To start, one needs to establish a web site.

In establishing a beginning web site I was very conservative, and declared it (by browsing) to be C:\Inetpub\wwwroot which is the root of the IIS based server running on my desktop machine. I know, one can put these anywhere, but from the point of view of testing this using another computer (a Linux box also on my (physical) desktop) this is the absolutely simplest solution, i.e., appropriate for beginners. I also choose C# as my language of choice.

To add a web form:
I called it Intro1. This brought up a Web Page (in HTML) in which I typed the message: "This is what the new Web Form created Intro1.aspx" alright, not too cool, but once a nerd, always a nerd.
Right clicking on this page will bring up the C# code page for it, Intro1.aspx.cs and right click on it, get designer. Drag a button over from the Toolbox onto the form. Double click on “Button” to bring up code, and insert Response.Write(“Hello World”); Later, when you load the page and click on the button, Hello World appears at the top! Amazing!

Drag a Label onto the .apsx form, and change the button1_Click code:

```
public partial class Intro1 : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
    }

    protected void Button1_Click(object sender, EventArgs e)
    {
        Response.Write("Hello World");
        Label1.Text = DateTime.Now.Date.ToLongDateString() + " " +
                      DateTime.Now.TimeOfDay.ToString();
    }
}
```

To run this code

When on points a browser at the location where this is stored, one will see the following:
But, running the code turns out to be non-trivial. I use Firefox as my default browser, and so I need to point it at [http://137.99.123.189/Intro1.aspx](http://137.99.123.189/Intro1.aspx) to “run” the program. The URL was found using `ipconfig` in a cmd.exe window. Alternatively, you can point a browser at http://localhost:xxxx but one needs to know xxxx and that's coming soon *(vide infra)*.

The miracle is that when one clicks on the Button, one obtains:

![Illustration 4: Screen dump showing the results of the (above) code.](image)

Amazing! By the way, I selected the color Green for the Label by finding the color (background) in the properties list and choosing a color.

Parenthetically, dialing [http://137.99.123.189/](http://137.99.123.189/) into your browser will leave you with:
which is just plain miraculous!

**Adding a Drop Down list**

Dragging a DropDownList item from the Toolbox onto the form allows one to create a drop down choice list, and clicking on the “Edit Items...” list allows you to adjust what items are required. You can edit the items, add new ones, edit what's shown as Text, etc..
Adding a PopUpWindow

This turns out to be non-trivial, and the code was stolen from http://forums.asp.net/t/1190488.aspx and adjusted. We have:

```csharp
using System;
using System.Collections;
using System.Configuration;
using System.Data;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.HtmlControls;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Xml.Linq;

public partial class Intro1 : System.Web.UI.Page {
    protected void Page_Load(object sender, EventArgs e) {
        // WebForm1 page_load event
        Session["Test"] = "Hello World";
        Button2.Attributes.Add("onclick", "window.open('WebForm2.aspx','','height=300,width=300');return false");
    }
    protected void Button1_Click(object sender, EventArgs e) {
        Response.Write("Hello World");
        //Label1.Text = DateTime.Now.ToLongDateString();
        Label1.Text = DropDownList_return + ", " + DateTime.Now.Date.ToDateTimeString() + " " +
```
When one puts the lines

```csharp
DateTime.Now.TimeOfDay.ToString();
}
protected void DropDownList1_SelectedIndexChanged(object sender, EventArgs e)
{
    DropDownList_return = DropDownList1.Text;
}
String DropDownList_return = "1st month";
protected void Button2_Click(object sender, EventArgs e)
{
    Response.Write("Button2 clicked"); // perhaps this is co-opted?
}
```

inside the Button2_Click response (as indicated in the original posting), the first time you click on the button nothing happens and the second time you get a child window. So I moved the code to the Page_Load object, which means that the first time you click on Button2 you get what you expect. More important, apparently, the javascript code I've inserted co-opts the C# code for Button2_Click, since the "Response.Write("Button2 clicked"); // perhaps this is co-opted?" command is never executed.

Carl David
Department of Chemistry
University of Connecticut
Storrs, Connecticut 06269-3060
Carl.David@uconn.edu