ASP.NET Fundamentals

Introduction

The HTTP Protocol

- Web applications are based on the HTTP protocol. This protocol defines how clients (web browsers in most cases) interact with HTTP servers.
- The HTTP interaction between the clients and the servers is based on a request response cycle.
- The HTTP protocol is a stateless one. We need to use a separated mechanism for managing the sessions.

The Web Application

- The web application is a collection of files, such as `*.aspx`, `*.html`, `*.css` and others.
- Each web application has a life cycle with specific evens, such as its startup or its final shutdown,that we can hook with.

The DNS Server

- Each domain is mapped with a specific IP address. This mapping is hand-held by the DNS server.

Domains Registration

- In the past, the domains registration was controlled by one company.
- When the domains registration system stopped being a monopole the fees started to drop.

The Web Server

- The web server is a software application capable of hosting web applications.
- The web server provides the web applications it hosts with various services, such as FTP support, mail exchange service, security features and others.

The IIS Server

- IIS is Microsoft web server. We can use it for hosting the web applications we develop in .NET.

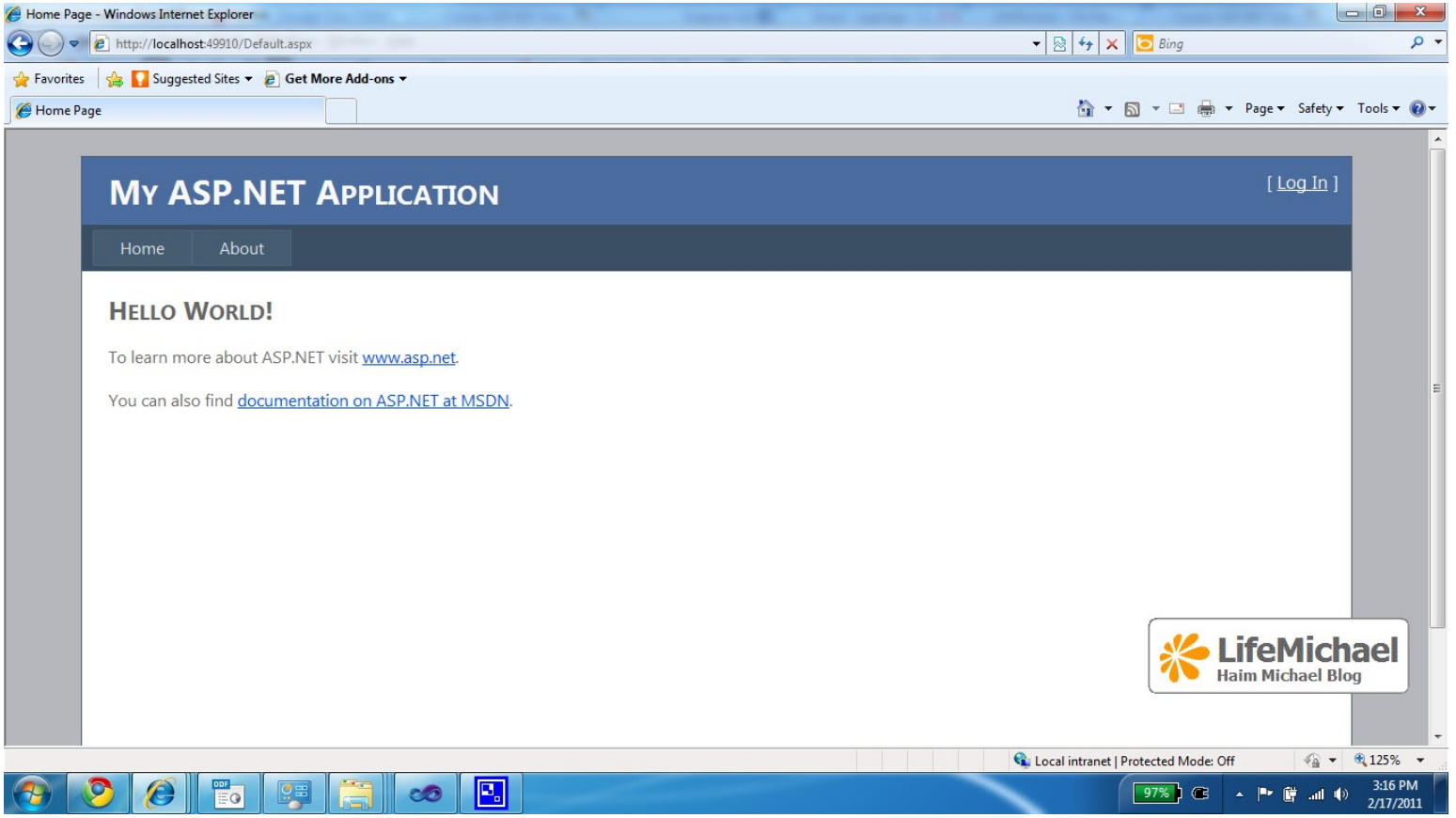
Virtual Directories

- Single IIS installation is capable of hosting more than one web application.
- Each web application resides in a virtual directory. Each virtual directory is mapped with a physical directory on the hard drive.
- When using the visual studio we can instruct it to create a new virtual directory for the web site we develop.

The Development Web Server

- The ASP.NET development web server is a lightweight web server that allows us to host an ASP.NET web application outside the bounds of the IIS.
- We can use this light weight server during the development phase and avoid the IIS.

Hello World in ASP.NET

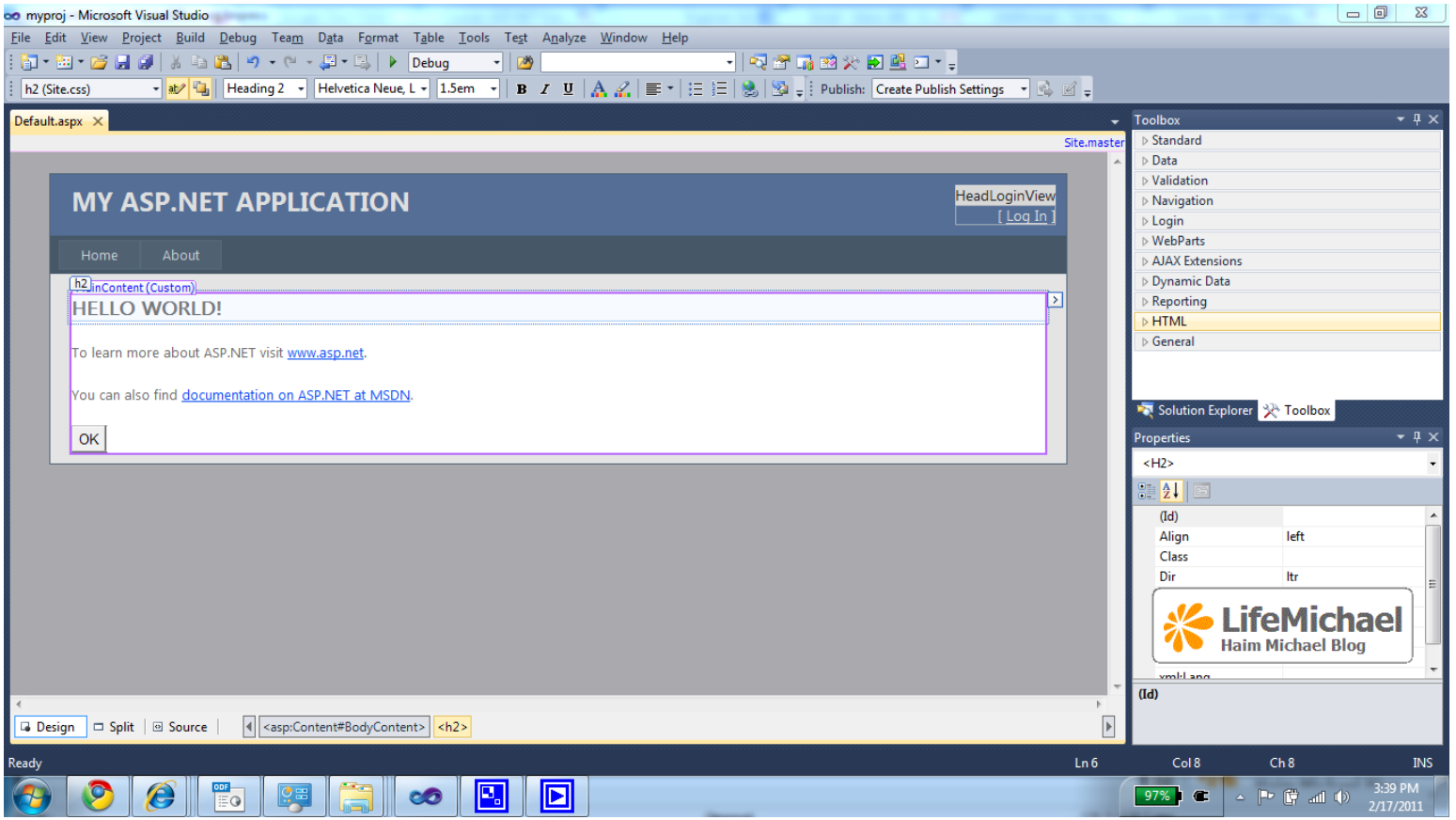


YouTube

The Visual Studio Designer Tools

- The visual studio provides an HTML toolbox that allows us to select an HTML control we want to place on our web page.

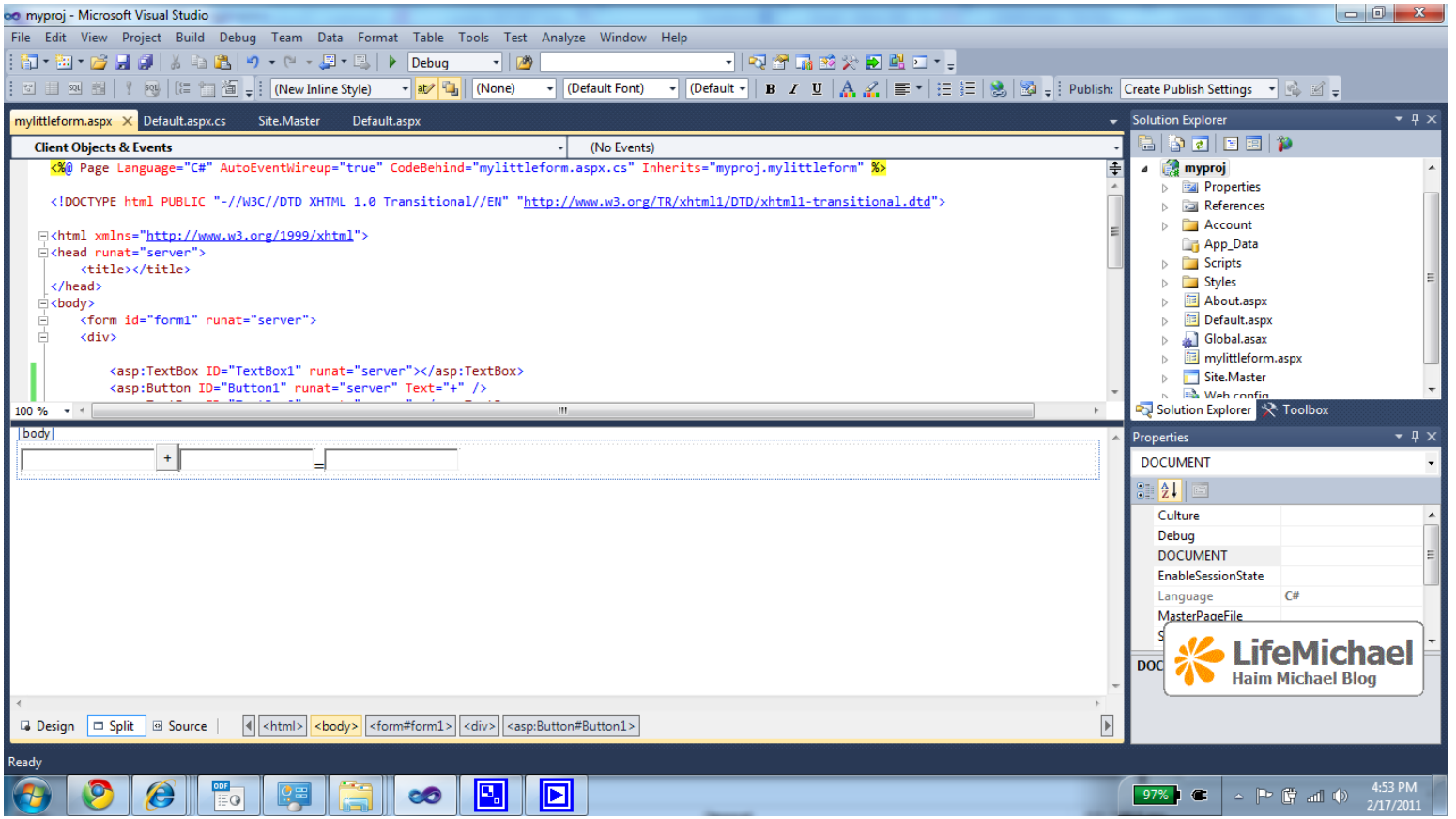
The Visual Studio Designer Tools



Creating Simple Web Form

- We can select the project we are working on, right click the mouse and select 'Add New Item'.
- The next step would be selecting the 'Web Form' option.

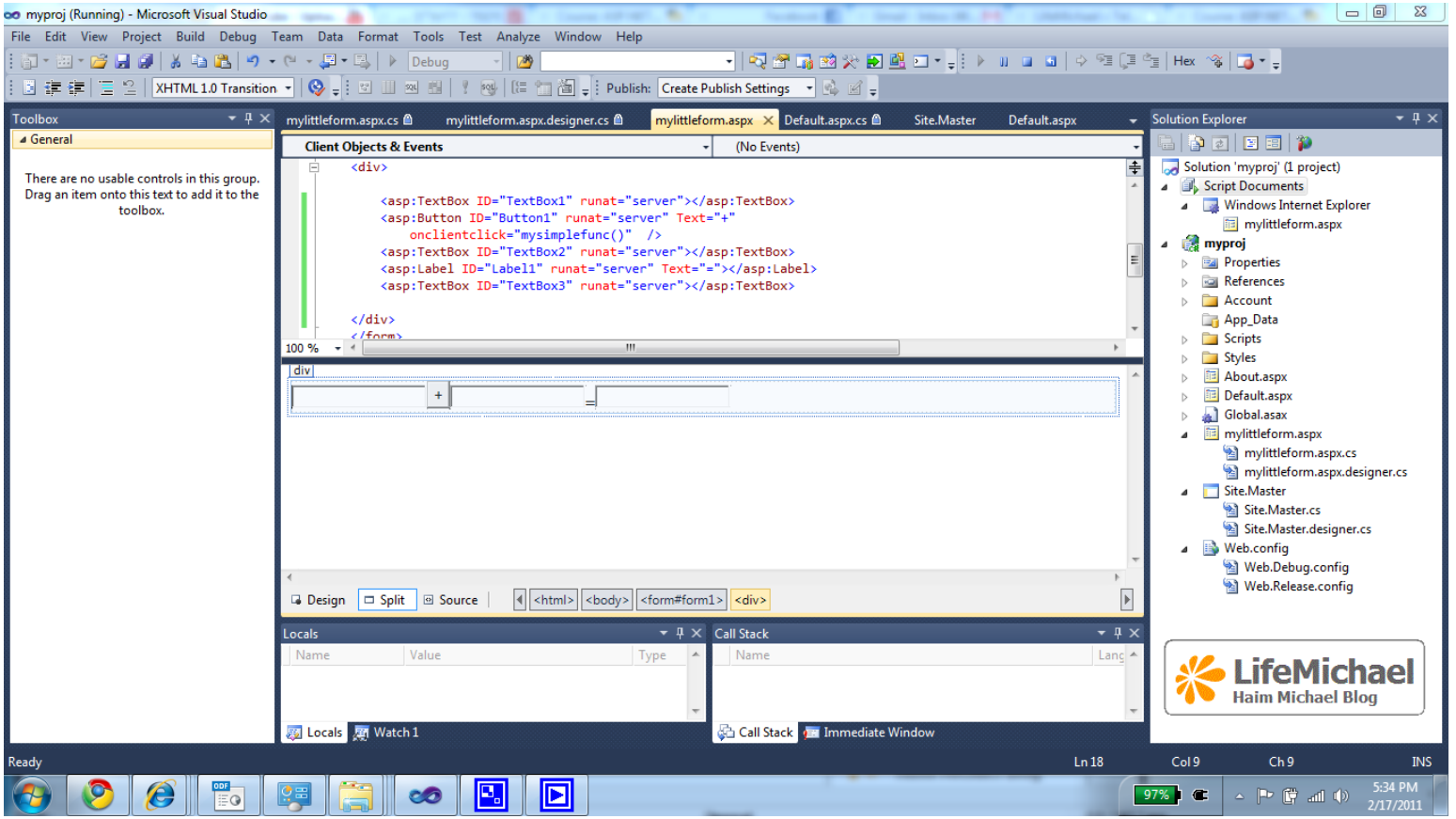
Creating Simple Web Form



The Importance of JavaScript

- The two most important roles the JavaScript fulfills are validating the user input in order to create a friendly user interface and interacting with the document object model while using the JavaScript Ajax capabilities.
- We can easily add JavaScript code into our aspx web page. We can also assign form components events handling attributes with calls to JavaScript functions we define.

The Importance of JavaScript



The Code Behind

- ASP.NET provides us with the possibility to write code in C# that will be executed on the server side. That code is known as the code behind.
- The code behind allows us getting a clear separation between the presentation and the business logic.

Compilation into Valid .NET *.dll

- The code behind can be written in any of the available .NET programming languages.
- That code is compiled into valid .NET *.dll assemblies that provides us with excellent performance.

Web Controls

- The available web controls allow us to develop the user interface in a similar way to building WPF applications.

Automatic State Maintenance

- The ASP.NET web controls automatically maintain their state using the `_VIEWSTATE` hidden field.

The .NET Base Class Libraries

- The ASP.NET web application can use any of the available assemblies in the .NET base class libraries.

Simple Configuration

- We can easily configure our ASP.NET web application by editing the `Web.config` file.

Rich ASP.NET Web Controls

- The variety of ASP.NET web controls allow us to write less HTML code by our self and get auto generated HTML emitted back by the server.

```
<asp:Button ID="Button1" runat="server" Text="+" onclientclick="mysimplefunc()"  />
```

```
<input type="submit" name="Button1" value="+" onclick="mysimplefunc();" id="Button1" />
```

Master Pages

- Using the master pages we can attach a common user interface frame to a set of related pages we want to display in a similar way.
- The master page defines place holders that other *.aspx files can plug.
- Each and every new ASP.NET project already includes a master page. The master page was introduced in ASP.NET 2.0.

Themes

- We can create a theme that will govern the look and feel of our entire web application. The support for themes was added in ASP.NET 2.0.

Web Parts

- We can develop a web application composed of web parts, that each and every one of them can be customized by the users who can also store their setting for a later usage. The support for themes was added in ASP.NET 2.0.

LINQ

- The ASP.NET 3.5 framework added the ability to use the LINQ programming model.

Silverlight

- The ASP.NET 3.5 framework added a huge range of Silverlight based components.

The Entity Framework

- The ASP.NET 3.5 framework added the support for the ADO Entity Framework. We can bind our user interface GUI components with entity classes.

Dynamic Data

- ASP.NET 3.5 supports the development of data driven web applications that tables in its database expose their data through customized URI addresses.
- ASP.NET 3.5 supports this capability similarly to Ruby on Rails.

Ajax

- ASP.NET 3.5 added an integrated support for Ajax code development.

GZIP

- ASP.NET 4.0 allows us to compress the view state data using the GZIP format.

Charts

- ASP.NET 4.0 introduces the ASP.NET chart control, that allows us to develop ASP.NET web pages that include charts.

MVC

- ASP.NET 4.0 introduces the Model View Controller template, that assists developers with the implementation of the MVC design pattern.

HTML 5

- Microsoft develops new HTML5 based rich internet web components.