text	Clear Form
<b>Behavior</b>	
CausesValidation	<b>False</b>
ClientIDMode	Inherit
CommandArgument	

אישור לפקק שיפעל  
למרות שיש שגיאה

Password	<input type="text"/>	asp:requiredfield... #RequiredField...	ToolTip
Password Again	<input type="text"/>	Password is mandatory field asp:comparevalidator #Comparevalida...	ValidateRequestMode
	<input type="button" value="Save"/>	Passwords dont match	ValidationGroup

חלוקת פקי הבדיקה  
לקבוצות

```

protected void btnGoto_Click1(object sender, EventArgs e)
{
    //Method 1:
    Server.Transfer("SecondPage.aspx")מעבר לעמוד אחר
}

//Method 2:
Response.Redirect("SecondPage.aspx");

```

```
protected void btnGoto_Click1(object sender, EventArgs e)
```

```
{
    //Method 1:
    //Server.Transfer("SecondPage.aspx");
```

```
Session["userName"] = txtUserName.Text;
```

```
}

//Method 2:
Response.Redirect("SecondPage.aspx");
}
```

מעבר נתונים

```

namespace FormsValidationAndSession
{
    public partial class SecondPage : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {
            if (!Page.IsPostBack)
                lblUserName.Text = Session["userName"].ToString();
        }
    }
}
```

```

protected void Page_Load(object sender, EventArgs e)
{
    if (!Page.IsPostBack)
    {
        //lblUserName.Text = ((int)Session["age"]).ToString();

        string[] items = Session["cart"].ToString().Split('#');
        for (int i = 0; i < items.Length - 1; i++) ≤ 1ms elapsed
            lblItems.Text += items[i] + "<br/>";
    }
}

```

**רשימת קניות**

```

protected void btnAdd_Click(object sender, EventArgs e)
{
    Session["cart"] = Session["cart"] + ddlItems.SelectedItem.Text + "#";
}

```

## שמירת מידע בכל דפי האתר לכל משתמש Session

- ▶ לכל משתמש הולש לאתר נוצר אובייקט הנקרא **Session**. באובייקט זה אנו יכולים גם לשמר מידע אישי על המשתמש, למשל מידע שהוא רוצים לשמור במעבר בין דפים.
- ▶ באובייקט אנו משתמשים בשיטה של **Key/value**.
- ▶ באובייקט **Session** ניתן לאחסן נתונים מסוגים שונים: **string, int, bool** ו-**אFILE** מחלקות **Dictionary**.
- ▶ דוגמה להשמה וקריאה:
- ▶ **Session[“name”] = “Tiferet”**
- ▶ **string name = Session[“name”].ToString()**

## שמירת מידע בכל דפי האתר לכל משתמש Session

- ▶ לכל משתמש הולש לאתר נוצר אובייקט הנקרא `Session`. באובייקט זה אנו יכולים גם לשמר מידע אישי על המשתמש, למשל מידע שהוא רוצה לשמור במעבר בין דפים.
- ▶ באובייקט אנו משתמשים בshiota של `Key/value`
- ▶ באובייקט `Session` ניתן לאחסן נתונים מסוגים שונים: `string`, `int`, `bool` ו-`file` מחלוקת.
- ▶ דוגמה להשנה וקריאה:  
`Session[“name”] = “Tiferet”`
- ▶  
`string name = Session[“name”].ToString()`

## שמירת מידע בכל דפי האתר לכל משתמש Session

לכל משתמש הולש לאתר נוצר אובייקט הנקרא `Session`. באובייקט זה אנו יכולים גם לשמר מידע אישי על המשתמש, למשל מידע שאנו רוצים לשמור בעבר בין דפים.

באובייקט אנו משתמשים בשיטה של `Key/value`

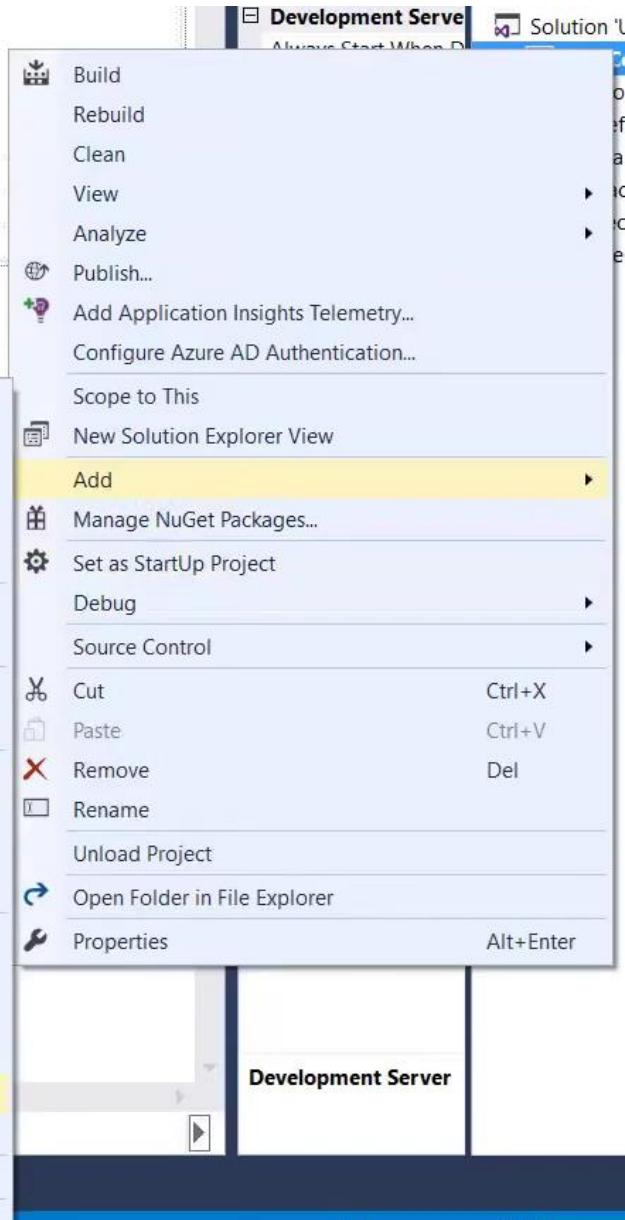
באובייקט `Session` ניתן לאחסן נתונים מסווגים שונים: `string`, `int`, `bool` וafilו מחלקות. דוגמה להשמה וקריאה:

```
Session[“name”] = “Tiferet”
```

```
string name = Session[“name”].ToString()
```

Name	<input type="text"/>
Number	<input type="text"/>
	<input type="button" value="Add"/> <input type="button" value="Clear"/>

**אנו עושים**

**אנו עושים****Add New Item - UserControls**

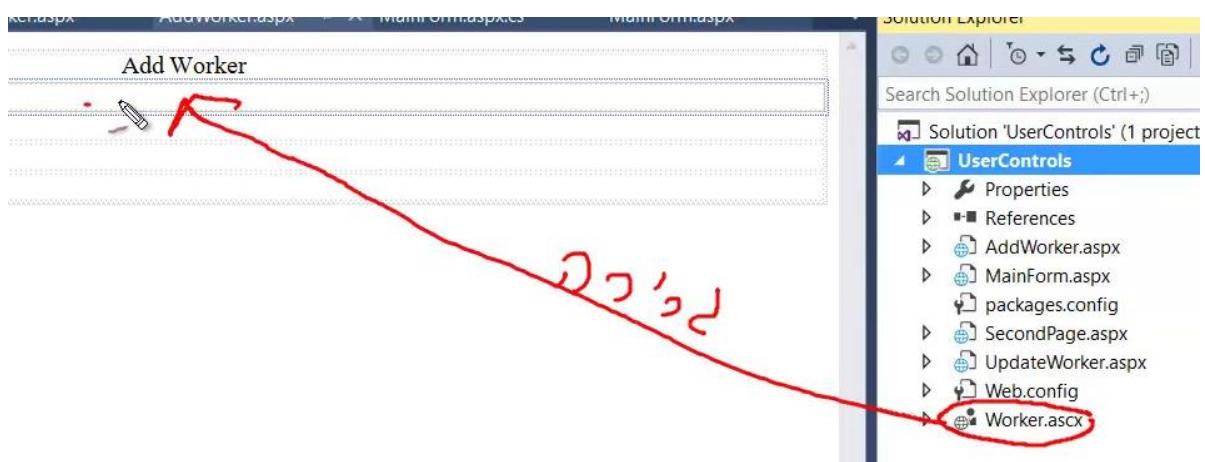
**Installed**

- Visual C#
  - Code
  - Data
  - General
- Web
  - General
  - Markup
  - MVC
  - Razor
  - Scripts
  - SignalR
  - Web API
  - Web Forms**
- Windows Forms
- WPF
- Reporting
- Silverlight

Sort by: Default

	Web Form	Visual C#
	Web Form with Master Page	Visual C#
	Web Forms Master Page	Visual C#
	Web Forms Master Page (Nested)	Visual C#
	Web Forms Server Control	Visual C#
	Web Forms Skin File	Visual C#
	<b>Web Forms User Control</b>	Visual C#

*WebForms → .ASPX  
User Controls → .ASCX*



```

</tr>
<tr>
    <td class="auto-style2">
        <uc1:Worker ID="Worker1" runat="server" />
    </td>
</tr>
<tr>

```

The screenshot shows the code editor with the following content:

```

<%@ Register src="Worker.ascx" tagname="Worker" tagprefix="uc1" %>
<!DOCTYPE html>

```

A red oval highlights the registration directive line: `<%@ Register src="Worker.ascx" tagname="Worker" tagprefix="uc1" %>`. A red circle is also drawn around the 'uc1:Worker' tag in the previous code snippet.

```

public void ChangeNameToAdd()
{
    btnAdd.Text = "Add Worker";
}

public void ChangeNameToUpdate()
{
    btnAdd.Text = "Update Worker";
}

```

נגידר פונקציות באב הקונטROL ציבורי  
שנוכל לגשת אליהו

```

namespace UserControls
{
    public partial class UpdateWorker : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {
            if (!Page.IsPostBack)
                Worker1.ChangeNameToUpdate();
        }
    }
}

```



The screenshot shows the Visual Studio Properties window for a control named 'Worker1'. The 'Behavior' section contains the following properties:

ClientIDMode	Inherit
EnableTheming	True
EnableViewState	True
ValidateRequestMode	Inherit
ViewStateMode	Inherit
Visible	True

The 'Misc' section contains the following properties:

(ID)	Worker1
runat	server

The screenshot shows a Microsoft Visual Studio IDE interface with the following details:

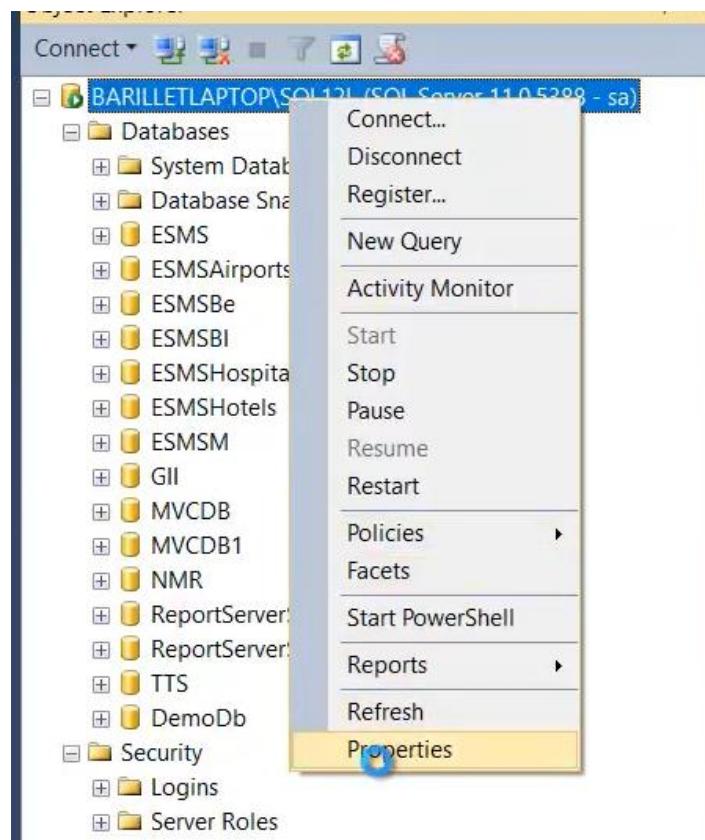
- Project Explorer:** Shows files like px.cs, UpdateWorker.aspx, AddWorker.aspx, Worker.ascx.cs\*, and Worker.aspx.
- Toolbox:** Shows icons for UserControls.Worker and btnAdd.
- Code Editor:** Displays the Worker.ascx.cs code. The code includes logic for handling button clicks and loading details based on an ID number.

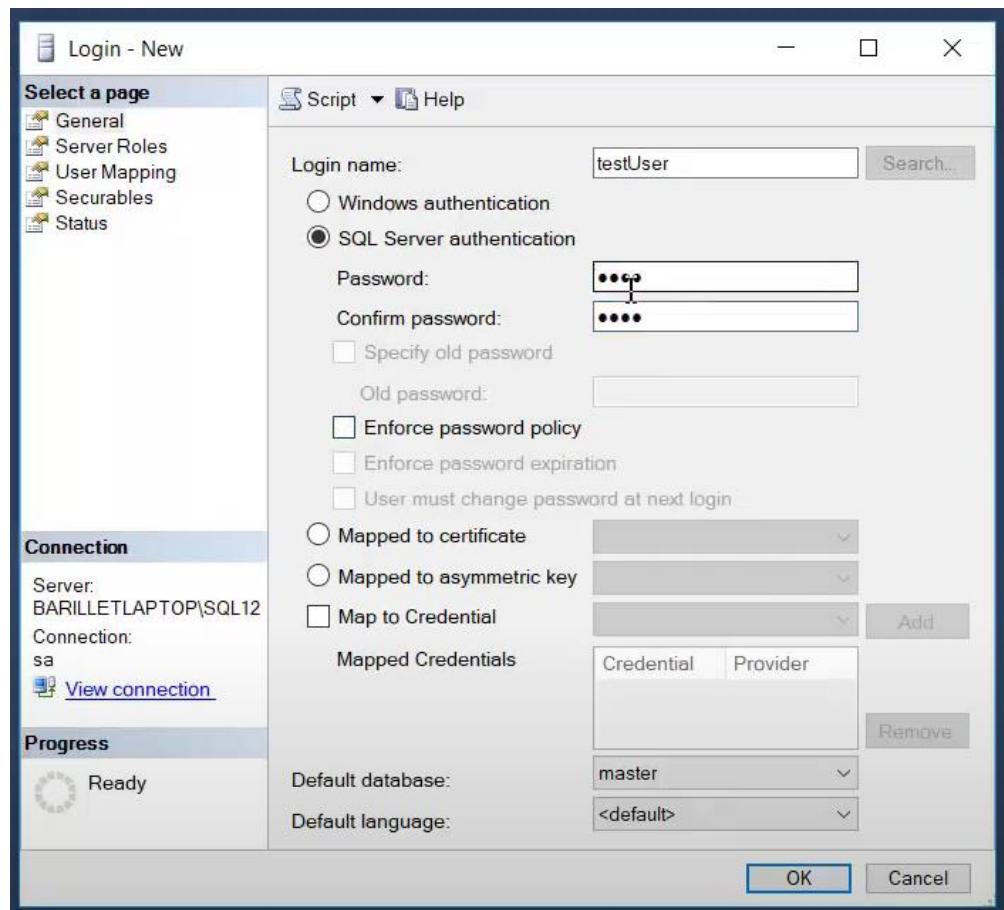
```
protected void Page_Load(object sender, EventArgs e)
{
}

protected void btnAdd_Click(object sender, EventArgs e)
{
    if(btnAdd.Text== "Add Worker")
    {
        lblPressed.Text = "Add worker presses!";
        //Add code to addd worker
    }
    else if(btnAdd.Text=="Update Worker")
    {
        lblPressed.Text = "Add worker presses!";
        //Add code to addd worker
    }
}

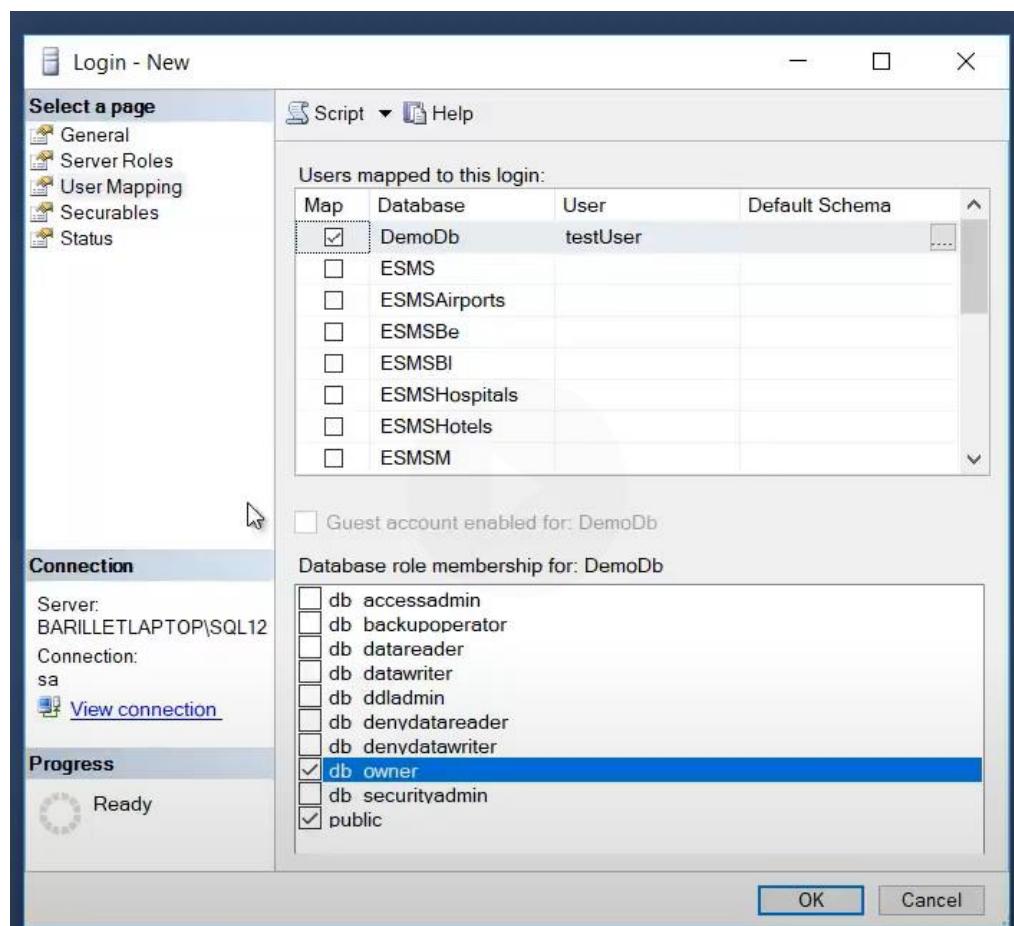
public void LoadDetails(string idNum)
{
    if(idNum=="123456789")
    {
        txtFN.Text = "Moshe";
        txtID.Text = idNum;
        txtLN.Text = "Cohen";
    }
    else if (idNum == "222222222")
    {
        txtFN.Text = "Shaira";
        txtID.Text = idNum;
        txtLN.Text = "Ben Ami";
    }
    else if (idNum == "777777777")
    {
        txtFN.Text = "Yarden";
        txtID.Text = idNum;
        txtLN.Text = "Cohva";
    }
}
```

UpdateWorker.aspx.cs\* UpdateWorker.aspx AddWorker.aspx Worker.aspx\* Worker.aspx  
UserControls UserControls.UpdateWorker btnLoad\_Click()  
using System;  
using System.Collections.Generic;  
using System.Linq;  
using System.Web;  
using System.Web.UI;  
using System.Web.UI.WebControls;  
namespace UserControls  
{  
 public partial class UpdateWorker : System.Web.UI.Page  
 {  
 protected void Page\_Load(object sender, EventArgs e)  
 {  
 if (!Page.IsPostBack)  
 Worker1.ChangeNameToUpdate();  
 }  
 protected void btnLoad\_Click(object sender, EventArgs e)  
 {  
 Worker1.LoadDetails(txtIdSearch.Text);  
 }  
 }  
}





הרשאות



הוסף מפתח רץ ב 1

Collation	<database default>
Computed Column Specification	
Condensed Data Type	int
Description	
Deterministic	Yes
DTS-published	No
Full-text Specification	No
Has Non-SQL Server Subscriber	No
Identity Specification	Yes
(Is Identity)	Yes
Identity Increment	1
Identity Seed	1

BARILLETLAPTOP\S...oDb - dbo.Table\_1\* SQLQuery1.sql - BA...LDemoDb (sa (53))

Column Name	Data Type	Allow Nulls
pkid	int	<input type="checkbox"/>
		<input type="checkbox"/>
		<input type="checkbox"/>
		<input checked="" type="checkbox"/>
		<input checked="" type="checkbox"/>
		<input type="checkbox"/>
		<input type="checkbox"/>
		<input type="checkbox"/>

Set Primary Key

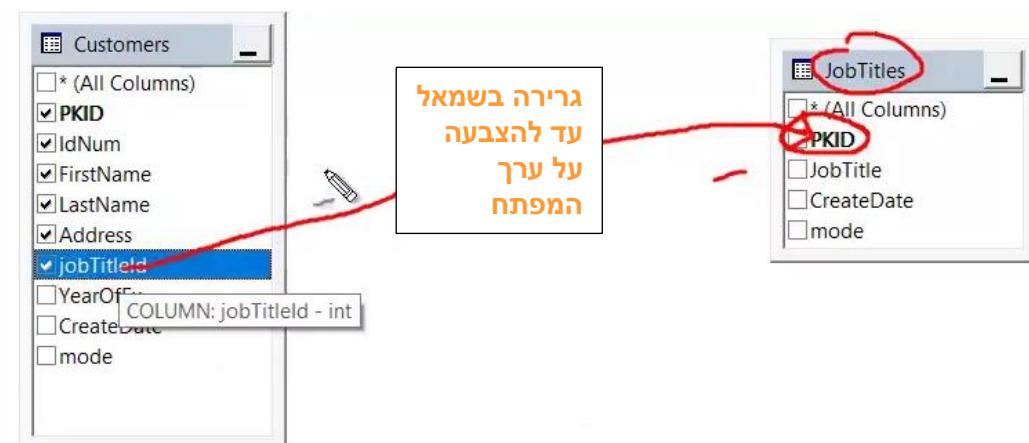
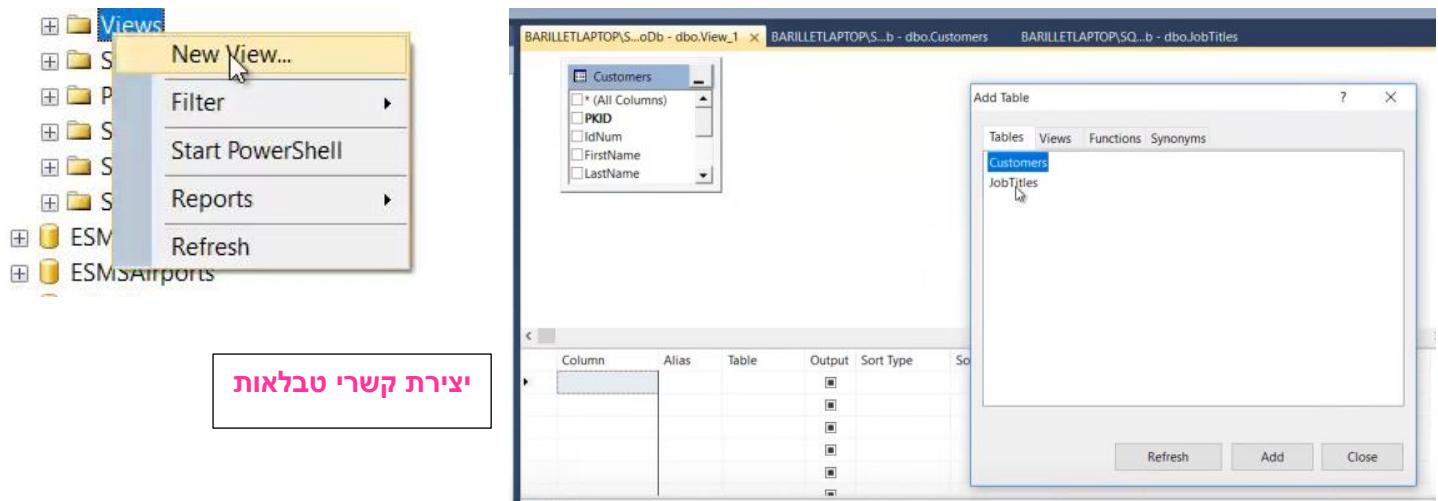
- Insert Column
- Delete Column
- Relationships...
- Indexes/Keys...
- Fulltext Index...
- XML Indexes...
- Check Constraints...
- Spatial Indexes...
- Generate Change Script...
- Properties

Alt+Enter

(General)

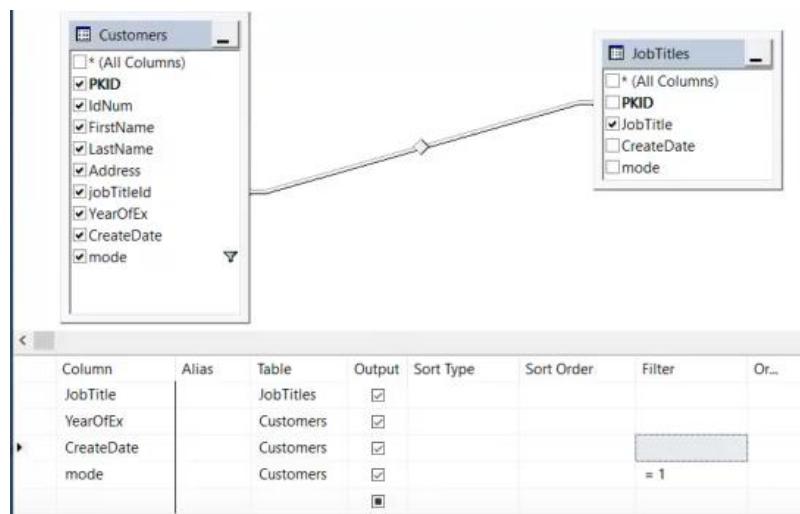
(Name)	CreateDate
Allow Nulls	No
Data Type	date
Default Value or Binding	getdate()

Table Designer



Column	Alias	Table	Output	Sort Type	Sort Order	Filter	Or...
PKID		Customers	<input checked="" type="checkbox"/>				
IdNum		Customers	<input checked="" type="checkbox"/>				
FirstName		Customers	<input checked="" type="checkbox"/>				
LastName		Customers	<input checked="" type="checkbox"/>				
Address		Customers	<input checked="" type="checkbox"/>				
jobTitleId		Customers	<input checked="" type="checkbox"/>				
YearOfEx		Customers	<input checked="" type="checkbox"/>				
CreateDate		Customers	<input checked="" type="checkbox"/>				
mode		Customers	<input checked="" type="checkbox"/>				

```
:CT dbo.Customers.PKID, dbo.Customers.IdNum, dbo.Customers.FirstName, dbo.Customers.LastName, dbo.Customers.A
M   dbo.Customers CROSS JOIN
      dbo.JobTitles
```



```
SELECT dbo.Customers.PKID, dbo.Customers.IdNum, dbo.Customers.FirstName, dbo.Customers.LastName, dbo.Customers.A
```

Screenshot of Microsoft SQL Server Management Studio (SSMS) showing the creation of a view.

The title bar shows three tabs: "BARI II FTI LAPTOP\S...nDb - dbo.View\_1\*" (active), "BARI II FTI LAPTOP\S...b - dbo.Customers", and "BARI II FTI LAPTOP\S...b - dbo.JobTitles".

The left pane displays the "Customers" table with columns: PKID, IdNum, FirstName, LastName, Address, JobTitleId, YearOfEx, CreateDate, and mode. The "JobTitleId" column is selected.

The right pane displays the "JobTitles" table with columns: PKID, JobTitle, CreateDate, and mode. The "JobTitle" column is selected.

A relationship line connects the "JobTitleId" column in the "Customers" table to the "PKID" column in the "JobTitles" table.

The bottom pane shows the SQL query being generated:

```
SELECT dbo.Customers.PKID, dbo.Customers.IdNum, dbo.Customers.FirstName, dbo.Customers.LastName, dbo.Customers.Address, dbo.Customers.JobTitleId, dbo.Customers.YearOfEx, dbo.Customers.CreateDate, dbo.Customers.mode
FROM dbo.Customers INNER JOIN
dbo.JobTitles ON dbo.Customers.jobTitleId = dbo.JobTitles.PKID
WHERE (dbo.Customers.mode = 1)
```

A "Choose Name" dialog box is open, prompting for a view name. The input field contains "View\_1".

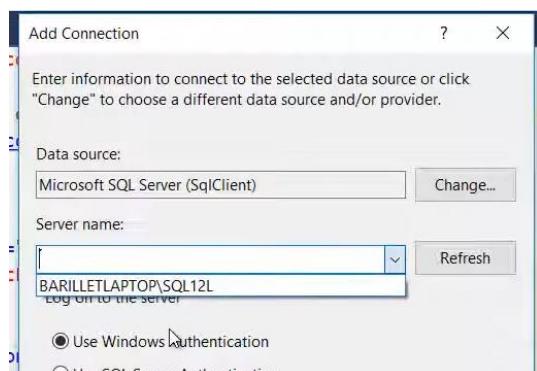
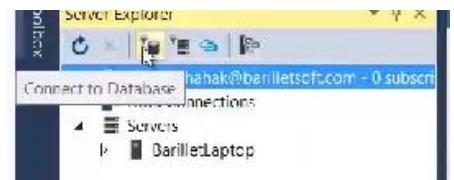
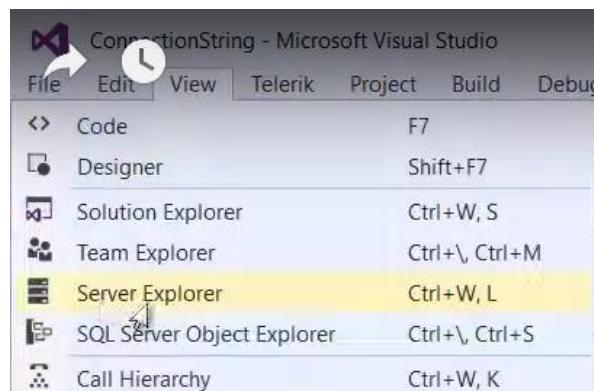
# Web Programming using ASP.NET Web Forms

## ► Lab Agenda

### ► Connection String

- Create Connection to MSSQL in the web.config file

▷ Global.asax  
  └ packages.config  
▷ Web.config



## Connect to MSSQL from ASP.NET

### Connection String using Windows Authentication

In Web.Config file:

```
<ConnectionStrings>
  <add name="MyConnection" ConnectionString="Data Source=ServerName;Initial Catalog=DBNAME; Integrated Security=True"/>
</ConnectionStrings>
```

## Connect to MSSQL from ASP.NET

### Connection String properties when using Windows Authentication

- ▶ <ConnectionStrings> .... </ConnectionStrings>

במקום זה ניתן לרשום מספר רב של חיבורים לבסיס נתונים שונים לכל חיבור יש להגדיר את המאפיינים שלו

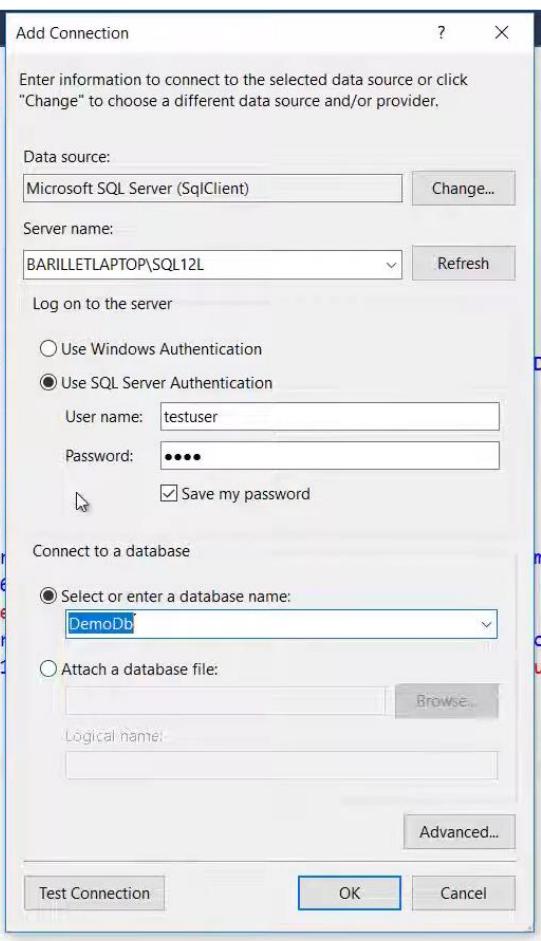
- ▶ <add name="conName" ...>

- ▶ ConnectionString="..."

  - ▶ Data Source - שם שרת בסיס הנתונים

  - ▶ Initial Catalog - שם בסיס הנתונים בתוך השרת

  - ▶ Integrated Security = True / SSPI - Windows Authentication הגדרה כי החיבור יעבוד בשיטת



```
xml version="1.0" encoding="utf-8"?>
<!--
For more information on how to configure your ASP.NET application, please visit
http://go.microsoft.com/fwlink/?LinkId=169433
-->
<configuration>
  <system.web>
    <compilation debug="true" targetFramework="4.5.1"/>
    <httpRuntime targetFramework="4.5.1"/>
  </system.web>
  <connectionStrings>
    <add name="DemoDbConnection" connectionString="Data Source=BARILLETLAPTOP\SQL12L;Initial Catalog=DemoDb;Integrated Security=True" />
    <add name="DemoDbConnectionSQL" connectionString="Data Source=BARILLETLAPTOP\SQL12L;Initial Catalog=DemoDb;Persist Security Info=True;User ID=testuser;Password=*****" />
  </connectionStrings>
</configuration>
```

APTOP\SQL12L;Initial Catalog=DemoDb;Integrated Security=True" />

ETLAPTOP\SQL12L;Initial Catalog=DemoDb,Persist Security Info=True;User ID=testuser;Password=\*\*\*\*\*"/>

SQL Server Auth

## Connect to MSSQL from ASP.NET

### Connection String using SQL server Authentication

In Web.Config file:

```

.....
<ConnectionStrings>
    <add name="MyConnection" ConnectionString=" Data Source=ServerName;Initial;Initial Catalog=DBNAME; User Id=username;
password=password; Persist Security Info=True"/>
</ConnectionStrings>
```

## Connect to MSSQL from ASP.NET

### Connection String properties when using SQL Server Authentication

► <ConnectionStrings> .... </ConnectionStrings>

במקום זה ניתן לרשום מספר רב של חיבורים לבסיס נתונים שונים לכל חיבור יש להגדיר את המאפיינים שלו

► <add name="conName" ConnectionString=" ... " />

► ConnectionString=" ... "

► Data Source - שם שרת בסיס הנתונים

► Initial Catalog - שם בסיס הנתונים בתוך השרת

► User Name - שם היחסור שמתחבר אל בסיס הנתונים

► Password - הסיסמה בה היחסור שהוגדר יכול להתחבר אל שרת בסיס הנתונים

► Persist Security Info = True/False - SQL Server Authentication החיבור מתרבצע בשיטת



```

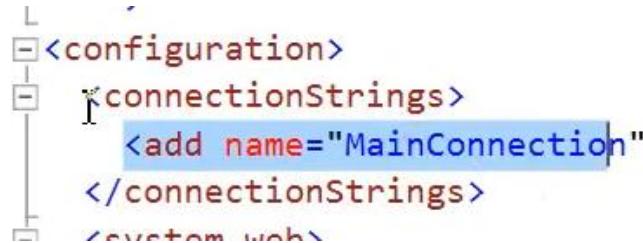
; Initial Catalog=DemoDb; Integrated Security=True" providerName="System.Data.SqlClient"/>
```

### הגדרת החיבור

ל DB

```
protected void btnAdd_Click(object sender, EventArgs e)
{
    //Step 1 - Declare SqlConnection object
    SqlConnection con = new SqlConnection();

    //Step 2 - Assign Connection properties to the connection (from web.config)
    con.ConnectionString = System.Configuration.ConfigurationManager.ConnectionStrings["MainConnection"].ConnectionString;
}
```



//Step 3 - Declare SqlCommand object

```
SqlCommand cmd = new SqlCommand();
```

//Step 4 - Assign the Command to the Connection object, and define the type of the command  
cmd.Connection = con;  
cmd.CommandType = System.Data.CommandType.Text;

### הגדרת SQL הפקודה ל DB

הגדרת  
הפקודה ל DB

//Step 5 - Load the command with the SQL Syntax (insert into...)

```
cmd.CommandText = "INSERT INTO Employees(IdNum,FirstName,LastName,MartialStatusId) VALUES('" + txtIDNumber.Text + "','" +
                txtFirstName.Text + "','" + txtLastName.Text + "','" + ddlMartialStatus.SelectedValue + "')";
```

//Step 6 - Open the connection (for real..)

```
con.Open();
```

### הרצה הפקודה ל DB

//Step 7 - Run the command  
int rows = cmd.ExecuteNonQuery();

//Step 8 - Close the connection  
con.Close();

//Step 7 - Run the command  
int rows = cmd.ExecuteNonQuery();

```
if (rows == 1)
    lblMessage.Text = "Employee inserted successfully!";
else
    lblMessage.Text = "Error, try again later";
```

בדיקות תוצאה

# שלבים ביצוע פקודה Insert

- .1 הגדרת אובייקט חיבור SqlConnection
- .2 טעינת פרמטרים של החיבור לאובייקט החיבור מקובץ config
- .3 הגדרת אובייקט פקודה SqlCommand
- .4 קישור פקד הפקודה לאובייקט החיבור והגדלת סוג הפקודה (טקסט)
- .5 טעינת פקודת SQL לפקד הפקודה
- .6 פתיחת חיבור לבסיס הנתונים
- .7 הרצת הפקודה
- .8 סגירת החיבור

Server Error in '/' Application.

*Keyword not supported: 'dat source'.*

**Description:** An unhandled exception occurred during the execution of the

**Exception Details:** System.ArgumentException: Keyword not supported

**Source Error:**

```

5   |   -->
6  <configuration>
7  |   <connectionStrings>
8  |       <add name="MainConnection" connectionString="Data Source=BARILLETLAPTOP\SQL12L; Initial Catalog=DemoDb; Integrated Security=True" providerName="!>
9  |   </connectionStrings>
10 |   <system.web>
11     <compilation debug="true" targetFramework="4.5"/>
```

localhost:55262/InsertNewEmployee.aspx

Server Error in '/' Application.

*A network-related or instance-specific error occurred while establishing a connection to SQL Server. The server was not found or was not accessible. Verify that the instance name is correct and that SQL Server is configured to allow remote connections. (provider: SQL Network Interfaces, error: 26 - Error Locating Server/Instance Specified)*

**Description:** An unhandled exception occurred during the execution of the current web request. Please review the stack trace for more information about the error and where it

```

5   |   -->
6  <configuration>
7  |   <connectionStrings>
8  |       <add name="MainConnection" connectionString="Data Source=BA\SQL12L; Initial Catalog=DemoDb; Integrated Security=True" providerName="!>
9  |   </connectionStrings>
10 |   <system.web>
11     <compilation debug="true" targetFramework="4.5"/>
12     <httpRuntime targetFramework="4.5"/>
```

**ExecuteNonQuery: Connection property has not been initialized.**

**Description:** An unhandled exception occurred during the execution of the current web request. Please review the stack trace for more information about the error and where it originated in the code.

**Exception Details:** System.InvalidOperationException: ExecuteNonQuery: Connection property has not been initialized.

**Source Error:**

```
//Step 4 Assign the Command to the Connection object
//cmd.Connection = con;
cmd.CommandType = System.Data.CommandType.Text;
```



```
cmd.CommandText = "UPDATE Employees SET FirstName = '" + txtFirstName.Text + "', LastName = '" + txtLastName.Text + "', MartialStatusId = " + ddlMartialStatus.SelectedValue.ToString() +
    " WHERE IdNum = '" + txtIDNumber.Text + "'";

//Step 6 - Open the connection (for real..)

con.Open();

//Step 7 - Run the command
int rows = cmd.ExecuteNonQuery();

if (rows>0)
    lblMessage.Text = "Employee updated successfully!";
else
    lblMessage.Text = "Employee was not found in the database";
```

## שלבים בביצוע פקודה Update

- .1. הגדרת אובייקט חיבור `SqlConnection`
- .2. טיענת פרמטרים של החיבור לאובייקט החיבור מקובץ `web.config`
- .3. הגדרת אובייקט פקודה `SqlCommand`
- .4. קישור פקד הפקודה לאובייקט החיבור והגדרת סוג הפקודה (טקסט)
- .5. טיענת פקודת ה SQL לפקד הפקודה (הפעם פקודת `Update`)
- .6. פתיחת חיבור לבסיס הנתונים
- .7. הרצת הפקודה
- .8. נציג הודעה מתאימה למשתמש האם העובד התעדכן או לא.
- .9. סגירת החיבור

```
//Step 4 - Assign the Command to the Connection object, and define the type of the command
cmd.Connection = con;
cmd.CommandType = System.Data.CommandType.Text;

//Step 5 - Load the command with the SQL Syntax (insert into...)
cmd.CommandText = "DELETE FROM Employees WHERE IdNum=''' + txtIDNumber.Text + '''';

//Step 6 - Open the connection (for real...)
con.Open();

//Step 7 - Run the command
int rows = cmd.ExecuteNonQuery();

if (rows > 0)
    lblMessage.Text = "Employee deleted successfully!";
else
    lblMessage.Text = "Employee was not found in the database";

//Step 8 - Close the connection
con.Close();
```

## שלבים ביצוע פקודה Delete

- .1 הגדרת אובייקט חיבור SqlConnection
  - .2 טיענת פרמטרים של החיבור לאובייקט החיבור מקובץ config web.config
  - .3 הגדרת אובייקט פקודה SqlCommand
  - .4 קישור פקד הפקודה לאובייקט החיבור והגדלת סוג הפקודה (טקסט)
  - .5 טיענת פקודה SQL לפקד הפקודה (הפעם פקודה Delete)
  - .6 פתיחת חיבור לבסיס הנתונים
  - .7 הרצת הפקודה
  - .8 נציג הודעה מתאימה למשתמש האם העובד נמחק או לא.
  - .9 סגירת החיבור
- זכרו - פקודות Delete לא ניתנת לשחזור. מידע שנמחק לא ניתן לשחזור  
כיצד "נמחק" עובד מבלתי לווחקו פיזית מבסיס הנתונים?**

## SqlDataReader

- ▶ האובייקט SqlDataReader מאפשר לנו קריאה של מידע מבסיס הנתונים Read-only, Forward Only.
- ▶ מצוין לבדיקת קיום של מידע בסיס הנתונים SqlDataReader.CanRead
- ▶ ניתן להשתמש באובייקט לקריאת שורות של מידע (שורה אחת כל פעם) SqlDataReader.Read
- ▶ מכיוון שאנו קוראים כל פעם שורה אחת של מידע, לא נשתמש באובייקט זה לקריאה של מידע רב בבאת אחת.

```

//Step 5 - Load the command with the SQL Syntax (insert into...)
cmd.CommandText = "select UserName from Employees where UserName='"
+ txtUserName.Text.Trim() + "' and Pass='"
+ txtPassword.Text.Trim() + "'";

//Step 6 - Declare SqlDataReader
SqlDataReader reader;

//Step 7 - Open the connection (for real..)
con.Open();

//Step 8 - Run the command and open the SqlDataReader
reader = cmd.ExecuteReader();

//Step 9 - Check if Reader.Read() return true/false
if (reader.Read())
{
    reader.Close();
    con.Close();
    Response.Redirect("InsertNewEmployee.aspx");
}

lblMessage.Text = "Username or Password are incorrect";
reader.Close();
con.Close();

```

## דוגמה להצגת מידע

נמשיך את הדוגמה הקודמת,

הפעם במדעה והמשתמש הזרהה בצורה תקינה, נציג את פרטיו בטופס עדכון פרטי העובך.

```

//Step 9 - Check if Reader.Read() return true/false
if (reader.Read())
{
    reader.Close();
    con.Close();
    Session["UserName"] = txtUserName.Text;
    Response.Redirect("InsertNewEmployee.aspx");
}

lblMessage.Text = "Username or Password are incorrect";
reader.Close();
con.Close();

```

```

protected void Page_Load(object sender, EventArgs e)
{
    if(!Page.IsPostBack)
    {
        SqlConnection con = new SqlConnection();
        con.ConnectionString = System.Configuration.ConfigurationManager.ConnectionStrings["MainConnection"].ConnectionString;
        SqlCommand cmd = new SqlCommand();
        cmd.Connection = con;
        cmd.CommandType = System.Data.CommandType.Text;

        cmd.CommandText = "select IdNum,FirstName,LastName,MartialStatusId from Employees where UserName='" + Session["Us
        SqlDataReader reader;

        con.Open();

        reader = cmd.ExecuteReader();

        if(reader.Read())
        {
            txtIDNumber.Text = reader.GetString(0);
            txtFirstName.Text = reader.GetString(1);
            txtLastName.Text = reader.GetString(2);
            ddlMartialStatus.SelectedValue = reader.GetInt32(3).ToString();
        }
    }
}

```

```

reader = cmd.ExecuteReader();

while (reader.Read())
{
    txtIDNumber.Text = reader.GetString(0);
    txtFirstName.Text = reader.GetString(1);
    txtLastName.Text = reader.GetString(2);
    ddlMartialStatus.SelectedValue = reader.GetInt32(3).ToString();
}

//else
//    lblMessage.Text = "Error";

reader.Close();
con.Close();
}
}

```

**נעבור בלולאה על  
תוצאת הקבלה**

