גירסה 1.00 – 1.6.2004

# DataGrid – An ASP.Net Control

## ניר אדר



Nir Adar

Email: underwar@hotmail.com

Home Page: http://underwar.livedns.co.il

אנא שלחו תיקונים והערות אל המחבר.

# DataGrid

סביבת ASP.Net מגיעה עם מספר פקדים חדשים ושימושיים. אחד מהם, ואולי השימושי ביותר, הוא הפקד DataGird. DataGrid הוא Control המאפשר להציג נתונים ממקור נתונים בתוך טבלה.

מקור הנתונים יכול להיות ArrayList או קשר ל-DB וכדו'.

למרות, ואולי מכיוון, שאוביקט זה כה שימושי, למתכנת שרק הכיר אובייקט זה קשה בתחילה להסתדר ולעבוד עימו.

במסמך זה ניסיתי לרכז טיפים רבים ולהדריך צעד אחרי צעד כיצד להשתמש ב-Control זה. המסמך אינו מהווה מדריך שלם לעבודה עם הפקד על כל האפשרויות שבו, אולם לאחר קריאת מסמך זה תוכלו להתחיל מיידית בכתיבת אפליקציות Web המשתמשות בפקד.

# 1. עבודה עם מידע

## 1.1. ArrayList כמקור נתונים עבור DataGrid

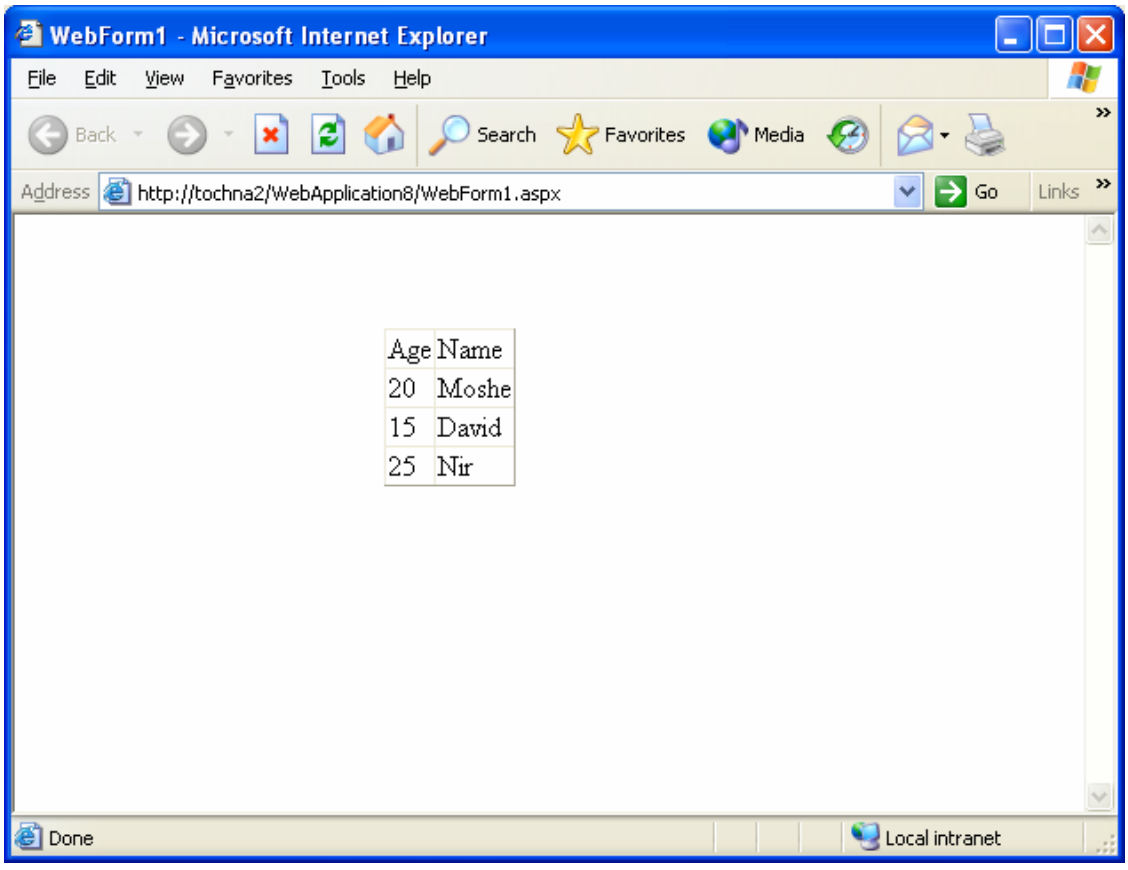
נשתמש ב-class הבא בתור הנתון שאנו רוצים להציג:

```
public class MyData
{
	public string Name
	{
		get { return _name; }
	}
	public int Age
	{
		get { return _age; }
	}
	public MyData(string vName, int vAge)
	{
		_age = vAge;
		_name = vName;
	}

	private int _age;
	private string _name;
}
```

נגרור DataGrid אל דף ה-ASP.Net ונכתוב את הקוד הבא:

```
private void Page_Load(object sender, System.EventArgs e)
{
	ArrayList list = new ArrayList();
	list.Add(new MyData("Moshe", 20));
	list.Add(new MyData("David", 15));
	list.Add(new MyData("Nir", 25));
	DataGrid1.DataSource = list;
	DataGrid1.DataBind();
}
```

והתוצאה:




ה-DataGrid הופך אוטומטית את השמות של ה-Properties השונים של המחלקה לכותרות של העמודות.

נשפר את הקוד:

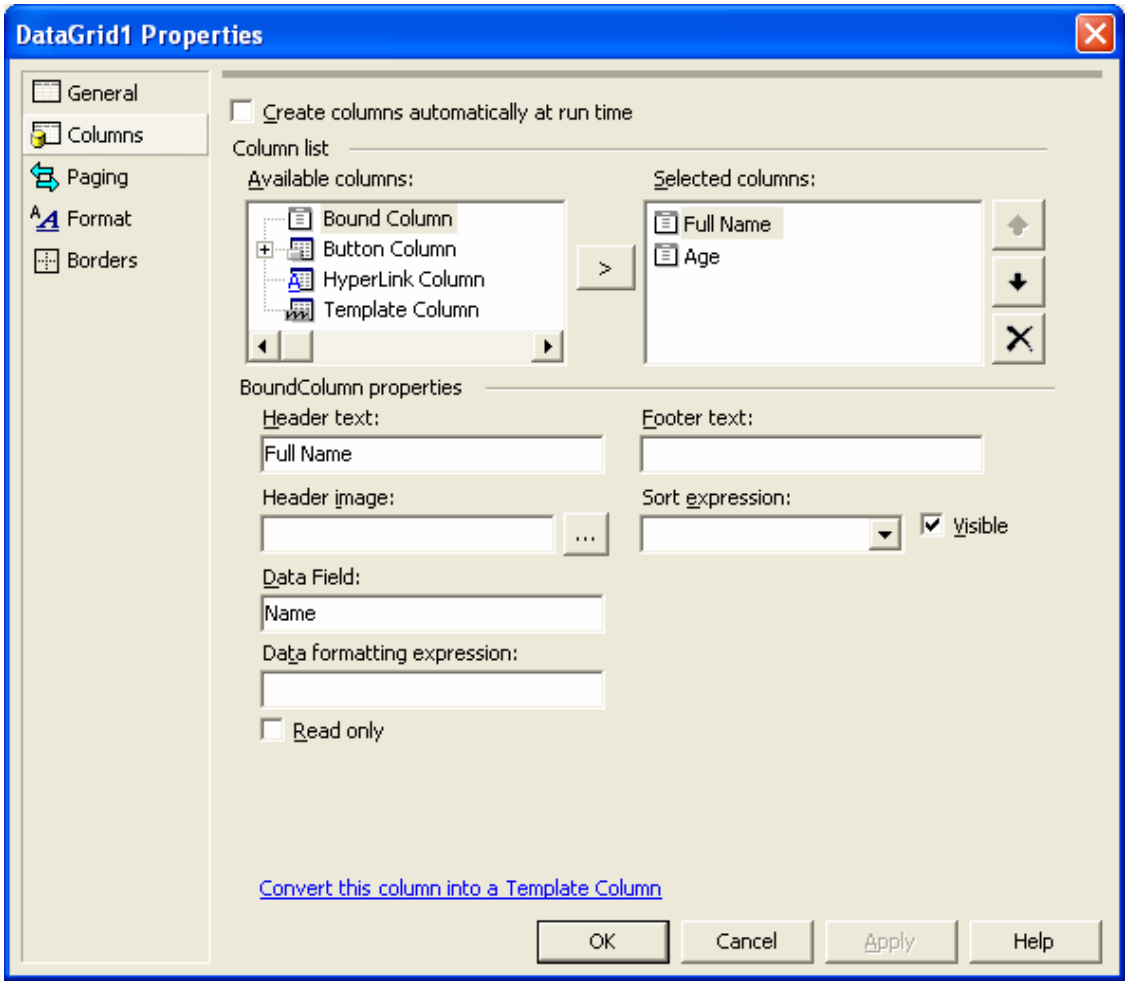
```
private void Page_Load(object sender, System.EventArgs e)
{
      if (!IsPostBack)
      {
            ArrayList list = new ArrayList();
            list.Add(new MyData("Moshe", 20));
            list.Add(new MyData("David", 15));
            list.Add(new MyData("Nir", 25));
            DataGrid1.DataSource = list;
            DataGrid1.DataBind();
      }

}
```

איננו רוצים שכל פעם הטבלה תתמלא מחדש, ולכן נדאג שרק בטעינה הראשונית של הדף נמלא את הטבלה.

בנוסף בהמשך אם לא נעשה זאת, לא נוכל לערוך את הנתונים בטבלה מכיוון שהיא תיווצר בכל פעם מחדש.

## 1.2. בחירת העמודות אותן נרצה להציג

כברירת מחדל ב-DataGrid מוצגות עמודות לפי ה-Properties של האיבר ששודך ל-DataGrid.
אפשרות זו איננה רלוונטית מלבד עבור טבלאות פשוטות ביותר.
נרצה למשל לשלוט בטקסט הכותרות, כך שהכותרות לא יהיו בהכרח שמות המאפיינים אלא כותרות כרצוננו.
נרצה לקבוע אילו מאפיינים מתוך כל הקיימים יופיעו בטבלה. לשם כך נבצע מספר שלבים:
נקבע את התכונה AutoGenerateColumns של ה-DataGrid ל-false, ונלך להגדיר את Columns:




Header Text מגדיר את כותרת העמודה. Data Field מגדיר את ה-Property ממנו ניקח את המידע.

בקוד זה נראה ככה:

```
<asp:DataGrid id="DataGrid1" style="Z-INDEX: 101; LEFT: 208px;
POSITION: absolute; TOP: 64px" runat="server"
AutoGenerateColumns="False">
      <Columns>
            <asp:BoundColumn DataField="Name" HeaderText=
            "Full Name"></asp:BoundColumn>
            <asp:BoundColumn DataField="Age"
            HeaderText="Age"></asp:BoundColumn>
      </Columns>
</asp:DataGrid>
```

דרך שניה לעשות את אותו דבר הינה בזמן ריצה.

ניצור את ה-DataGrid כמו בהתחלה ונקשיב לאירוע ItemDataBound.

אירוע זה קורה כאשר איבר מידע נכנס אל ה-datagrid.

נכתוב את הקוד הבא:

```
            private void DataGrid1_ItemDataBound(object sender,
System.Web.UI.WebControls.DataGridItemEventArgs e)
            {
                  if (e.Item.ItemType == ListItemType.Header)
                  {
                        e.Item.Cells[0].Text = "Age";
                        e.Item.Cells[1].Text = "Full Name";
                  }
            }
```

והאפקט יהיה זהה.

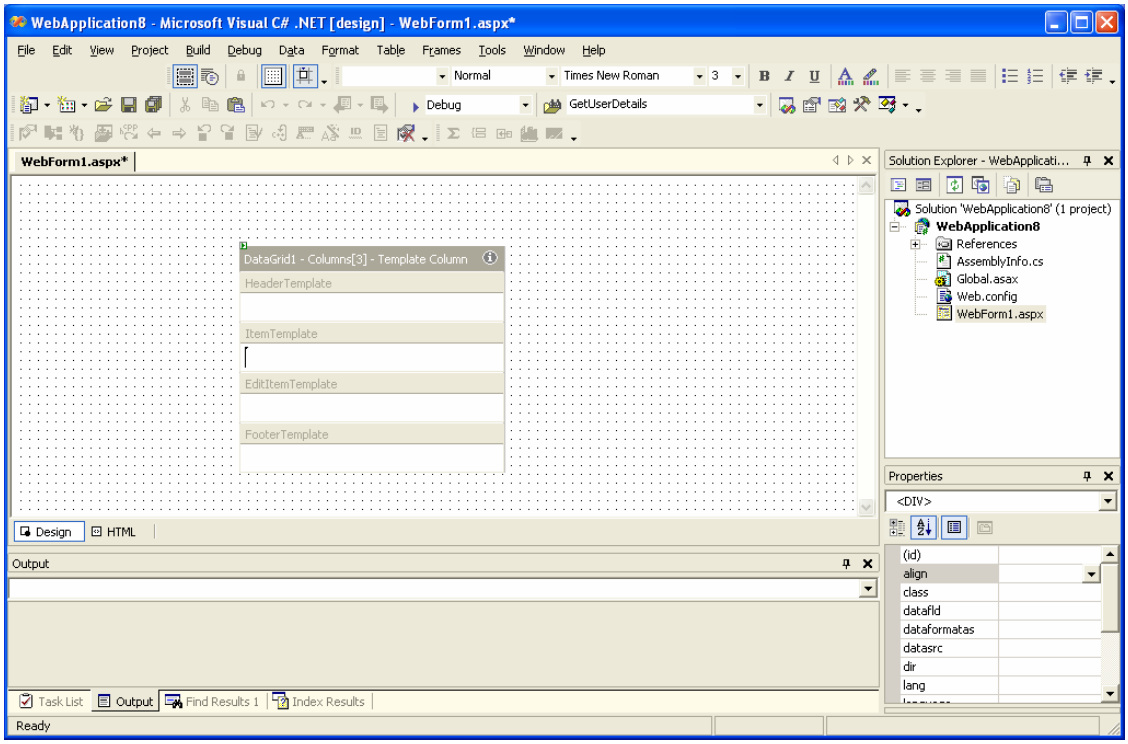
לטעמי עדיפה השיטה הראשונה, מכיוון שהיא נותנת לנו שליטה מלאה במראה של ה-DataGrid, גם אם המחלקה השומרת את המידע תשתנה.

## Template Column .1.3

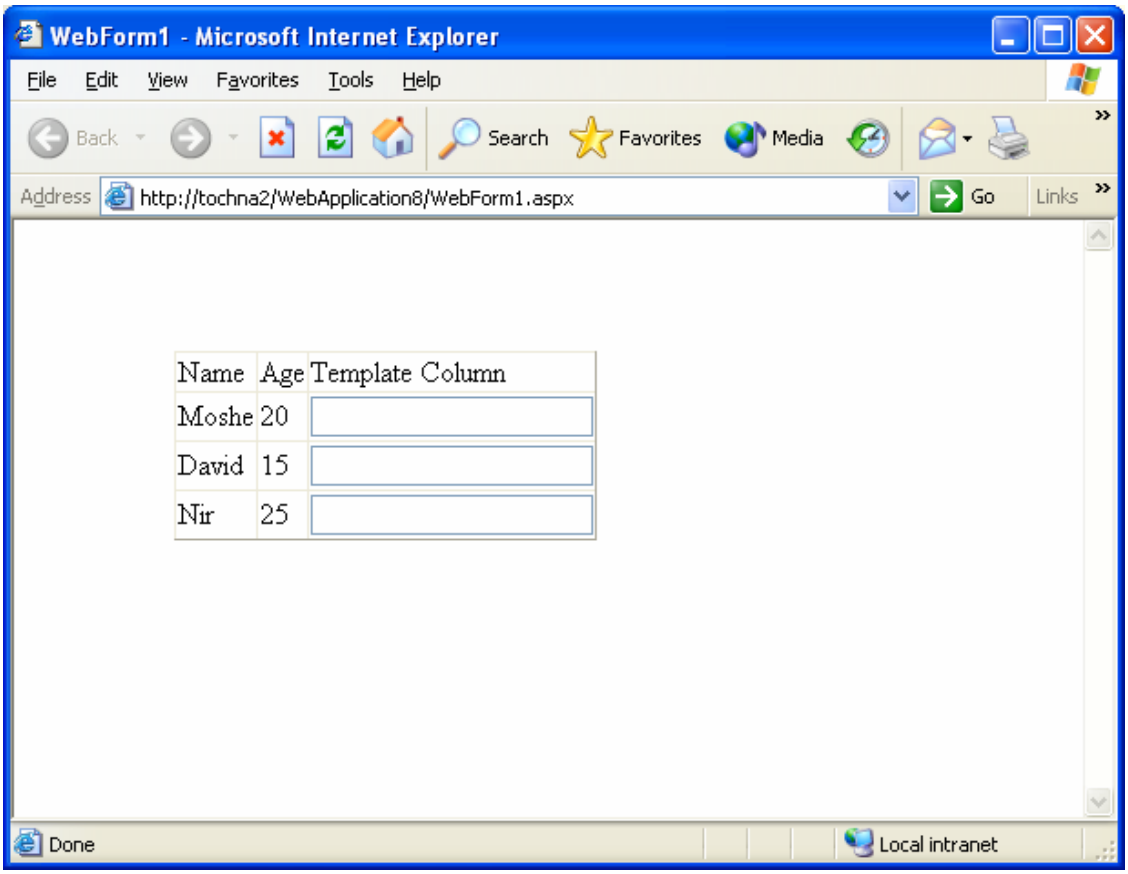
Template Column זו עמודה שלמשתמש יש שליטה מלאה על התוכן שלה, בניגוד לעמודות הרגילות שתוכנן נקבע לפי המידע שנשים בהן. עמודה כזו יכולה להכיל תיבות טקסט, מספר כפתורים וכל דבר אחר שיעלה על דעתנו.

ניצור עמודה template דרך תפריט העמודות כפי שהודגם קודם לכן. לאחר מכן, ב-design view נלחץ מקש ימני על ה-DataGrid, נבחר Edit Template ונבחר את העמודה שיצרנו.

יופיע החלון הבא:




בו אנו יכולים להגיד כל אחד מהשדות שיהיו בתבנית. למשל נוכל לגרור TextBox ולקבל את התוצאה הבאה:




נוסיף כפתור ותווית. הקוד הבא יעבור על כל עמודת ה-template ויציג את הטקסט הנמצא בה:

```
            private void Button1_Click(object sender, System.EventArgs
e)
            {
                  string s = "";
                  for (int i = 0; i < DataGrid1.Items.Count; ++i)
                  {
                        s +=
((TextBox)DataGrid1.Items[i].FindControl("TextBox1")).Text + "<br>";

                  }
                  Label1.Text = s;
            }
```

## 1.4. הרחבת Data Bound על ידי Template Column

לעתים לעשות Bound לנתונים בעזרת עמודת Bound לא מספיק, מכיוון שנרצה לעשות עיבוד נוסף על הנתון לפני הצגתו, או למשל כי נרצה לעשות Bound לשני איברים באותה עמודה, דבר ש- Bound Column אינו מאפשר לנו.

הפתרון הוא שימוש ב-Template Button, בצורה הבאה, למשל:

```
<asp:TemplateColumn HeaderText="My Title">
      <ItemTemplate>
            <%# ((MyClass) Container.DataItem).MyField %>
      </ItemTemplate>
</asp:TemplateColumn>
```

Container.DataItem נותן לנו גישה אל האיבר הנוכחי ממנו נלקחים הנתונים. בדוגמא אנחנו ממירים אותו לסוג MyClass (בהנחה שזה אכן הסוג של אותו האיבר), ומציגים את אחד השדות שבו.

שימוש לדוגמא: אם כל איבר ב-ArrayList אותו אנחנו משדכים לטבלה מכיל בתוכו מחלקה אחרת, ואנחנו מעוניינים לגשת ולהציג את אחד המאפיינים שלה בטבלה.

# 2. שליטה בכותרות

## 2.1. צבע ועיצוב הכותרות

עיצוב נוסף לראש הטבלה נעשה על ידי המאפיין HeaderStyle. ניתן להגדיר CSS class עבור הכותרת, לקבוע צבע רקע ועוד.

## 2.2. הסתרת עמודה

נרצה להסתיר עמודה לפעמים. למשל – למשתמש מסוים אין הרשאה לראות את כל העמודות.
שוב נתלבש על האירוע ItemDataBound, למשל:

```
            private void DataGrid1_ItemDataBound(object sender,
System.Web.UI.WebControls.DataGridItemEventArgs e)
            {
                  e.Item.Cells[1].Visible = false;
            }
```

## 2.3. קביעת רוחב העמודה

כדי לעצב את ה-DataGrid כרצוננו, נרצה לשלוט בגודל עמודה.
נעשה זאת על ידי עריכת דף ה-aspx, למשל בצורה הבאה:

```
<asp:BoundColumn DataField="Name" HeaderText="Name" ItemStyle-
Width="150px"></asp:BoundColumn>
```

# 3. בחירת איברים ברשימה

## 3.1. הוספת קישור עליו ניתן ללחוץ

נוכל להוסיף עמודת קישור עליה ניתן יהיה ללחוץ, וכך לבחור עמודה. כדי לעשות זאת נלך למסך עריכת ה-Columns ונוסיף עמודה מסוג Select. תיווצר עמודה עם כפתורים שעליהם ניתן ללחוץ כדי לבחור את אותה עמודה.

כאשר לוחצים על כפתור כזה, יתרחש אירוע מסוג SelectedIndexChanged.

כדי להגיע אל האיבר שנבחר, נשתמש במאפיין SelectedItem, לדוגמא:

```
            private void DataGrid1_SelectedIndexChanged(object sender,
System.EventArgs e)
            {
                  Label1.Text = DataGrid1.SelectedItem.Cells[1].Text;
            }
```

לכל כפתור Select ניתן להגדיר שם. תכף נראה שנוכל להוסיף מספר כפתורים מהם ניתן לבחור, ולזהות ביניהם לפי השם.

הערה חשובה: כדי שכפתור יעורר את אירוע SelectedIndexChanged המאפיין CommandName שלו חייב להיות Select.

אירוע מעניין נוסף הוא ItemCommand. נשתמש ב-ItemCommand במקרה שקורה מצב שיש לנו בטבלה יותר מכפתור אחד שניתן ללחוץ עליו, ואנחנו רוצים לדעת על איזה כפתור לחץ המשתמש.

במקרה שלוחצים על כפתורים עם שם שונה מ-Select, לא נבחר איבר, ולכן איננו יכולים להשתמש במאפיין SelectedItem. עם זאת, נוכל לגלות את האיבר עליו לחצו בצורה הבאה:

```
            private void DataGrid1_ItemCommand(object source,
System.Web.UI.WebControls.DataGridCommandEventArgs e)
            {
                  Label2.Text = e.CommandName;
                  Label1.Text = e.Item.Cells[1].Text;
```

```
        }
```

כאשר e.CommandName זהו שם הכפתור עליו לחץ המשתמש, ואילו e.Item זו העמודה שבה לחצנו על הכפתור.

ניתן מתוך אירוע זה להגדיר שהעמודה תיבחר, למשל:

```
DataGrid1.SelectedIndex = e.Item.ItemIndex;
```

## 3.2. שינוי הסגנון של האיבר הנבחר

כאשר נבחרת שורה, נוכל לשנות את הסגנון שלה, כדי להדגיש שהיא נבחרה. למשל, נוכל לשנות את צבע הרקע שלה.

השינוי יעשה על ידי המאפיין SelectedItemStyle. האיבר הנבחר בכל רגע יקבל את המאפיין שנגדיר.

נשים לב שכאשר נבחר איבר אחר, האיבר המקורי יחזור לסגנון המקורי, ואין לנו צורך לכתוב קוד מיוחד כדי לדאוג לכך.

דוגמא:

```
        private void DataGrid1_SelectedIndexChanged(object sender,
System.EventArgs e)
        {
              DataGrid1.SelectedItemStyle.BackColor = Color.Ivory;
        }
```

## 3.3. בחירת שורה שלמה

נרצה לאפשר שלחיצה על כל מקום בשורה יבחר את השורה, ולא רק על ה-Select.
לצורך כך נשתמש בטכניקה הבאה: נוסיף איבר Select בתור איבר ראשון ב-DataGrid. ניתן להשאיר אותו נראה או לא נראה, כרצוננו.

תחת האירוע ItemDataBound נבדוק ראשית שאנחנו לא לוחצים על אחת העמודות המיוחדות. לאחר מכן נוסיף לכל אחד מהאיברים קוד JavaScript שיגרום לכך שבמקרה שלוחצים עליו, המידע ישלח בחזרה לשרת. בקוד נניח שאכן ה-Select הוא בעמודה הראשונה.

```
            private void DataGrid1_ItemDataBound(object sender,
                  System.Web.UI.WebControls.DataGridItemEventArgs e)
            {
                  ListItemType itemType = e.Item.ItemType;
                  if ((itemType == ListItemType.Pager) ||
                        (itemType == ListItemType.Header) ||
                        (itemType == ListItemType.Footer))
                  {
                        return;
                  }
                  LinkButton button =
(LinkButton)e.Item.Cells[0].Controls[0];
                  e.Item.Attributes["onclick"] =
                        Page.GetPostBackClientHyperlink(button, "");
            }
```

## 3.4. CheckBoxs בסגנון Hotmail

אפשרות נוספת שנרצה היא ליצור תיבות סימון בצורה דומה לזו ש-Hotmail עושים – טבלה בה בכל שורה תיבת סימון המאפשרת לבחור את אותה שורה, וכמו כן בראש עמודת תיבות הסימון ישנה תיבת סימון אחת, המאפשרת לבחור או לבטל את הבחירה של כל האיברים בשורה.

משימה כזו מערבת קוד בצד השרת ובצד הלקוח.
ראשית ניצור Template Column בה בכותרת נשים CheckBox לו ניתן את השם chkAll והאיבר שנשים בכל אחד מהתאים יהיה CheckBox בשם chkSelect.

קוד ה-ASP.Net הרלוונטי ייראה כך:

```
<Columns>
	<asp:TemplateColumn>
		<HeaderTemplate>
			<asp:CheckBox id="chkAll"
			onclick="javascript:SelectAllCheckboxes(this);"
			runat="server">
			</asp:CheckBox>
		</HeaderTemplate>
		<ItemTemplate>
			<asp:CheckBox id="chkSelect"
			onclick="javascript:HighlightRow(this);"
			runat="server" OnCheckedChanged="dg1_CheckedChanged">
			</asp:CheckBox>
		</ItemTemplate>
	</asp:TemplateColumn>
</Columns>
```

בנוסף נשים בדף ה-ASP.Net את הסקריפטים הבאים, המטפלים בבחירת שורה ובבחירת כל השורות בטבלה:

```
<script language="javascript">

    //---------------------------------------------------------------
    // Select all the checkboxes (Hotmail style)
    //---------------------------------------------------------------
    function SelectAllCheckboxes(spanChk)
    {
              var tableID =
spanChk.parentNode.parentNode.parentNode.parentNode.id;
       var xState = spanChk.checked;
        elm = spanChk.form.elements;
        for (i = 0; i < elm.length; i++)
        {
                    if (elm[i].type == "checkbox" && elm[i].id != spanChk.id &&
                    elm[i].parentNode.parentNode.parentNode.parentNode.id ==
                    tableID)
                    {
                           if(elm[i].checked!=xState)
                           elm[i].click();
                    }
        }
    }

    //---------------------------------------------------------------
    //----Select highlight rows when the checkboxes are selected
    //
    // Note: The colors are hardcoded, however you can use
    //       RegisterClientScript blocks methods to use Grid's
    //       ItemTemplates and SelectTemplates colors.
    //         for ex: grdEmployees.ItemStyle.BackColor OR
    //                 grdEmployees.SelectedItemStyle.BackColor
    //---------------------------------------------------------------
    function HighlightRow(chkB)
    {
              if (chkB.checked)
        {

       chkB.parentElement.parentElement.style.backgroundColor='lightcoral';
             // grdEmployees.SelectedItemStyle.BackColor
                    chkB.parentElement.parentElement.style.color='white';
             // grdEmployees.SelectedItemStyle.ForeColor
        }
        else
        {

       chkB.parentElement.parentElement.style.backgroundColor='white';
             //grdEmployees.ItemStyle.BackColor
                    chkB.parentElement.parentElement.style.color='black';
             //grdEmployees.ItemStyle.ForeColor
        }
    }

</script>
```

ואילו בצד השרת נשים את הקוד הבא:

```
public void dg1_CheckedChanged(object sender, System.EventArgs e)
{
	CheckBox chkTemp = (CheckBox)sender;
	DataGridItem dgi = (DataGridItem)chkTemp.Parent.Parent;
	if (chkTemp.Checked)
	{
		dgi.BackColor = DataGrid1.SelectedItemStyle.BackColor;
		dgi.ForeColor = DataGrid1.SelectedItemStyle.ForeColor;
	}
	else
	{
		dgi.BackColor = DataGrid1.ItemStyle.BackColor;
		dgi.ForeColor = DataGrid1.ItemStyle.ForeColor;
	}
}
```

תפקידו של קוד זה הוא לשמור על צבע השורות שנקבע, בזמן שהדף נטען מחדש באירוע PostBack.

EOF