

Connecting an ASP.NET-form to a database

Connecting an ASP.NET form created with SpreadsheetConverter to a database is very easy. We will do it in 3 steps:

1. Calculate and save the form contents into a database.
2. Retrieve previous entered data from the database, show it in the form and let the user edit it and recalculated and save it again.
3. Show all submitted entries so that we can click on them to edit them.

Step 1 can be implemented in several other ways. You do not even need an ASP.NET-page for that, for example SpreadsheetConverter for HTML + our advanced service or an external tool like Frontpage extension can be used for that.

We are using Visual Studio 2005/2008. If you do not have it, there are free or cheap alternative IDEs out there: [Visual Web Developer 2005/2008 Express Edition Beta](#), and [ASP.NET Web Matrix](#). Actually, you do not even need an IDE, a text editor like notepad plus DOT.NET version 2.0 is enough.

You can download all files in this example by clicking [here](#).

You need to use SpreadsheetConverter for ASP&ASP.NET version 5.2.4 or later.

Part 1: Saving a ASP.NET-form into a database

Part 1 is to save the entered data into a database. In part 1, the saved data cannot be accessed from the website.

The starting point: the Excel spreadsheet

Todo: create a spreadsheet and name the input and output cells we want to store in the database.

We started by creating a simple spreadsheet in Excel. It is a very simple and naive time reporting form containing different controls like radio buttons, dropdown list etc. The user has to enter name & select values for date, arrival, departure, lunch & experience. The spreadsheet calculates the number of hours worked.

There are 6 input fields:

1. Name
2. Today
3. Arrival
4. Departure
5. Lunch
6. Experience

There is 1 output field:

1. Hours, the number of hours worked

	A	B	C
1	Name		
2	Today		
3	Arrival	9	
4	Departure	17	
5	Lunch	1	
6	Experience		
7			
8	Hours	7	
9			

We have called the spreadsheet time_report.xlsx.

The important is that we have named the cells and set with appropriate control attributes:

- B1: name (mark as input cell by using "Mark input cell" menu from SpreadsheetConverter)
- B2: today2 (set calendar control using "Calendar" menu with "Button to popup calendar" as option)
- B3: arrival (set drop down control using "Dropdown list" menu with labels from 1-5)
- B4: departure (set slider control using "Slider" menu with "Horizontal slider" as option & Minimum=0 to Maximum=20)
- B5: lunch (set radio buttons control using "Radio buttons" menu with layout as "1 row", "Horizontal ordering" & labels from 1-5)
- B6: experience (set rating control using "Rating" menu with "5 stars" as option)
- B8: hours (= B4-B3-B5), formula for hours calculation

Note: Please refer SpreadsheetConverter manual for using each controls

Naming cells is done by placing the cursor in the cell, and writing the name into the small textbox at the top left corner.

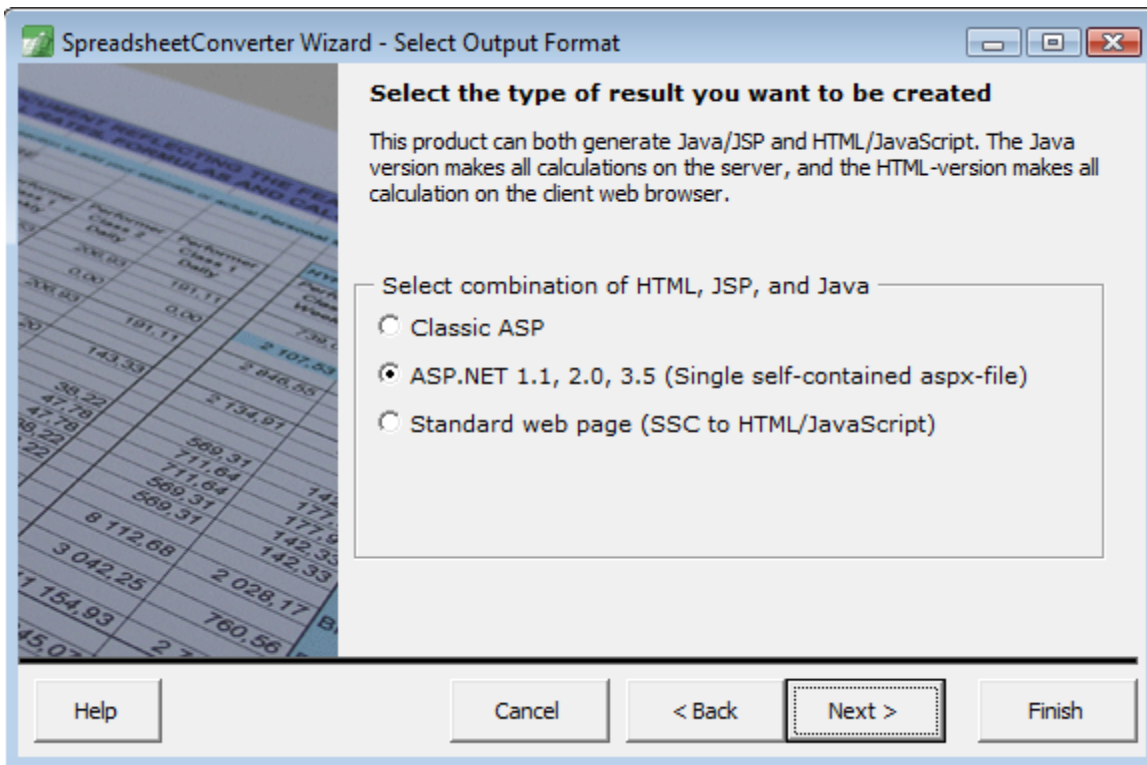
Originally, I called the date field today. However, today seems to be a reserved name in ASP.NET, so I renamed it to today2. When you remove names in Excel, delete the old name and create a new one. You have to remove the old one, since otherwise SpreadsheetConverter might use it.

Generate the ASP.NET-web page

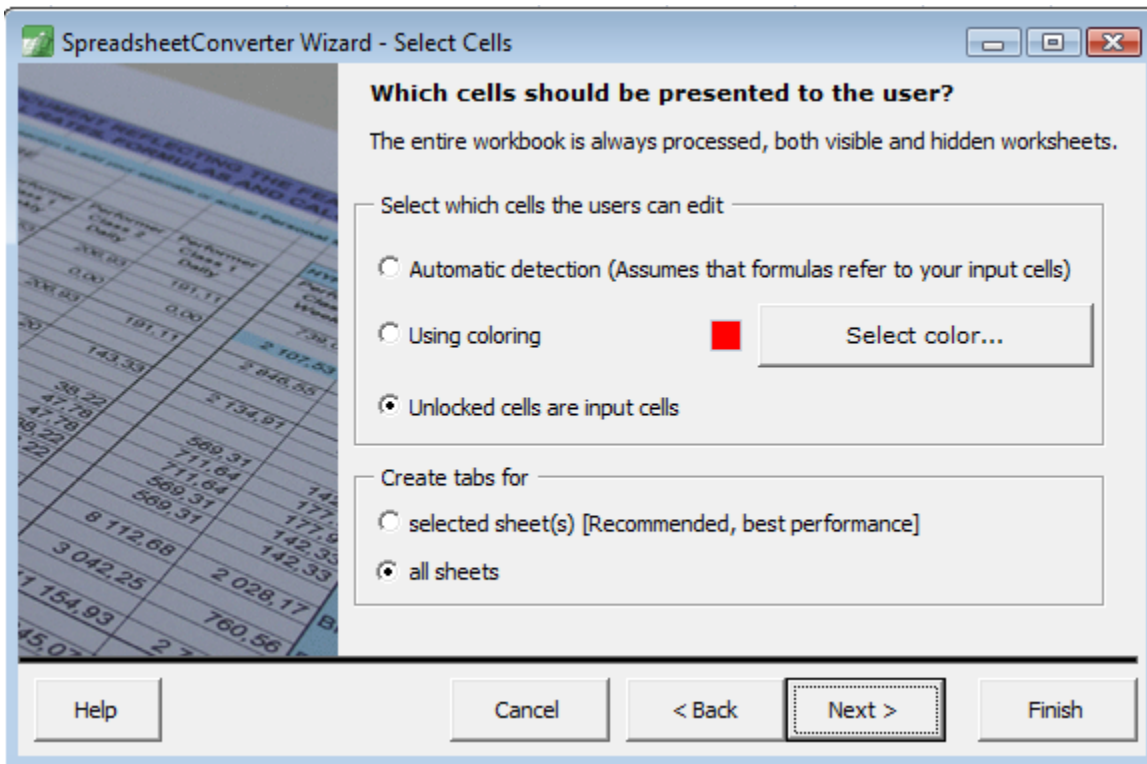
Start SpreadsheetConverter by selecting Convert from the SpreadsheetConverter menu in Excel.

We will select to create an ASP.NET page with Single self-contained aspx-file.

All our adaptations are placed in two aspx files. One aspx contains all the html generated & other calculation.aspx contain formulas as server side in the form of JScript.



All cells in the spreadsheet are locked by default, and we have unlocked the 6 input cells so that SpreadsheetConverter can identify them.



We only keep the submit button, which is the button that submits the page to the server side for saving into the database. Recalculate button is not required to be selected as the formula update is done using ajax calling the calculation.aspx page.

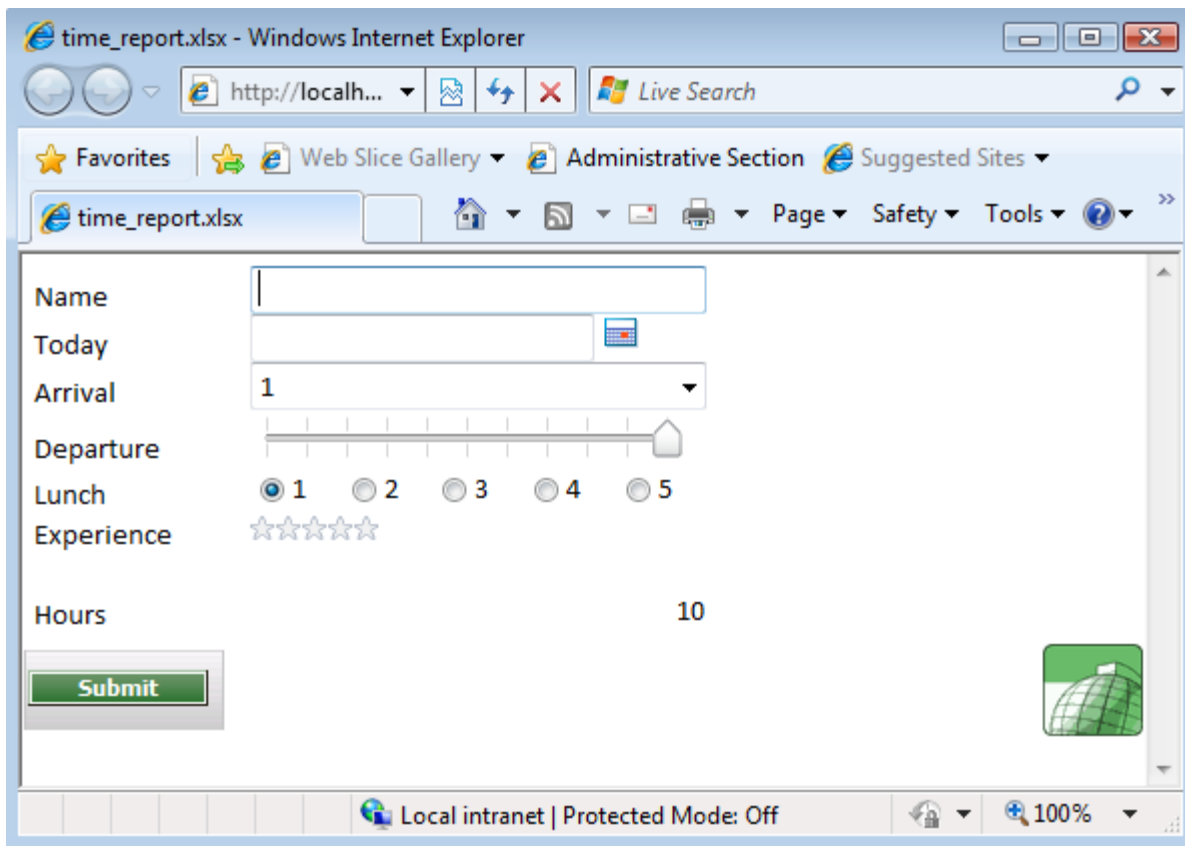
Define the toolbar

The toolbar is the row of buttons that can be shown at the top and at the bottom of each sheet.

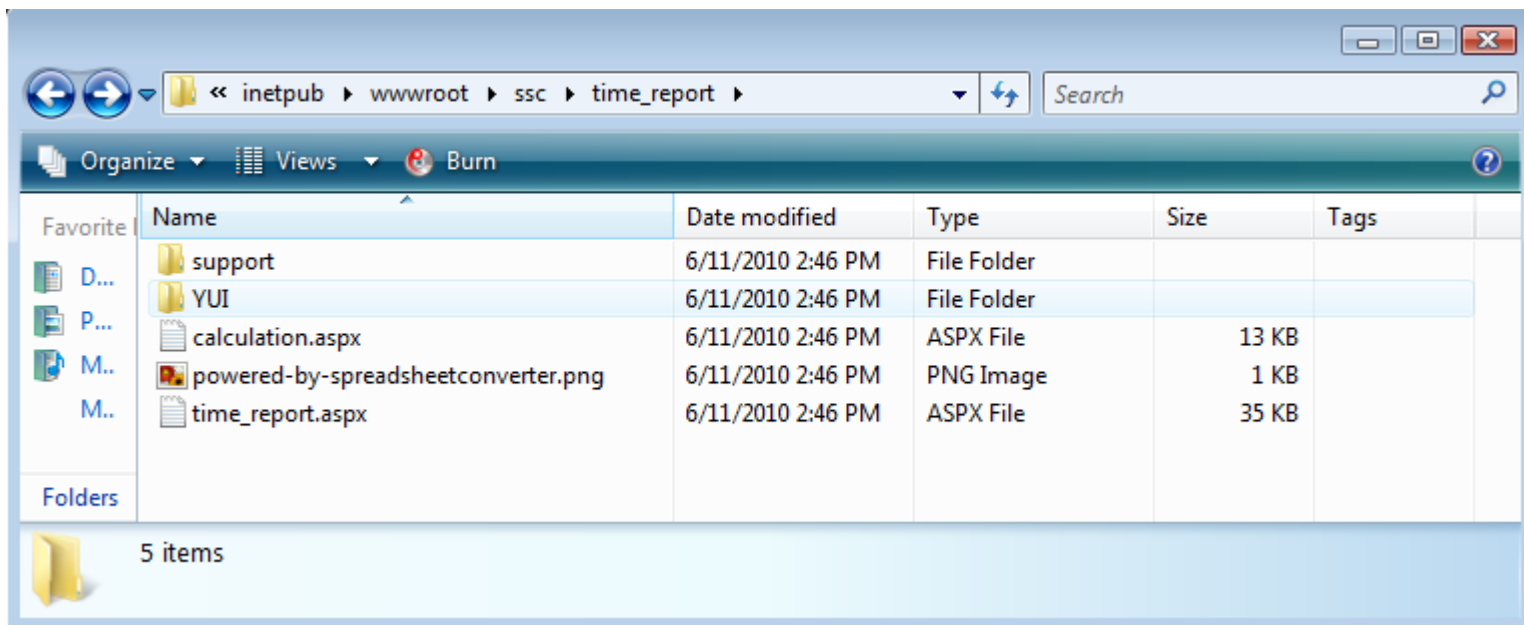
Insert toolbar at (v4 only)

Select buttons to include	Text on button	Wizard buttons	Text on wizard buttons
<input type="checkbox"/> Add recalculate button	<input type="text" value="Update"/>	Next button	<input type="text" value="Next"/>
<input type="checkbox"/> Add reset button	<input type="text" value="Reset"/>	Previous button	<input type="text" value="Previous"/>
<input type="checkbox"/> Add print sheet button	<input type="text" value="Print"/>	Cancel button	<input type="text" value="Cancel"/>
<input type="checkbox"/> Add print all sheets button	<input type="text" value="Print"/>	Finish button	<input type="text" value="Finish"/>
<input checked="" type="checkbox"/> Add submit button	<input type="text" value="Submit"/>		

We can test the form directly. Just change the value for arrival, departure or lunch and the value for hours will be updated. By default Submit button submits the page to <http://www.spreadsheetserver.com/server1/g/submit/submit.aspx> for emailing. We will change this behavior to submit itself so that we can save the data.



These are the files that SpreadsheetConverter generates:



Creating the database

Todo: We need to create a database where the values can be stored.

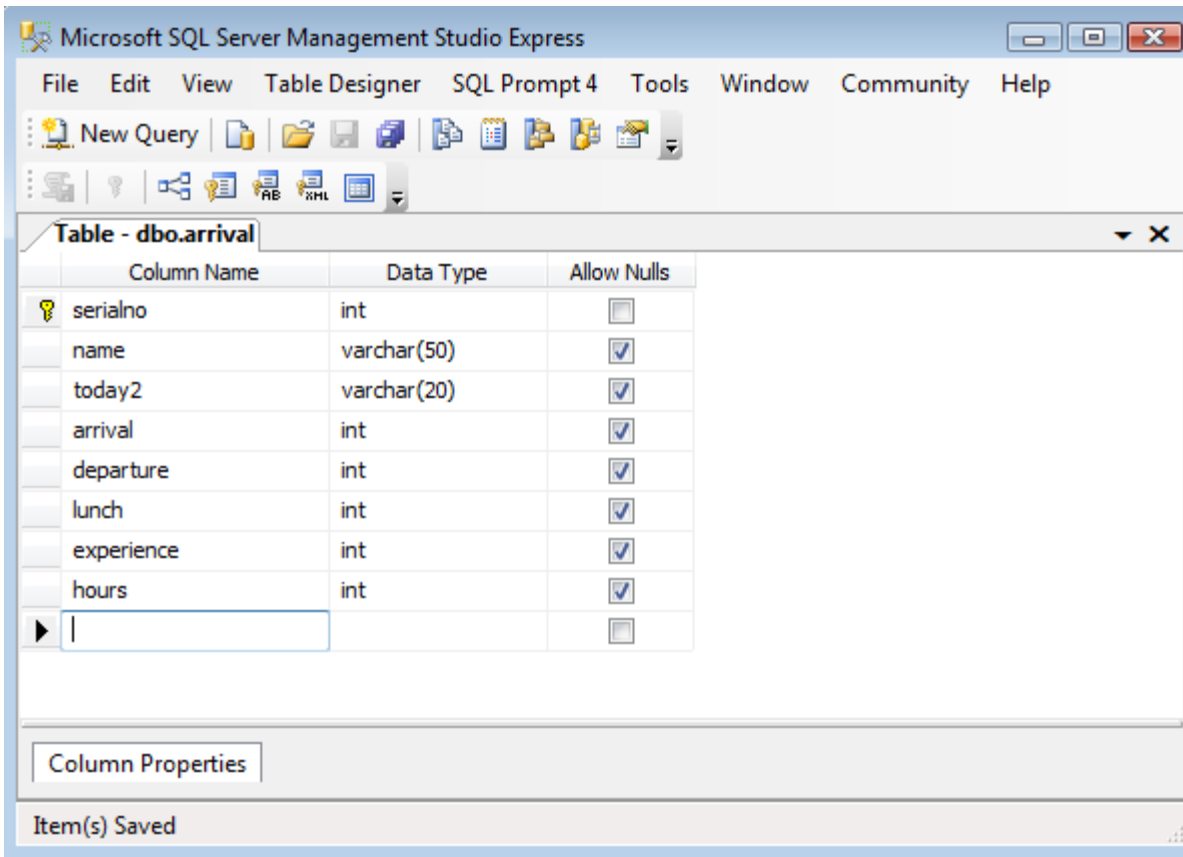
The database will only contain one single table called arrival. Each submit will add one row to that table.

Let us create a database with one single table where we can store the columns. The columns of the table arrival are

- serialno: int, identity
- name: string 50
- today2: string 20 (we could have used a date format)
- arrival: number (int)
- departure: number (int)
- lunch: number (int)
- experience: number(int)
- hours: number(int)

where serialno is the primary key. In this first part, serialno will have no purpose. We call the table arrival.

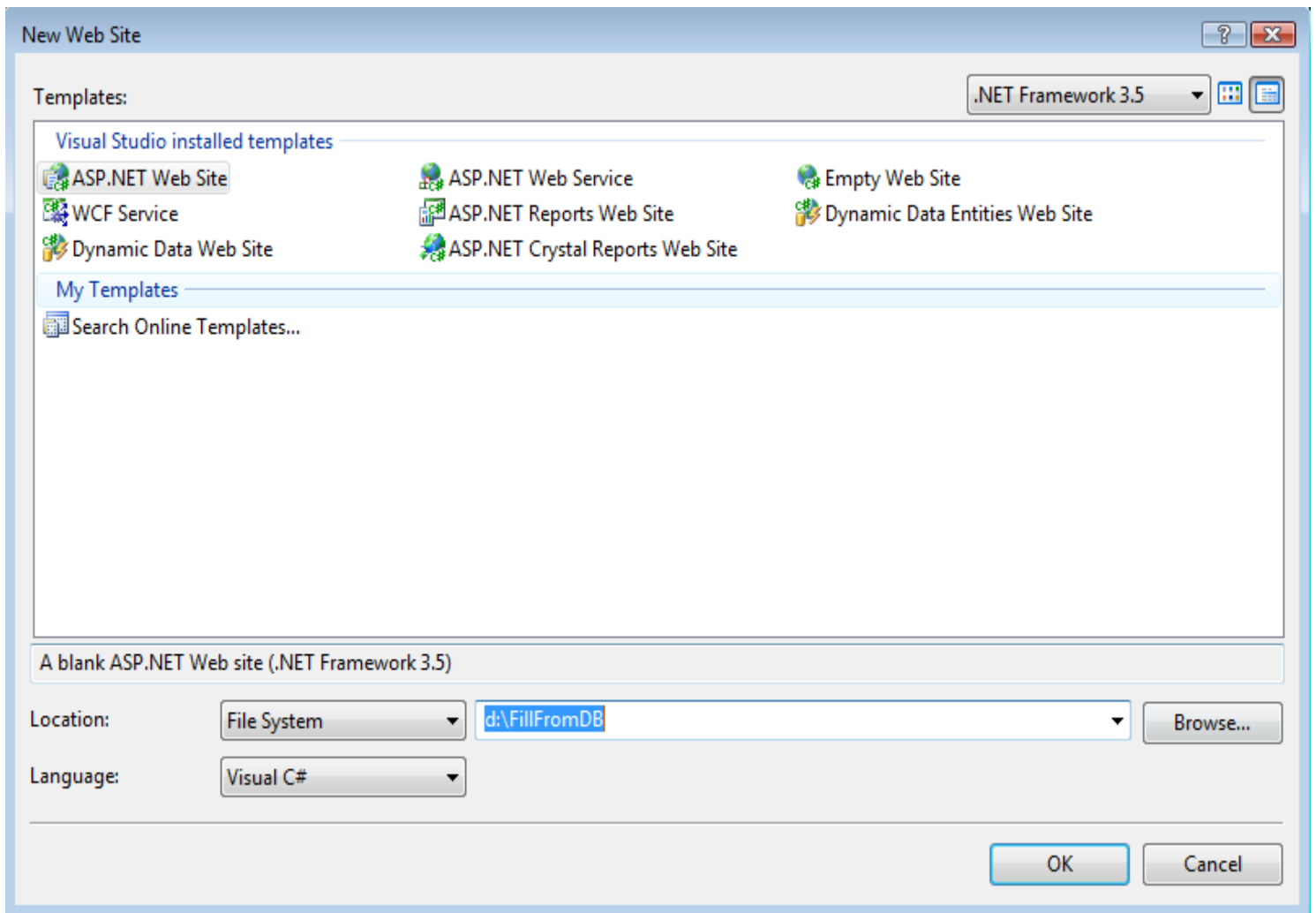
We start Ms Sql Server & create a table named “arrival”.



Creating the ASP.NET-website/application

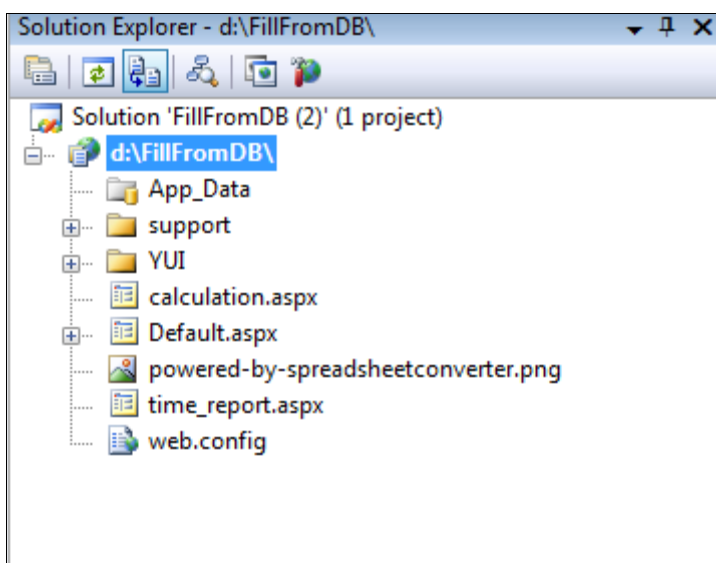
Todo: Create an ASP.NET website and paste all the generated files into the project.

Start Visual Studio 2005/2008 and create a new C# ASP.NET website called FillFromDB.



Paste the files into Visual Studio 2005/2008.

We paste all the generated files into our Visual Studio project using copy/paste.



Change for submit to work

- i) Empty the action attribute of the form

```
<form id='formc' name='formc' method='post' action='http://www.spreadsheetserver.com/serve
<form id='form1' name='formc' method='post' action=''>
```

- ii) Add hidden element “<input type="hidden" id="actionButton" name="actionButton" />” just after form tag.

```
<form id='formc' name='formc' method='post' action=''>
<input type="hidden" id="actionButton" name="actionButton" />
```

- iii) Replace “document.formc.submit(); ” with

```
“ document.getElementById("actionButton").value = "save"; document.formc.submit();
”
```

to include “document.getElementById("actionButton").value = "save";”

- iv) Add following at the top of the page:

```
<%@ Page Language="C#" %>
```

- v) Add server side c# code to handle submit just before the end of <head> tag:

```
<script runat="server" language="c#">
    void Page_Load(object sender, EventArgs e)
    {
        if (Request.Form["actionButton"] != null && Request.Form["actionButton"] ==
"save")
        {
            /* submit was pressed: save and redirect to confirmation page */

        }
    }
</script>
<!-- SpreadsheetConverter Header end -->
</head>
```

Save submitted data to database

Add following code in the server side for complete functionality to save the data:

```
<script runat="server" language="c#">
    // variable declaration equivalent to the columns
    string name = string.Empty;
    string today2 = string.Empty;
    int arrival;
    int departure;
    int lunch;
    int experience;
    int hours;

    // remember to change the connection string with appropriate values
```



```

    string connStr =
"Server=<Server>;Database=<Database>;uid=<Username>;pwd=<Password>";

    void Page_Load(object sender, EventArgs e)
    {
        // Process for insert/save
        if (Request.Form["actionButton"] != null && Request.Form["actionButton"] ==
"save")
        {
            /* submit was pressed: save and redirect to confirmation page */
            name = Request.Form["name"];
            today2 = Request.Form["today2"];
            arrival = int.Parse(Request.Form["arrival"]);
            departure = int.Parse(Request.Form["departure"]);
            // validation for radio button list control
            if (!string.IsNullOrEmpty(Request.Form["lunch"]))
            {
                lunch = int.Parse(Request.Form["lunch"]);
            }
            // validation for rating control as no value may come for the rating if
no selection
            if (!string.IsNullOrEmpty(Request.Form["experience"]))
            {
                experience = int.Parse(Request.Form["experience"]);
            }

            hours = int.Parse(Request.Form["hours"]);

            if (InsertArrival(name, today2, arrival, departure, lunch, experience,
hours))
                Response.Write("Arrival saved.");
        }
    }

    // Method to insert into database
    public bool InsertArrival(string name, string today2, int arrival, int departure,
int lunch, int experience, int hours)
    {
        string sql = "Insert into
Arrival(name,today2,arrival,departure,lunch,experience,Hours)
values (@Name,@Today2,@Arival,@Departure,@Lunch,@Experience,@Hours) ";
        using (System.Data.SqlClient.SqlConnection conn = new
System.Data.SqlClient.SqlConnection(connStr))
        {
            conn.Open();
            System.Data.SqlClient.SqlCommand cmd = new
System.Data.SqlClient.SqlCommand(sql, conn);

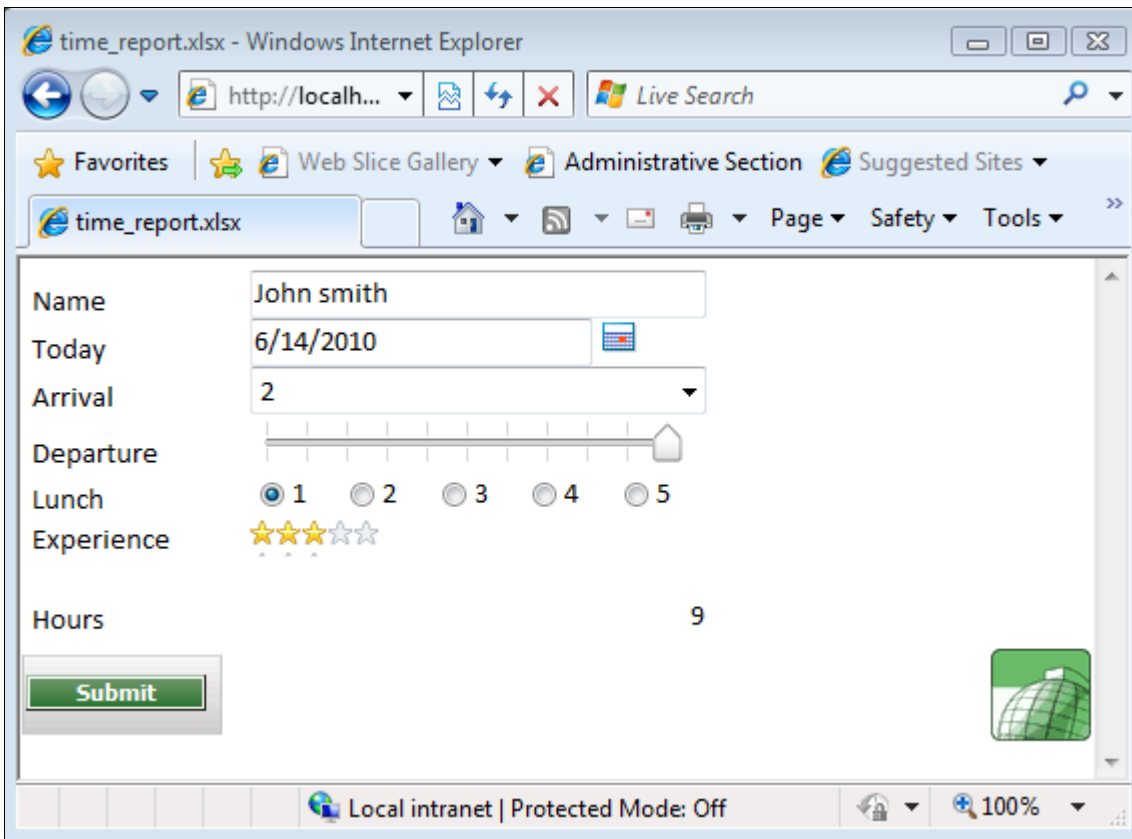
            cmd.Parameters.AddWithValue("@Name", name);
            cmd.Parameters.AddWithValue("@Today2", today2);
            cmd.Parameters.AddWithValue("@Arival", arrival);
            cmd.Parameters.AddWithValue("@Departure", departure);
            cmd.Parameters.AddWithValue("@Lunch", lunch);
            cmd.Parameters.AddWithValue("@Experience", experience);
            cmd.Parameters.AddWithValue("@Hours", hours);

            return cmd.ExecuteNonQuery() == 1;
        }
    }
}
</script>

```

Note: Remember to change the variable “connStr” with appropriate values to point to the database created as above.

If we start the application and enters a person

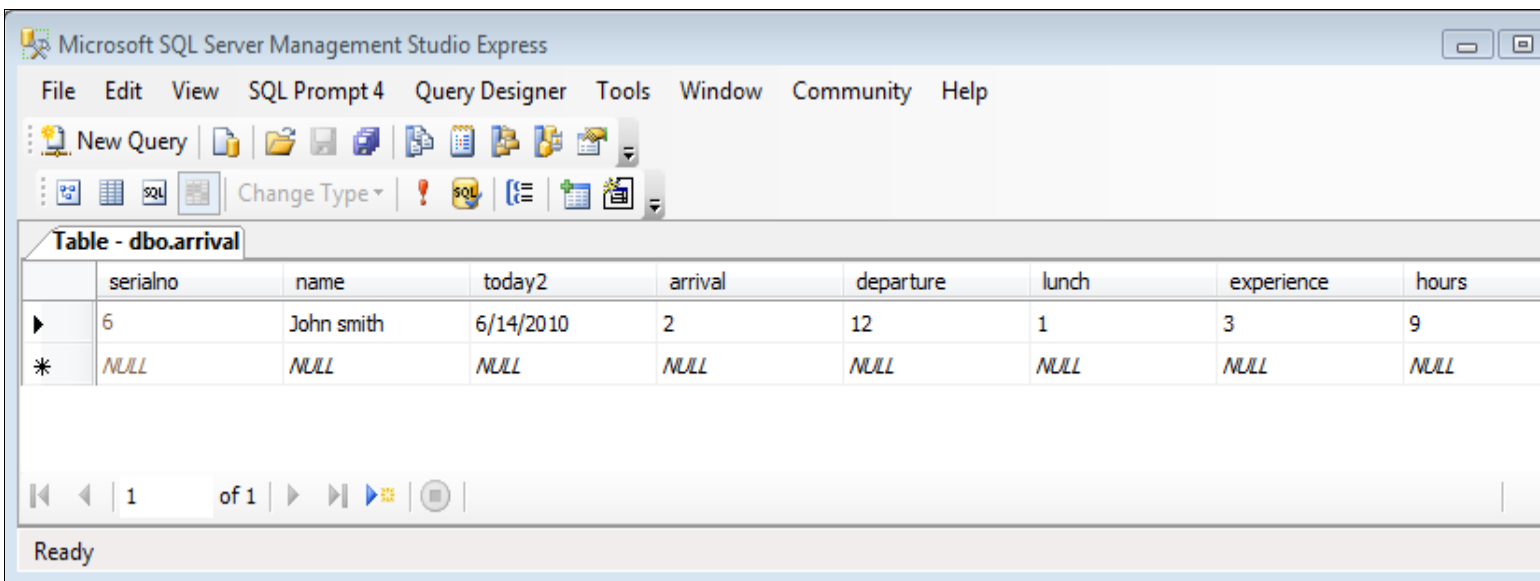


The screenshot shows a web browser window titled "time_report.xlsx - Windows Internet Explorer". The address bar shows "http://localh...". The page content includes a form with the following fields:

- Name: John smith
- Today: 6/14/2010
- Arrival: 2
- Departure: A slider set to 12
- Lunch: Radio buttons for 1, 2, 3, 4, 5, with 1 selected.
- Experience: Five stars, with the first three filled.
- Hours: 9

A "Submit" button is located at the bottom left of the form area.

And look at the database from Ms Sql Server:



The screenshot shows Microsoft SQL Server Management Studio Express. The table view for "Table - dbo.arrival" is displayed with the following data:

	serialno	name	today2	arrival	departure	lunch	experience	hours
▶	6	John smith	6/14/2010	2	12	1	3	9
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

You can also view the database from inside Visual Studio 2005/2008. Right-click on the Server Explorer, Add connection, select Microsoft SQL Server (SqlClient), and enter “Server name”, “Windows authentication” or “Sql server authentication” & select database in the list.

Which database to use: Microsoft Access or Microsoft SQL Server?

If you use Microsoft Access, the code prefix for the classes is OleDb, if you use SqlServer, it is Sql, ie. OleDbParameter and SqlParameter. If you only need to save data and no advanced querying is needed Access is a good database. Reliability for Access has increased during the years, but SqlServer is a better database. [Microsoft has a free version of MS Sql Server called MSDE.](#)

SQL injection

You can read more about SQL injection and why we have to build parameterized SQL-statements here:

- <http://www.developer.com/db/article.php/2243461>
- <http://dotnetjunkies.com/WebLog/richard.dudley/articles/13706.aspx>
- <http://blogs.wdevs.com/ColinAngusMackay/archive/2004/09/25/652.aspx>
- <http://weblogs.asp.net/bleroy/archive/2004/08/18/216861.aspx>

A general checklist on how to secure ASP.NET-sites is found here:

- http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnnetsec/html/CL_SecuAsp.asp
- <http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnnetsec/html/THCMCh10.asp>

Conclusion

Adding code to an ASP.NET-page that stores the contents into a database is easy. In the next step, we will add more database code that fills the form with data from the database and lets the user update it.