PRACTICAL- 01

Study of .NET framework with comparison of various framework versions**.**

Microsoft started development on the .NET Framework in the late 1990s originally under the name of Next Generation Windows Services (NGWS). By late 2000 the first beta versions of .NET 1.0 were released.

Version 3.0 of the .NET Framework is included with Windows Server 2008 and Windows Vista. Version 3.5 is included with Windows 7, and can also be installed on Windows XP and the Windows Server 2003 family of operating systems. On 12 April 2010, .NET Framework 4 was released alongside Visual Studio 2010.

The .NET Framework family also includes two versions for mobile or embedded device use. A reduced version of the framework, the.NET Compact Framework, is available on Windows CE platforms, including Windows Mobile devices such as smart phones. Additionally, the .NET Micro Framework is targeted at severely resource-constrained devices.

* .NET Framework 1.0

This is the first release of the .NET Framework, released on 13 February 2002 and available for Windows 98, ME, [NT 4.0](http://en.wikipedia.org/wiki/Windows), [2000](http://en.wikipedia.org/wiki/Windows), and [XP](http://en.wikipedia.org/wiki/Windows). Mainstream support by [Microsoft](http://en.wikipedia.org/wiki/Microsoft) for this version ended 10 July 2007, and extended support ended 14 July 2009, with the exception of XP [Media Center](http://en.wikipedia.org/wiki/Windows_XP_Media_Center) and [Tablet PC](http://en.wikipedia.org/wiki/Windows_XP_Tablet_PC_Edition) editions.

* .NET Framework 1.1

This is the first major .NET Framework upgrade. It is available on its own as a redistributable package or in a [software development kit](http://en.wikipedia.org/wiki/Software_development_kit), and was published on 3 April 2003. It is also part of the second release of Microsoft [Visual Studio .NET](http://en.wikipedia.org/wiki/Visual_Studio_.NET) (released as Visual Studio .NET 2003). This is the first version of the .NET Framework to be included as part of the Windows operating system, shipping with Windows Server 2003. Mainstream support for .NET Framework 1.1 ended on 14 October 2008, and extended support ends on 8 October 2013. Since .NET 1.1 is a component of Windows Server 2003, extended support for .NET 1.1 on Server 2003 will run out with that of the OS – currently 14 July 2015. .NET 1.1 is the last available version for [Windows NT 4.0](http://en.wikipedia.org/wiki/Windows_NT_4.0).

If .NET Framework 1.0 is removed, installing only .NET Framework 1.1 also provides the system support for version 1.0, except in rare instances where an application will not run because it checks the version number of a library.

**Changes in 1.1 in comparison with 1.0**

* Built-in support for mobile [ASP.NET](http://en.wikipedia.org/wiki/ASP.NET) controls. Previously available as an add-on for .NET Framework, now part of the framework.
* Security changes – enable Windows Forms assemblies to execute in a semi-trusted manner from the Internet, and enable [Code Access Security](http://en.wikipedia.org/wiki/Code_Access_Security) in ASP.NET applications.
* Built-in support for [ODBC](http://en.wikipedia.org/wiki/ODBC) and [databases](http://en.wikipedia.org/wiki/Oracle). Previously available as an add-on for .NET Framework 1.0, now part of the framework.
* [.NET Compact Framework](http://en.wikipedia.org/wiki/.NET_Compact_Framework) – a version of the .NET Framework for small devices.
* Internet Protocol version 6 ([IPv6](http://en.wikipedia.org/wiki/IPv6)) support.
* Numerous [API](http://en.wikipedia.org/wiki/Application_programming_interface) changes.
* .NET Framework 2.0

Released with [Visual Studio 2005](http://en.wikipedia.org/wiki/Visual_Studio_2005), [Microsoft SQL Server 2005](http://en.wikipedia.org/wiki/Microsoft_SQL_Server_2005), and [BizTalk](http://en.wikipedia.org/wiki/BizTalk) 2006.

* The 2.0 Redistributable Package can be downloaded for free from [Microsoft](http://www.microsoft.com/downloads/details.aspx?FamilyID=0856eacb-4362-4b0d-8edd-aab15c5e04f5&DisplayLang=en), and was published on 22 January 2006.
* The 2.0 Software Development Kit (SDK) can be downloaded for free from [Microsoft](http://www.microsoft.com/downloads/details.aspx?FamilyID=fe6f2099-b7b4-4f47-a244-c96d69c35dec&DisplayLang=en).
* It is included as part of [Visual Studio 2005](http://en.wikipedia.org/wiki/Visual_Studio_2005) and [Microsoft SQL Server 2005](http://en.wikipedia.org/wiki/Microsoft_SQL_Server_2005).
* Version 2.0 without any Service Pack is the last version with support for [Windows 98](http://en.wikipedia.org/wiki/Windows_98) and [Windows Me](http://en.wikipedia.org/wiki/Windows_Me). Version 2.0 with Service Pack 2 is the last version with official support for [Windows 2000](http://en.wikipedia.org/wiki/Windows_2000) although there have been some unofficial workarounds published online to use a subset of the functionality from Version 3.5 in Windows 2000.[[5]](http://en.wikipedia.org/wiki/.NET_Framework_version_history#cite_note-5) Version 2.0 with Service Pack 2 requires [Windows 2000](http://en.wikipedia.org/wiki/Windows_2000) with SP4 plus KB835732 or KB891861 update, [Windows XP](http://en.wikipedia.org/wiki/Windows_XP) with SP2 or later and [Windows Installer](http://en.wikipedia.org/wiki/Windows_Installer) 3.1 (KB893803-v2)
* It shipped with [Windows Server 2003 R2](http://en.wikipedia.org/wiki/Windows_Server_2003_R2) (not installed by default).

**Changes in 2.0 in comparison with 1.1**

* Generics
* Language support for [generics](http://en.wikipedia.org/wiki/Generic_programming) built directly into the .NET CLR.
* Full [computing](http://en.wikipedia.org/wiki/64-bit) support for both the [x64](http://en.wikipedia.org/wiki/X64) and the [IA-64](http://en.wikipedia.org/wiki/IA-64) hardware platforms.
* Numerous API changes.
* Microsoft SQL Server integration – .NET 2.0, VS 2005, and SQL Server 2005 are all tied together. This means that instead of using [T-SQL](http://en.wikipedia.org/wiki/Transact-SQL), one can build [stored procedures](http://en.wikipedia.org/wiki/Stored_procedures) and triggers in any of the .NET-compatible languages.
* A new hosting API for native applications wishing to host an instance of the .NET runtime. The new API gives a fine grain control on the behavior of the runtime with regards to [multithreading](http://en.wikipedia.org/wiki/Multithreading_(computer_architecture)), memory allocation, assembly loading and more ([detailed reference](http://winfx.msdn.microsoft.com/library/en-us/dv_fxunmanref/html/703b8381-43db-4a4d-9faa-cca39302d922.asp)). It was initially developed to efficiently host the runtime in [Microsoft SQL Server](http://en.wikipedia.org/wiki/Microsoft_SQL_Server), which implements its own scheduler and memory manager.
* Many additional and improved ASP.NET web controls.
* New data controls with declarative data binding.
* New personalization features for [ASP.NET](http://en.wikipedia.org/wiki/ASP.NET), such as support for themes, skins, master pages and web parts.
* [.NET Micro Framework](http://en.wikipedia.org/wiki/.NET_Micro_Framework) – a version of the .NET Framework related to the [Smart Personal Objects Technology](http://en.wikipedia.org/wiki/Smart_Personal_Objects_Technology) initiative.
* Membership provider
* Partial classes
* Nullable types
* Anonymous methods
* Iterators
* Data tables
* Introduces Common Language Runtime 2.0
* .NET Framework 3.0

.NET Framework 3.0, formerly called WinFX, was released on 21 November 2006. It includes a new set of [managed code](http://en.wikipedia.org/wiki/Managed_code) APIs that are an integral part of [Windows Vista](http://en.wikipedia.org/wiki/Windows_Vista) and [Windows Server 2008](http://en.wikipedia.org/wiki/Windows_Server_2008) operating systems. It is also available for Windows XP SP2 and [Windows Server 2003](http://en.wikipedia.org/wiki/Windows_Server_2003) as a download. There are no major architectural changes included with this release; .NET Framework 3.0 uses the same [Common Language Runtime](http://en.wikipedia.org/wiki/Common_Language_Runtime) (CLR) as .NET Framework 2.0 Unlike the previous major .NET releases there was no .NET Compact Framework release made as a counterpart of this version. Version 3.0 of the .NET Framework shipped with Windows Vista. It also shipped with Windows Server 2008 as an optional component (disabled by default).

.NET Framework 3.0 consists of four major new components:

* [Windows Presentation Foundation](http://en.wikipedia.org/wiki/Windows_Presentation_Foundation) (WPF), formerly code-named Avalon; a new [user interface](http://en.wikipedia.org/wiki/User_interface) subsystem and [API](http://en.wikipedia.org/wiki/Application_programming_interface) based on [XML](http://en.wikipedia.org/wiki/XML) and [vector graphics](http://en.wikipedia.org/wiki/Vector_graphics), which uses [3D computer graphics](http://en.wikipedia.org/wiki/3D_computer_graphics) hardware and [Direct3D](http://en.wikipedia.org/wiki/Direct3D) technologies. See [WPF SDK](http://msdn.microsoft.com/en-us/library/ms754130.aspx) for developer articles and documentation on WPF.
* [Windows Communication Foundation](http://en.wikipedia.org/wiki/Windows_Communication_Foundation) (WCF), formerly code-named Indigo; a service-oriented messaging system which allows programs to interoperate locally or remotely similar to [web services](http://en.wikipedia.org/wiki/Web_service).
* [Windows Workflow Foundation](http://en.wikipedia.org/wiki/Windows_Workflow_Foundation) (WF) allows for building of task automation and integrated transactions using [workflows](http://en.wikipedia.org/wiki/Workflow).
* [Windows CardSpace](http://en.wikipedia.org/wiki/Windows_CardSpace), formerly code-named InfoCard; a software component which securely stores a person's digital identities and provides a unified [interface](http://en.wikipedia.org/wiki/Interface_(computing)) for choosing the identity for a particular transaction, such as logging in to a website.
* .NET Framework 3.5

Version 3.5 of the .NET Framework was released on 19 November 2007, but it is not included with [Windows Server 2008](http://en.wikipedia.org/wiki/Windows_Server_2008). As with .NET Framework 3.0, version 3.5 uses Common Language Runtime (CLR) 2.0, that is, the same version as .NET Framework version 2.0. In addition, it installs .NET Framework 2.0 SP1, (installs .NET Framework 2.0 SP2 with 3.5 SP1) and .NET Framework 3.0 SP1 (installs .NET Framework 3.0 SP2 with 3.5 SP1), which adds some methods and properties to the BCL classes in version 2.0 which are required for version 3.5 features such as [Language Integrated Query (LINQ)](http://en.wikipedia.org/wiki/Language_Integrated_Query). These changes do not affect applications written for version 2.0, however

As with previous versions, a new .NET Compact Framework 3.5 was released in tandem with this update in order to provide support for additional features on Windows Mobile and[Windows Embedded CE](http://en.wikipedia.org/wiki/Windows_Embedded_CE) devices.

The source code of the [Base Class Library](http://en.wikipedia.org/wiki/Base_Class_Library) in this version has been partially released (for debugging reference only) under the [Microsoft Reference Source License](http://en.wikipedia.org/wiki/Microsoft_Reference_License#Microsoft_Reference_Source_License_.28Ms-RSL.29)

Service Pack 1

The .NET Framework 3.5 Service Pack 1 was released on 11 August 2008. This release adds new functionality and provides performance improvements under certain conditions, especially with WPF where 20–45% improvements are expected. Two new data service components have been added, the [ADO.NET Entity Framework](http://en.wikipedia.org/wiki/ADO.NET_Entity_Framework) and [ADO.NET Data Services](http://en.wikipedia.org/wiki/ADO.NET_Data_Services). Two new assemblies for web development, System.Web.Abstraction and System.Web.Routing, have been added; these are used in the [ASP.NET MVC Framework](http://en.wikipedia.org/wiki/ASP.NET_MVC_Framework) and, reportedly, will be utilized in the future release of ASP.NET Forms applications. Service Pack 1 is included with [SQL Server 2008](http://en.wikipedia.org/wiki/SQL_Server_2008) and [Visual Studio 2008 Service Pack 1](http://en.wikipedia.org/wiki/Visual_Studio_2008). It also featured a new set of controls called "Visual Basic Power Packs" which brought back Visual Basic controls such as "Line" and "Shape". Version 3.5 SP1 of the .NET Framework shipped with Windows 7. It also shipped with Windows Server 2008 R2 as an optional component (disabled by default).

.NET Framework 3.5 SP1 Client Profile

For the .NET Framework 3.5 SP1 there is also a new variant of the .NET Framework, called the ".NET Framework Client Profile", which at 28 MB is significantly smaller than the full framework and only installs components that are the most relevant to [desktop](http://en.wikipedia.org/wiki/Desktop_computer) applications. However, the Client Profile amounts to this size only if using the online installer on Windows XP SP2 when no other .NET Frameworks are installed or using [Windows Update](http://en.wikipedia.org/wiki/Windows_Update). When using the off-line installer or any other OS, the download size is still 250 MB.

* .NET Framework 4

Key focuses for this release are:

* [Parallel Extensions](http://en.wikipedia.org/wiki/Parallel_Extensions) to improve support for [parallel computing](http://en.wikipedia.org/wiki/Parallel_computing), which target [multi-core](http://en.wikipedia.org/wiki/Multi-core) or [distributed](http://en.wikipedia.org/wiki/Distributed_computing) systems To this end, technologies like PLINQ (Parallel [LINQ](http://en.wikipedia.org/wiki/Language_Integrated_Query)) a parallel implementation of the LINQ engine, and [Task Parallel Library](http://en.wikipedia.org/wiki/Task_Parallel_Library), which exposes parallel constructs via method calls. are included.
* New [Visual Basic .NET](http://en.wikipedia.org/wiki/Visual_Basic_.NET) and [C#](http://en.wikipedia.org/wiki/C_Sharp_(programming_language)) language features, such as implicit line continuations, [dynamic dispatch](http://en.wikipedia.org/wiki/Dynamic_dispatch), [named parameters](http://en.wikipedia.org/wiki/Named_parameter), and [optional parameters](http://en.wikipedia.org/wiki/Parameter_(computer_science)#Default_arguments).
* Support for [Code Contracts.](http://msdn.microsoft.com/en-us/devlabs/dd491992.aspx)
* Inclusion of new types to work with [arbitrary-precision arithmetic](http://en.wikipedia.org/wiki/Arbitrary-precision_arithmetic) ([System.Numerics.BigInteger](http://msdn.microsoft.com/en-us/library/system.numerics.biginteger.aspx)) and [complex numbers](http://en.wikipedia.org/wiki/Complex_number) ([System.Numerics.Complex](http://msdn.microsoft.com/library/system.numerics.complex(VS.100).aspx)).
* Introduce Common Language Runtime (CLR) 4.0.
* .NET Framework 4.5

.NET Framework 4.5 was released on 15 August 2012. a set of new or improved features were added into this version. The .NET Framework 4.5 is only supported on [Windows Vista](http://en.wikipedia.org/wiki/Windows_Vista) or later The .NET Framework 4.5 uses Common Language Runtime 4.0, with some additional runtime features.

.NET for Metro style apps

[Metro-style](http://en.wikipedia.org/wiki/Metro_(design_language)) [apps](http://en.wikipedia.org/wiki/Application_software) are designed for specific form factors and leverage the power of the Windows operating system. A subset of the .NET Framework is available for building Metro style apps for [Windows 8](http://en.wikipedia.org/wiki/Windows_8) using [C#](http://en.wikipedia.org/wiki/C_Sharp_(programming_language)) or [Visual Basic](http://en.wikipedia.org/wiki/Visual_Basic). This subset is called *.NET APIs for apps*.

The version of .NET Framework, runtime and libraries, used for Metro style apps is a part of the new [Windows Runtime](http://en.wikipedia.org/wiki/Windows_Runtime), which is the new platform and application model for [Metro style apps](http://en.wikipedia.org/wiki/Metro_(design_language)). It is an ecosystem that houses many platforms and languages, including [.NET Framework](http://en.wikipedia.org/wiki/.NET_Framework), [C++](http://en.wikipedia.org/wiki/C%2B%2B) and [HTML5](http://en.wikipedia.org/wiki/HTML5)/[JavaScript](http://en.wikipedia.org/wiki/JavaScript).

* Framework Core

Core Features

* Ability to limit how long the [regular expression](http://en.wikipedia.org/wiki/Regular_expression) engine will attempt to resolve a regular expression before it times out.
* Ability to define the culture for an [application domain](http://en.wikipedia.org/wiki/Application_domain).
* Console support for [Unicode](http://en.wikipedia.org/wiki/Unicode) ([UTF-16](http://en.wikipedia.org/wiki/UTF-16)) encoding.
* Support for versioning of cultural string ordering and comparison data.
* Better performance when retrieving resources.
* [Zip](http://en.wikipedia.org/wiki/Zip_(file_format)) compression improvements to reduce the size of a compressed file.
* Ability to customize a [reflection context](http://en.wikipedia.org/w/index.php?title=Reflection_context&action=edit&redlink=1) to override default [reflection](http://en.wikipedia.org/wiki/Reflection_(computer_programming)) behavior through the **CustomReflectionContext** class.

Asynchronous operations

In the .NET Framework 4.5, new asynchronous features were added to the C# and Visual Basic languages. These features add a task-based model for performing asynchronous operations.

* **ASP.NET**
* Support for new [HTML5](http://en.wikipedia.org/wiki/HTML5) form types.
* Support for model binders in [Web Forms](http://en.wikipedia.org/wiki/Web_Forms). These let you bind data controls directly to [data-access](http://en.wikipedia.org/w/index.php?title=Data-access&action=edit&redlink=1) methods, and automatically convert user input to and from [.NET Framework](http://en.wikipedia.org/wiki/.NET_Framework)[data types](http://en.wikipedia.org/wiki/Data_types).
* Support for unobtrusive [JavaScript](http://en.wikipedia.org/wiki/JavaScript) in client-side validation scripts.
* Improved handling of client script through [bundling](http://en.wikipedia.org/wiki/Bundle_(software_distribution)) and [minification](http://en.wikipedia.org/wiki/Minification_(programming)) for improved page performance.
* Integrated encoding routines from the [AntiXSS](http://en.wikipedia.org/w/index.php?title=AntiXSS&action=edit&redlink=1) library (previously an [external library](http://en.wikipedia.org/w/index.php?title=External_library&action=edit&redlink=1)) to protect from cross-site [scripting attacks](http://en.wikipedia.org/w/index.php?title=Scripting_attacks&action=edit&redlink=1).
* Support for [WebSocket](http://en.wikipedia.org/wiki/WebSocket) protocol.
* Support for reading and writing [HTTP](http://en.wikipedia.org/wiki/HTTP) requests and responses [asynchronously](http://en.wikipedia.org/wiki/Ajax_(programming)).
* Support for asynchronous modules and handlers.
* Support for [content distribution network](http://en.wikipedia.org/wiki/Content_distribution_network) ([CDN](http://en.wikipedia.org/wiki/Content_delivery_network)) fallback in the [Scrip tManager](http://en.wikipedia.org/w/index.php?title=ScriptManager&action=edit&redlink=1) control.
* Support Networking

1. Provides a new [programming interface](http://en.wikipedia.org/wiki/Programming_interface) for [HTTP](http://en.wikipedia.org/wiki/HTTP) applications: System.Net.Http namespace and System.Net.Http.Headers namespaces are added.
2. Other improvements:
   * Improved [internationalization](http://en.wikipedia.org/wiki/Internationalization) and [IPv6](http://en.wikipedia.org/wiki/IPv6) support.
   * RFC-compliant [URI](http://en.wikipedia.org/wiki/URI) support.
   * Support for [Internationalized Domain Name](http://en.wikipedia.org/wiki/Internationalized_Domain_Name) ([IDN](http://en.wikipedia.org/wiki/Internationalized_domain_name)) parsing.
   * Support for [Email Address Internationalization](http://en.wikipedia.org/wiki/Email_Address_Internationalization) ([EAI](http://en.wikipedia.org/wiki/Email_Address_Internationalization)).

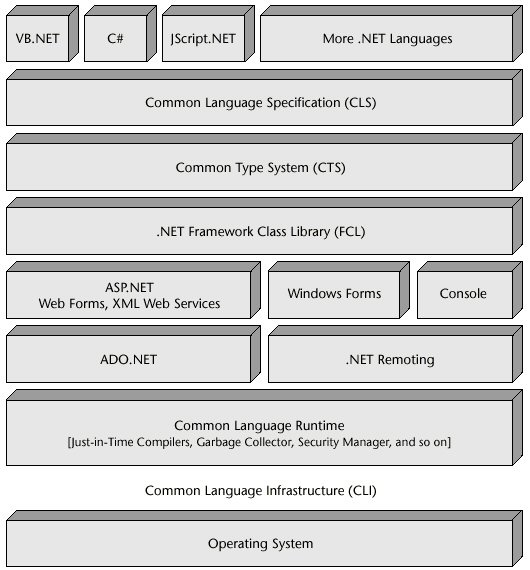
* Windows Presentation Foundation (WPF)
* The new [Ribbon](http://en.wikipedia.org/wiki/Ribbon_(computing)) control, which enables you to implement a ribbon user interface that hosts a Quick Access Toolbar, Application Menu, and tabs.
* The new [INotifyDataErrorInfo](http://en.wikipedia.org/w/index.php?title=INotifyDataErrorInfo&action=edit&redlink=1) interface, which supports [synchronous](http://en.wikipedia.org/wiki/Synchronization_(computer_science)) and asynchronous [data validation](http://en.wikipedia.org/wiki/Data_validation).
* New features for the [VirtualizingPanel](http://en.wikipedia.org/w/index.php?title=VirtualizingPanel&action=edit&redlink=1) and Dispatcher classes.
* Improved performance when displaying large sets of grouped data, and by accessing [collections](http://en.wikipedia.org/wiki/Collection_(computing)) on non-UI threads.
* [Data binding](http://en.wikipedia.org/wiki/Data_binding) to static properties, [data binding](http://en.wikipedia.org/wiki/Data_binding) to custom types that implement the [ICustomTypeProvider](http://en.wikipedia.org/w/index.php?title=ICustomTypeProvider&action=edit&redlink=1) interface and retrieval of data binding information from a binding expression.
* [Repositioning](http://en.wikipedia.org/w/index.php?title=Repositioning&action=edit&redlink=1) of data as the values change (live shaping).
* Better integration between [WPF](http://en.wikipedia.org/wiki/Windows_Presentation_Foundation) and [Win32](http://en.wikipedia.org/wiki/Win32) user interface components.
* Ability to check whether the [data context](http://en.wikipedia.org/w/index.php?title=Data_context&action=edit&redlink=1) for an item container is disconnected.
* Ability to set the amount of time that should elapse between property changes and [data source](http://en.wikipedia.org/wiki/Database) updates.
* Improved support for implementing [weak event patterns](http://en.wikipedia.org/w/index.php?title=Weak_event_pattern&action=edit&redlink=1). Also, events can now accept [markup extensions](http://en.wikipedia.org/w/index.php?title=Markup_extensions&action=edit&redlink=1).

| **Version No.** | **Release Date** | **Visual Studio Version** | **Default in Windows** | **CLR Version** | **Features in Release** |
| --- | --- | --- | --- | --- | --- |
| 1.0 | 13 Feb 2002 | Visual Studio .NET | NA | 1.0 | First Version of CLR and Base Class Library |
| 1.1 | 24 Apr 2003 | Visual Studio 2003 | Windows Server 2003 | 1.1 | 1. First Major version of .NET Framework  2. Built-in support for mobile ASP.NET controls. Previously available as an add-on for .NET Framework, now part of the framework  3. Security changes – enable Windows Forms assemblies to execute in a semi-trusted manner from the Internet, and enable Code Access Security in ASP.NET applications  4. Built-in support for ODBC and Oracle databases. Previously available as an add-on for .NET Framework 1.0, now part of the framework  5. .NET Compact Framework – a version of the .NET Framework for small devices  6. Internet Protocol version 6 (IPv6) support |
| 2.0 | 7 Nov 2005 | Visual Studio 2005 | Windows Server 2003 R2 | 2.0 | 1.  Generics  2.  Language support for generics built directly into the .NET CLR  3.  Full 64-bit support for both the x64 and the IA-64 hardware platforms   4.  SQL Server integration – .NET 2.0, VS 2005, and SQL Server 2005 are all tied together. This means that instead of using T-SQL, one can build stored procedures and triggers in any of the .NET-compatible languages  5.  A new hosting API for native applications wishing to host an instance of the .NET runtime. The new API gives a fine grain control on the behavior of the runtime with regards to multithreading, memory allocation, assembly loading and more  6.  Many additional and improved ASP.NET web controls  7.  New data controls with declarative data binding  8.  New personalization features for ASP.NET, such as support for themes, skins, master pages and webparts  9.  .NET Micro Framework – a version of the .NET Framework related to the Smart Personal Objects Technology initiative  10. Membership provider  11. Partial classes  12. Nullable types  13. Anonymous methods  14. Iterators  15. Data tables |
| 3.0 | 6 Nov 2006 | Visual Studio 2005 | Windows Vista, Windows Server 2008 | 2.0 | 1. Windows Presentation Foundation (WPF), a new user interface subsystem and API based on XML and vector graphics, which uses 3D computer graphics hardware and Direct3D technologies  2. Windows Communication Foundation (WCF), a service-oriented messaging system which allows programs to interoperate locally or remotely similar to web services  3. Windows Workflow Foundation (WF) allows for building of task automation and integrated transactions using workflows  4. Windows CardSpace, a software component which securely stores a person’s digital identities and provides a unified interface for choosing the identity for a particular transaction, such as logging in to a website |
| 3.5 | 19 Nov 2007 | Visual Studio 2008 | Windows 7, Windows Server 2008 R2 | 2.0, 2.0 SP1, 3.0 SP1 | 1. Added new features such as AJAX-enabled Web sites and LINQ  2. Source code of Base Class Library (BCL) has been partially released  3. New .NET Compact Framework 3.5 released |
| 3.5 SP1 | 11 Aug 2008 | Visual Studio 2008 | Windows 7, Windows Server 2008 R2 | 3.0 SP2 | 1. .NET Framework Client Profile   2. Dynamic Data  3. Two new data service components added, ADO.NET Entity Framework and ADO.NET Data Services  4. Two new assemblies for web development, System.Web.Abstraction and System.Web.Routing  5. New set of controls “Visual Basic Power Packs” introduced |
| 4.0 | 12 Apr 2010 | Visual Studio 2010 | NA | 4.0 | 1. New Version of CLR  2. Parallel Extensions to improve support for parallel computing, which target multi-core or distributed systems. To this end, technologies like PLINQ (Parallel LINQ), a parallel implementation of the LINQ engine, and Task Parallel Library, which exposes parallel constructs via method calls are included  3. New Visual Basic .NET and C# language features, such as implicit line continuations, dynamic dispatch, named parameters, and optional parameters  4. Code Contracts  5. Inclusion of new types to work with arbitrary-precision arithmetic (System.Numerics.BigInteger) and complex numbers (System.Numerics.Complex)  6. Dynamic Language Runtime (DLR)  7. Managed Extensibility Framework (MEF)  8. Windows Server AppFabric for application server capabilities in the form of AppFabric hosting and in-memory distributed caching support |
| 4.5 | 15 Aug 2012 | Visual Studio 2012 | Windows 8, Windows Server 2012 | 4.5 | 1. Windows Store Apps  2. The Portable Class Library project in Visual Studio 2012  3. Background Garbage Collection Servers  4. Asynchronous File Operations  5. Regex Timeout  6. Profile Optimization – Improved Startup Performance 7. Zip Compression 8. Async and Await – Code Markers |

PRACTICAL 2

Study of .NET Architecture

.NET is tiered, modular, and hierarchal. Each tier of the .NET Framework is a layer of abstraction. .NET languages are the top tier and the most abstracted level. The common language runtime is the bottom tier, the least abstracted, and closest to the native environment. This is important since the common language runtime works closely with the operating environment to manage .NET applications. The .NET Framework is partitioned into modules, each with its own distinct responsibility. Finally, since higher tiers request services only from the lower tiers, .NET is hierarchal. The architectural layout of the .NET Framework is illustrated in Figure.



The .NET Framework is a managed environment. The common language runtime monitors the execution of .NET applications and provides essential services. It manages memory, handles exceptions, ensures that applications are well-behaved, and much more. Language interoperability is one goal of .NET. .NET languages share a common runtime (the common language runtime, a common class library), the Framework Class Library (FCL), a common component model, and common types. In .NET, the programming language is a lifestyle choice. Except for subtle differences, C#, VB.NET, or JScript.NET offer a similar experience. .NET abstracts lower-level services, while retaining most of their flexibility. This is important to C based programmers, who shudder at the limitations presented in Visual Basic 6 and earlier.

* **Common language runtime :**

The common language runtime is the engine of .NET and the common runtime of all managed languages. In addition, as the final layer resting atop of the operating environment, the CLR provides the first level of abstraction. Since assemblies run within the context of the common language runtime, they are independent of the underlying operating environment and hardware. Assemblies or managed code are portable to any environment offering a .NETcompliant common language runtime, as defined by the Common Language Infrastructure (CLI). Of the many services offered by the common language runtime, we will focus

on the two most important: code execution and memory management.

* **Common Language Specification (CLS)**:

CLS is a set of specifications or guidelines defining a .NET language. Shared specifications promote language interoperability. For example, CLS defines the common types of managed languages, which is a subset of the Common Type System (CTS). This removes the issue of marshaling, a major impediment when working between two languages.

* **The Common Type System (CTS)**:

CTS is a catalog of .NET types—System.Int32,System.Decimal, System.Boolean, and so on. Developers are not required to use these types directly. These types are the underlying objects of the specific data types provided in each managed language.The contributions of CTS extend well beyond the definitions of common data types. CTS helps with type safeness, enhances language interoperability, aids in segregating application domains, and more.

* **NET Framework Class Library (FCL):**

The .NET Framework Class Library (FCL) is a set of managed classes that provide access to system services. File input/output, sockets, database access, remoting, and XML are just some of the services available in the FCL. Importantly, all the .NET languages rely on the same managed classes for the same services. This is one of the reasons that, once you have learned any .NET language, you have learned 40 percent of every other managed language. The same classes, methods, parameters, and types are used for system services regardless of the language. This is one of the most important contributions of FCL.

* **ASP.Net :**

ASP.NET is used to create dynamic Web applications and is the successor to ASP.Microsoft emphasizes scalability in ASP.NET. Free threads boost responsiveness and prevent internal bottlenecks. ASP.NET uses ADO.NET, server-side controls, and other techniques to promote a highly distributed and scalablemodel. Also, ASP.NET hosts Web applications in application domains within the worker process (aspnet\_wp.exe) to heighten performance and lower overhead. Finally, ASP.NET uses compiled pages instead of interpreted pages to improve performance.

* **Web Forms :**

Web Forms, is the forms generator for Web applications in ASP.NET and replaces Visual Interdev. Web Forms closely resembles Windows Forms or the Visual Basic 6 forms engine, one more consideration that helps developers move between client and Web programming. A variety of server and client-side controls are supported. Web Forms controls are server-side controls and typically more complex than HTML controls. The calendar control typifies a Web Form control that maps to multiple tags. HTML controls by default are client-side controls and map to a single tag. Web controls are instances of managed classes that write HTML tags.

* **Web Services:**

Web services are the basis of the programmable Web and distributed applications that transcend hardware and operating environments.Web services promote remote function calls over the Internet.

* **Windows Forms:**

Windows Forms is the form generator for client-side applications and is similar to the forms engine of Visual Basic 6.Windows Forms is primarily a code generator, generating managed classes for forms, buttons, text boxes, menus, and other graphical user interface elements.

* **Console Applications:**

Console applications have been available to C-based, but not to Visual Basic, programmers. In .NET, console applications are available to all managed languages. Console applications are useful for logging, instrumentation, and other text-based activities.

* **ADO.NET:**

ADO.NET accentuates disconnected data manipulation, is highly scalable, integrates open standards, and is perfected for Web application development. ADO.NET offers managed providers for Microsoft SQL and OLE DB databases and is a set of managed classes in the System.Data namespace. The System.Data.SqlClient namespace contains classes related to Microsoft SQL, while System.Data.OleDb encompasses classes pertaining to OLE DB providers.

* **.NET Remoting:**

.NET Remoting is a second option for remoting objects in the .NET Framework.

The first mentioned was XMLWeb services. .NET Remoting is similar to

Web services conceptually. However, with .NET Remoting the developer chooses the transmission protocol, data protocol, data port, and other aspects of the remoting architecture necessary to open a channel for client-server communication. In essence, a developer is setting the specifications of the remoting infrastructure.

PRACTICAL 3

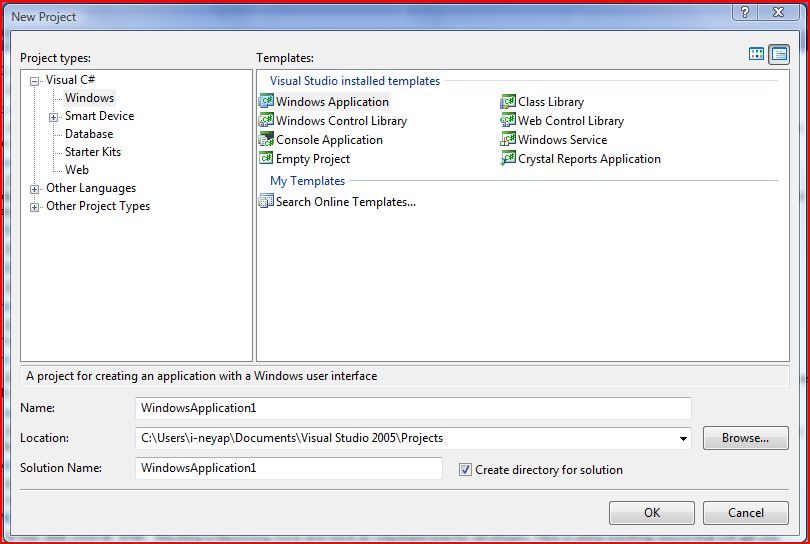
Introduction to IDE Visual Studio 2005

Visual Studio 2005 (VS 2005) is an integrated IDE enabling development of C++, C#, and Visual Basic and J #applications under one roof. It includes numerous features that help in building windows applications with less effort. The IDE is designed primarily for building applications with any .NET languages. The IDE includes features like an automatic build utility, a syntax highlighter, an easy-to-use debugger and many more. This guide introduces building a C# application using VS.Net 2005 by a simple hands-on example. First, let's start our VS.Net 2005 IDE. The following screen shown in Figure 1 below will be displayed when the IDE is started.



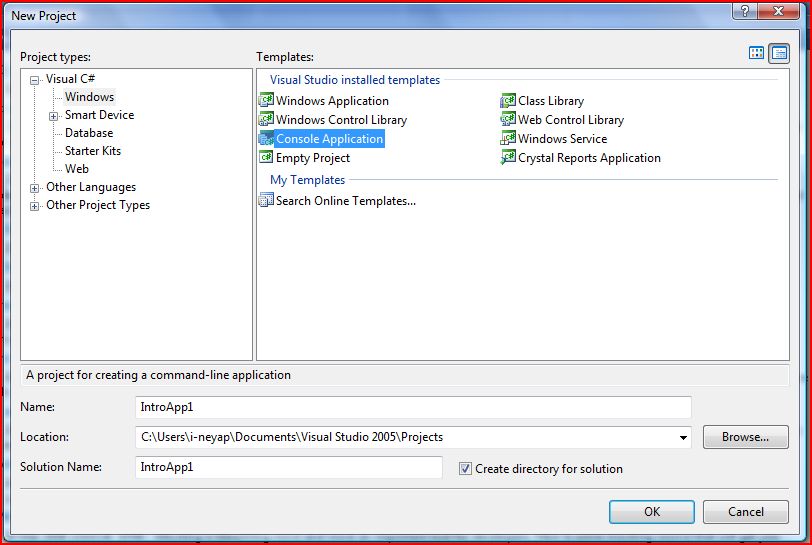
***Figure 1. Initial Screen***

Let's build a simple Hello World application. Click 'File > New > Project' from the menu bar. The following dialog box will appear.



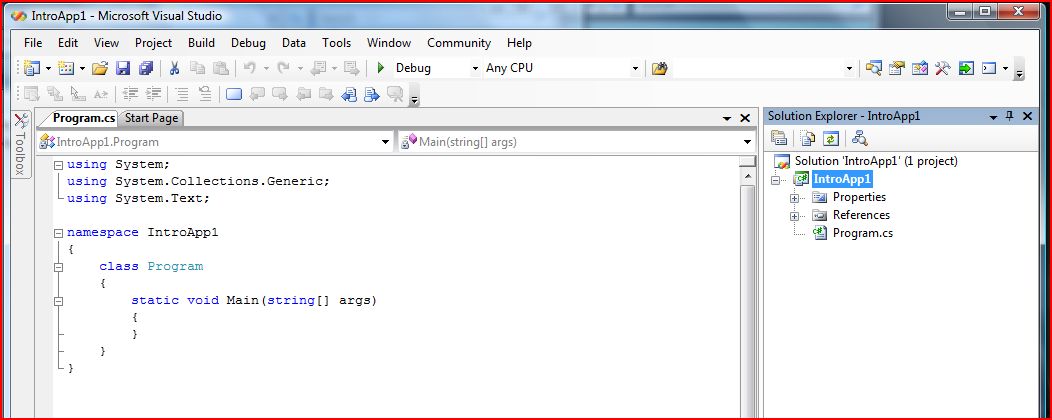
***Figure 2. New Project Dialog***

For this course, projects must be implemented as a C#  application. To start building a C# Console Application follow the following steps. On the category column on the left select 'Visual C#  > Windows'. Next, choose 'Console Application' and type 'IntroApp' as your project name. Also, choose a location to save your application. The above steps are illustrated by Figure 3 below.



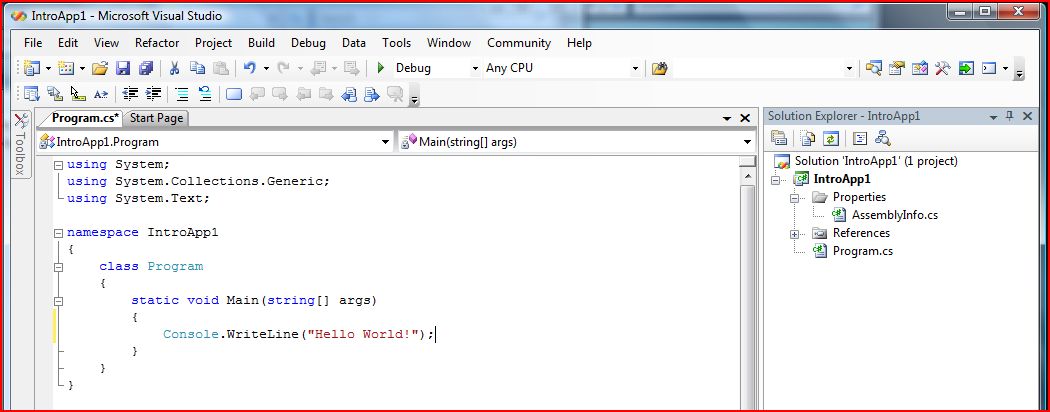
***Figure 3. New C# Console Project***

Click 'OK'. Another dialog will appear. One can specify additional settings through this dialog, however for this simple application simply accept the default settings by clicking 'Finish'. The following will be shown in your VS 2005 IDE after performing the above steps.



***Figure 4. IntroApp Project***

An application developed in VS.Net 2005 is organized into a hierarchy: solution, project and files. A solution contains one or more project(s); a project contains one or more file(s). Most files are either source files (.cs) or other reference files. Double-click the main source file of our IntroApp project (i.e. Program.cs) in the *Solution Explorer* to view its content in the *Code Editing window*. Enter the following C# codes as shown in Figure 5 below.



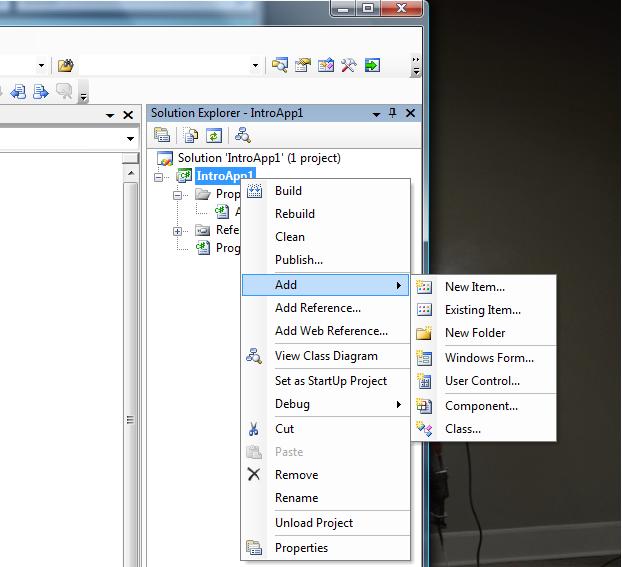
***Figure 5. A Simple Hello World Application***

To build the application press F6, alternatively click the Build Solution (http://www.comp.nus.edu.sg/~cs3215/tools/vsintro_files/vsintr6.gif) toolbar. To run the application press F5, alternatively click the Start toolbar (http://www.comp.nus.edu.sg/~cs3215/tools/vsintro_files/vsintr7.gif). When the application is run, the following console will appear.



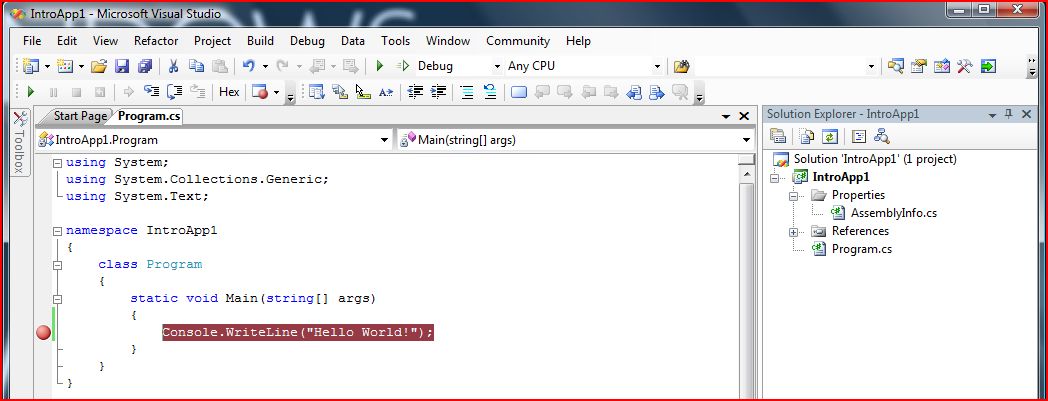
***Figure 6. Running Hello World***

Voila! You've got a C# application running. The executable of your application is saved by default to the '$ProjectDir$/Debug' folder.

To add files to your project, right click the *Project*folder in the *Project window* as shown below. Select 'New Item' and proceed accordingly.

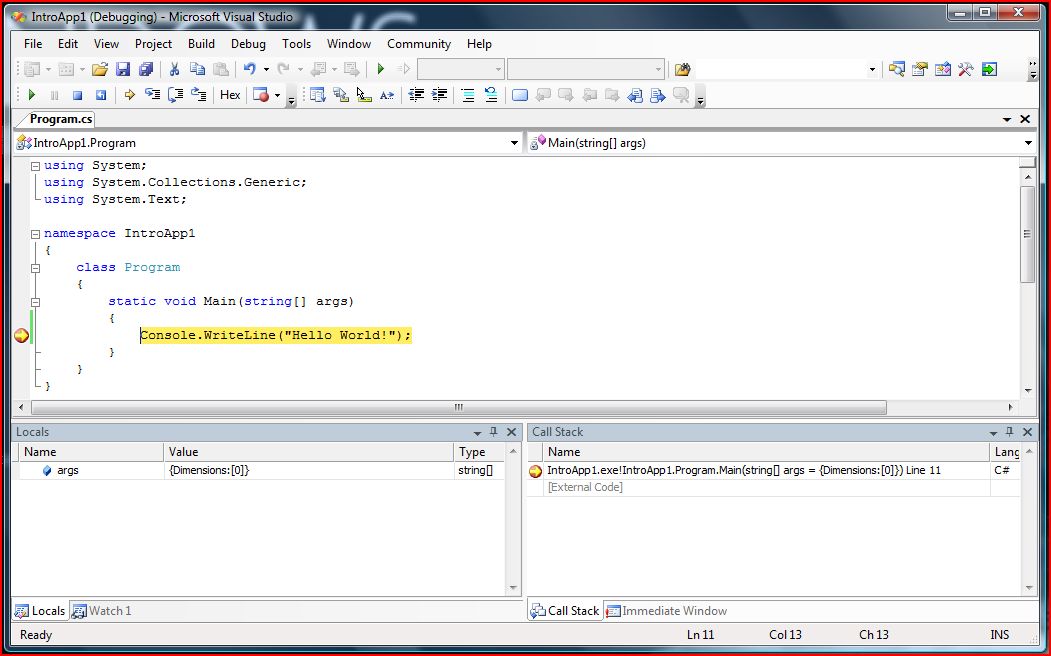
***Figure 7. Add Existing File***

Now let's see how debugging is performed using VS 2005 IDE. Click at the margin besides the ‘Console.WriteLine(“Hello World!”);' line in the *Code Editing* window. A red bullet will be shown at the margin -- this bullet signifies a breakpoint.



***Figure 8. Insert Breakpoint***

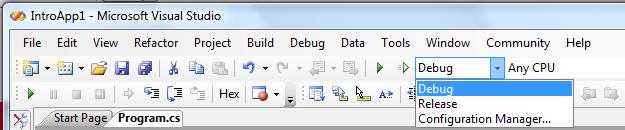
Press F5 or click the Start (http://www.comp.nus.edu.sg/~cs3215/tools/vsintro_files/vsintr7.gif) toolbar. The program will run and stop at the breakpoint we have just set. Note the *Locals window*at the bottom of the IDE. The window provides information of the accessible variables and their values at the breakpoint. At the breakpoint, we have 1 active variables (args) which is a local variable and parameters to the main function.



***Figure 9. Debugging In Process***

To continue, press F10 or alternatively click the Step Over (http://www.comp.nus.edu.sg/~cs3215/tools/vsintro_files/vsintr12.gif) toolbar to step over to the next statement (end of program). To continue to the next breakpoint or to the end of the program press F5 or alternatively click the Start (http://www.comp.nus.edu.sg/~cs3215/tools/vsintro_files/vsintr7.gif)) toolbar again.

There are two modes of compilation in VS 2005 namely 'Debug' and 'Release'. Debug mode let you insert breakpoints and debug your code. Release mode optimize your code but doesn't allow debugging. When your program is fully debugged, to let it run faster, compile it using the Release mode. One can switch to release mode by selecting 'Release' from the combo box at the top of the IDE as shown in Figure 10 below.



***Figure 10. Switching Compilation Mode***

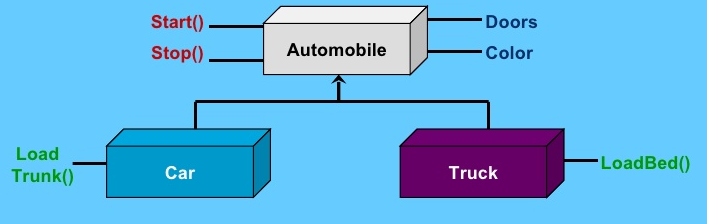
For the final evaluation, you are required to compile your application in Release mode. Due to poor programming practices, it is possible that your application runs well when compiled under Debug mode but not under Release mode. Hence, during the various stages of your project ensure that your application runs well when compiled under both Debug and Release modes.

PRACTICAL :04

Study of language features and comparison of VB.NET and C#

**FEATURES OF VB.NET**

* Inheritance(object-oriented language)
* VB.NET contains full support for inheritance
* Inheritance allows developers to extend an existing code base to add new functionality



* Type-safe
* VB.NET is a type safe language :code can only access memory locations it is allowed to access.
* Benefits of type-safe code:

1. Ability to isolate assemblies
2. Increased security
3. Unwanted side effects eliminated

* VB.NET can require explicit type casting so that types are not automatically converted to other types.
* Structured exception handling (Try..Catch)

Structured exception handling is an integral part of VB.NET :

Eg:

Try

InsertRecord(‘jdoe@hotmail.com’,’John’,’Doe’)

Catch exp as System.Exception

LogError(exp.Message)

End Try

* Method Overloading/overriding

Method overloading

* VB.NET fully supports method overloading
* Overloading allows different signatures to be applied to methods with the same name:

Public Overloads Function GetCustomer (id as Integer) as Customer

//Return Customer object based upon id’

End Function

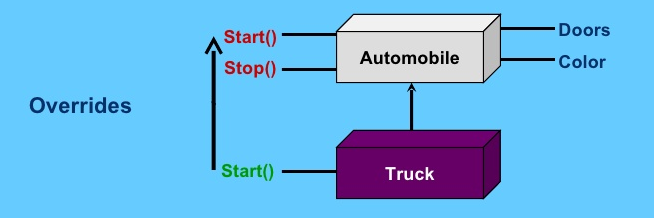
Public Overloads Function GetCustomer (fname as String,lname as String) as Customer

//Return Customer object based upon first and last name’

End Function

Method overriding

* Method overriding allows child objects to override behaviours found in the parent(base).



* Parameterized constructors
* VB.NET object constructors can have multiple parameters passed in allowing for greater flexibility when creating objects:
* //default constructor

Public Sub new()

MyBase.new() //call base class constructor

/\*Initialization of objects can occur here \*/

End Sub

Public Sub new(ByVal fname as String , ByVal lname as String)

Me.new() //call default constructor

Fname=fname

Lname=lname

End Sub

* Delegates And Events

Events

* VB.NET has a robust architecture in place for handling events.As a result,understanding how events work is crucial to developing successful VB.NET applications.



* Events are defined by using the Event keyword:

Public Event WorkPerformed(amount as Integer,workType as String)

* Events are raised using the RaiseEvent keyword:

RaiseEvent WorkPerformed(5, “Write code”)

Delegates

* A delegate type provides a type-safe pointer to the function and can be used to hook an event to an event handler.



Public Sub WorkPerformed\_Handler(amount as Integer, workType as String) Handles Worker.WorkPerformed

// Handling of event data occurs here

End Sub

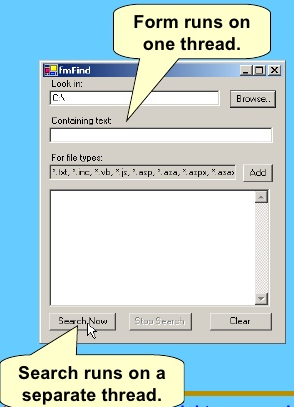
* Attributes
* Attributes are descriptive tags that can be added into VB.NET code to provide additional information(metadata) about what the code does:
* <WebMethod()>

Public Function GetUserPassword(email as string) as String

Dim golfer as GolferInfo =new GolferInfo(email)

Return golfer.Password End Function

* Multi Threading
* VB.NET developers can now create true multithreaded applications.
* Threading allows different processes to run separately without tying up the user interface or program.
* Threading functionality is located in the System.Threadingnamespace.



**FEATURES OF C# :**

* There are no global variables or functions. All methods and members must be declared within classes. Static members of public classes can substitute for global variables and functions.
* Local variables cannot shadow variables of the enclosing block, unlike C and C++. Variable shadowing is often considered confusing by C++ texts.
* C# supports a strict Boolean datatype, bool. Statements that take conditions, such as while and if, require an expression of a type that implements the true operator, such as the boolean type. While C++ also has a boolean type, it can be freely converted to and from integers, and expressions such as if(a) require only that a is convertible to bool, allowing a to be an int, or a pointer. C# disallows this "integer meaning true or false" approach on the grounds that forcing programmers to use expressions that return exactly bool can prevent certain types of common programming mistakes in C or C++ such as if (a = b) (use of assignment = instead of equality ==).
* In C#, memory address pointers can only be used within blocks specifically marked as unsafe, and programs with unsafe code need appropriate permissions to run. Most object access is done through safe object references, which always either point to a "live" object or have the well-defined null value; it is impossible to obtain a reference to a "dead" object (one which has been garbage collected), or to a random block of memory. An unsafe pointer can point to an instance of a value-type, array, string, or a block of memory allocated on a stack. Code that is not marked as unsafe can still store and manipulate pointers through the System.IntPtr type, but it cannot dereference them.
* Managed memory cannot be explicitly freed; instead, it is automatically garbage collected. Garbage collection addresses the problem of memory leaks by freeing the programmer of responsibility for releasing memory which is no longer needed.
* In addition to the try...catch construct to handle exceptions, C# has a try...finally construct to guarantee execution of the code in the finally block.  
  Multiple inheritance is not supported, although a class can implement any number of interfaces. This was a design decision by the language's lead architect to avoid complication and simplify architectural requirements throughout CLI.
* C# is more type safe than C++. The only implicit conversions by default are those which are considered safe, such as widening of integers. This is enforced at compile-time, during JIT, and, in some cases, at runtime. There are no implicit conversions between booleans and integers, nor between enumeration members and integers (except for literal 0, which can be implicitly converted to any enumerated type). Any user-defined conversion must be explicitly marked as explicit or implicit, unlike C++ copy constructors and conversion operators, which are both implicit by default.
* Enumeration members are placed in their own scope.
* C# provides properties as syntactic sugar for a common pattern in which a pair of methods, accessor (getter) and mutator (setter) encapsulate operations on a single attribute of a class.
* Full type reflection and discovery is available.
* C# currently (as of version 4.0) has 77 reserved words.
* Checked exceptions are not present in C# (in contrast to Java). This has been a conscious decision based on the issues of scalability and versionability.

**Features of Visual Basic .NET not found in C#**

* Variables can be declared using the WithEvents construct. This construct is available so that a programmer may select an object from the *Class Name' drop down list and then select a method from the* Declarations *drop down list to have the*[*Method signature*](http://en.wikipedia.org/wiki/Method_signature)*automatically inserted*
* Auto-wireup of events, VB.NET has the Handles syntax for events
* [Marshalling](http://en.wikipedia.org/wiki/Marshalling_(computer_science)) an object for multiple actions using an unqualified dot reference. This is done using the With ... End With structure
* [IsNumeric](http://en.wikipedia.org/wiki/Is_functions) evaluates whether a string can be cast into a numeric value (the equivalent for C# requires using int.TryParse)
* XML Literals
* Inline date declarations by using #1/1/2000# syntax (M/dd/yyyy).
* Module (although C#'s sealed static classes with additional semantics, but each field has to individually be declared as static)
* Members of Modules imported to the current file, can be accessed with no preceding container accessor
* The [My](http://msdn.microsoft.com/en-us/library/5btzf5yk.aspx) namespace
* Conversion of Boolean value True to Integer may yield -1 or 1 depending on the conversion use.

Assigning and comparing variables uses the same token, =. Whereas C# has separate tokens, == for comparison and = to assign a value

* VB.NET is not case-sensitive.
* When assigning a value to a variable with a different [data type](http://en.wikipedia.org/wiki/Data_type), VB.NET will [coerce](http://en.wikipedia.org/wiki/Type_conversion) the value if possible. This automatic coercion can sometimes lead to unexpected results, for example:

**Dim** i **As** **Integer** = "1" 'Compiler automatically converts String to Integer

**Dim** j **As** **String** = 1 'Compiler automatically converts Integer to String

**If** i = j **Then** 'Compiler does cast and compare between i and j

MsgBox("Avoid using, but this message will appear!") 'MsgBox() can be used instead of MessageBox.show()

**End** **If**

* Val() function which also parses a null value while converting into double (In c# Convert.ToDouble() is used to convert any object into double type value, but which throws exception in case of a null value)
* CInt, CStr, CByte, CDbl, CBool, CByte, CDate, CLng, CCur, CObj and a wide variety of converting functions built in the language
* COM components and interoperability was more powerful in VB.NET as the Object type is bound at runtime, however C# 4.0 added the dynamic type which functions as a late bound form of Object.
* Namespaces can be imported in project level, so they don't have to be imported to each individual file, like C#
* In-line exceptions filtering by a Boolean expression, using "When expression" blocks. C# emulates this functionality, using a catch block followed by an if block. It is not just syntax sugar.

**Features of C# not found in Visual Basic .NET**

By default, numeric operations are not checked. This results in slightly faster code, at the risk that numeric overflows will not be detected. However, the programmer can place arithmetic operations into a checked context to activate overflow checking. (It can be done in Visual Basic by checking an option)

* In Visual Basic .NET property methods may take parameters.
* C# is case-sensitive.
* Allows blocks of unsafe code (like C++/CLI) via the unsafe keyword and support for pointers
* Partial Interfaces
* Multi-line comments (note that the Visual Studio IDE supports multi-line commenting for Visual Basic .NET)
* Static classes (Classes which cannot contain any non-static members, although VB's Modules are essentially sealed static classes with additional semantics)
* Can use checked and unchecked contexts for fine-grained control of overflow/underflow checking

A few keywords have very different versions in Visual Basic and C#:

* Friend vs internal - access modifiers allowing inter-class but not intra-assembly reference
* Me vs this - a self-reference to the current object instance
* MustInherit vs abstract - prevents a class from being directly instantiated, and forces consumers to create object references to only derived classes
* MustOverride vs abstract - for forcing derived classes to override this method
* MyBase vs base - for referring to the base class from which the current class is derived
* NotInheritable vs sealed - for declaring classes that may not be inherited
* NotOverridable vs sealed - for declaring methods that may not be overridden by derived classes
* Overridable vs virtual - declares a method as being able to be overridden in derived classes
* Shared vs static - for declaring methods that do not require an explicit instance of an object

### Comments

|  |  |
| --- | --- |
| [**C#**](http://en.wikipedia.org/wiki/C_Sharp_(programming_language)) | [**Visual Basic .NET**](http://en.wikipedia.org/wiki/Visual_Basic_.NET) |
| *// Single line comment*    */\* Multi-line comment*  *line 2*  *line 3 \*/*    */// XML single line comment*    */\* \*XML multi-line comment*  *line 2*  *line 3 \*/* | *' Single line comment*    *''' XML 1st line comment*  *‘'' XML 2nd line comment*  *''' XML 3rd line comment*  Multi-line comments not available |

### Conditionals

|  |  |
| --- | --- |
| [**C#**](http://en.wikipedia.org/wiki/C_Sharp_(programming_language)) | **Visual Basic .NET** |
| **if** (condition) {  *// condition is true*  } **else** **if** (othercondition) {  *// condition is false and othercondition is true*  } **else** {  *// condition and othercondition are false*  } | **If** condition **Then**  *' condition is true*  **ElseIf** othercondition **Then**  *' condition is false and othercondition is true*  **Else**  *' condition and othercondition false*  **End** **If** |

### Loops

|  |  |
| --- | --- |
| [**C#**](http://en.wikipedia.org/wiki/C_Sharp_(programming_language)) | **Visual Basic .NET** |
| **for** (**int** i = 0; i < number; ++i) {  *// loop from zero up to one less than number*  } | **For** i **As** Integer = 0 **To** number - 1  *' loop from zero up to one less than number*  **Next** |
| **for** (**int** i = number; i >= 0; --i) {  *// loops from number down to zero*  } | **For** i **As** Integer = number **To** 0 **Step** -1  *' loops from number down to zero*  **Next** |
| **break**; *//breaks out of a loop* | **Exit** **For** *'breaks out of a for loop*  **Exit** **While** *'breaks out of a while loop*  **Exit** **Do** *'breaks out of a do loop* |

### Comparers

#### Primitive types

|  |  |
| --- | --- |
| [**C#**](http://en.wikipedia.org/wiki/C_Sharp_(programming_language)) | **Visual Basic .NET** |
| **if** (a == b) {  *// equal*  } | **If** a = b **Then**  *' equal*  **End** **If** |
| **if** (a != b) {  *// not equal*  }  Or:  **if** (!(a == b)) {  *// not equal*  } | **If** a <> b **Then**  *' not equal*  **End** **If**  Or:  **If** **Not** a = b **Then**  *' not equal*  **End** **If** |  |
| **if** (a == b & c == d | e == f) {  *// multiple comparisons*  } | **If** a = b **And** c = d **Or** e = f **Then**  *' multiple comparisons*  **End** **If** |  |
| **if** (a == b && c == d || e == f) {  *// short-circuiting comparisons*  } | **If** a = b **AndAlso** c = d **OrElse** e = f **Then**  *' short-circuiting comparisons*  **End** **If** |  |
|  |  |  |

#### Object types

|  |  |
| --- | --- |
| [**C#**](http://en.wikipedia.org/wiki/C_Sharp_(programming_language)) | **Visual Basic .NET** |
| **if** (**object**.ReferenceEquals(a, b)) {  *// variables refer to the same instance*  } | **If** a **Is** b **Then** *'Can also be written as If Object.ReferenceEquals(a, b) Then*  *' variables refer to the same instance*  **End** **If** |
|  |  |
| **if** (!**object**.ReferenceEquals(a, b)) {  *// variables do not refer to the same instance* } | **If** a **IsNot** b **Then**  *' variables do not refer to the same instance*  **End** **If** |
|  |  |
| **if** (a == b) {  *// instances are equivalent*  } | **If** a = b **Then**  *' instances are equivalent*  **End** **If** |
|  |  |
| **if** (a != b) {  *// not equivalent*  } | **If** a <> b **Then**  *' not equivalent*  **End** **If** |
|  |  |
| **var** type = typeof(**int**); | **Dim** type = **GetType**(Integer) |
|  |  |
| **if** (a is b) {  *// types of a and b are compatible*  } | **If** **TypeOf** a **Is** b **Then**  *' types of a and b are compatible*  **End** **If** |
|  |  |
| **if** (!(a is b)) {  *// types of a and b are not compatible*  } | **If** **Not** **TypeOf** a **Is** b **Then**  *' types of a and b are not compatible*  **End** **If** |

PRACTICAL : 05

Write a program for Arithmetic Calculator using Console Application.

Program :

Module Module1

Sub Main()

Dim a As Integer, b As Integer, c As Integer

Dim op As String

System.Console.Write("ENTER NO.1 :")

a = Val(System.Console.ReadLine())

System.Console.Write("ENTER NO.2 :")

b = Val(System.Console.ReadLine())

System.Console.WriteLine(" HERE ARE THE OPERATORS PLEASE ENTER THE OPERATOR ...")

System.Console.WriteLine("1. addition + ")

System.Console.WriteLine(" 2. subtraction -")

System.Console.WriteLine(" 3.multiplication \*")

System.Console.WriteLine(" 4. division /")

System.Console.Write(" OPERATOR : ")

op = System.Console.ReadLine()

Select Case op

Case "+"

c = a + b

System.Console.WriteLine(c)

Exit Select

Case "-"

c = a - b

System.Console.WriteLine(c)

Exit Select

Case "\*"

c = a \* b

System.Console.WriteLine(c)

Exit Select

Case "/"

Try

c = a / b

System.Console.WriteLine(c)

Catch ex As Exception

System.Console.WriteLine(ex.ToString)

End Try

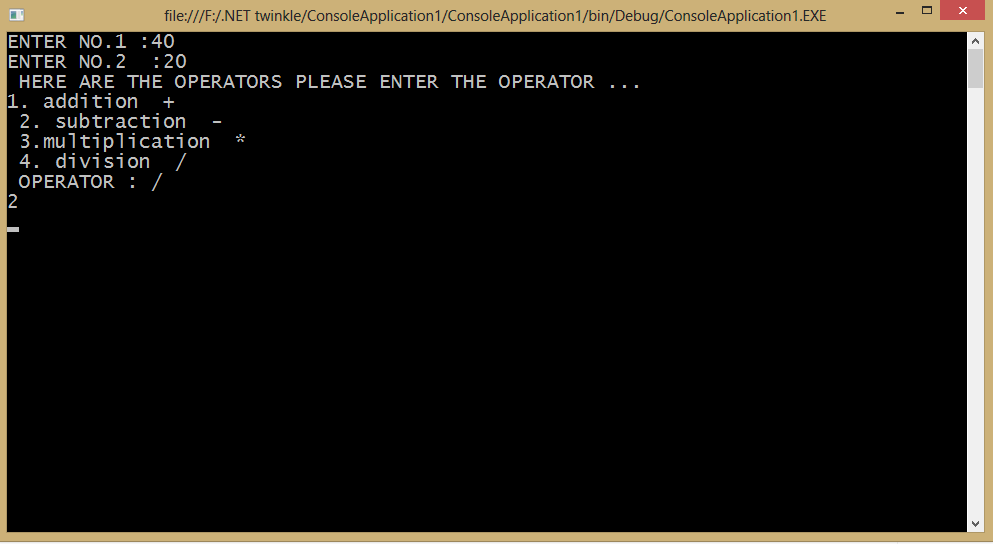
End Select

System.Console.Read()

End Sub

End Module

OUTPUT :



PRACTICAL : 06

Implement Windows Form based application using controls like Menus,Dialog and ToolTip.

Program :

Public Class Form1

Private Sub NewToolStripMenuItem\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles NewToolStripMenuItem.Click

Dim a As New Form1

a.Show()

Me.Hide()

End Sub

Private Sub txt1\_Leave(ByVal sender As Object, ByVal e As System.EventArgs)

If (txt1.Text = "") Then

MessageBox.Show("error! enter text in textbox")

End If

End Sub

Private Sub OpenToolStripMenuItem\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles OpenToolStripMenuItem.Click

If (OpenFileDialog1.ShowDialog() <> Windows.Forms.DialogResult.Cancel) Then

OpenFileDialog1.Filter = "vb.net files|\*.vb |all files | \*.\*"

Dim sFileName As String

sFileName = OpenFileDialog1.FileName

MessageBox.Show(" Opened file is " & sFileName)

End If

End Sub

Private Sub SaveToolStripMenuItem\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles SaveToolStripMenuItem.Click

If (SaveFileDialog1.ShowDialog <> Windows.Forms.DialogResult.Cancel) Then

SaveFileDialog1.Filter = "vb.net files|\*.vb|All files|\*.\*"

Dim fFileName As String

fFileName = SaveFileDialog1.FileName

MessageBox.Show("file" & fFileName & "is saved")

End If

End Sub

Private Sub ExitToolStripMenuItem\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles ExitToolStripMenuItem.Click

Me.Close()

End Sub

Private Sub UndoToolStripMenuItem\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles UndoToolStripMenuItem.Click

txt1.Undo()

End Sub

Private Sub CutToolStripMenuItem\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles CutToolStripMenuItem.Click

txt1.Cut()

End Sub

Private Sub CopyToolStripMenuItem\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles CopyToolStripMenuItem.Click

txt1.Copy()

End Sub

Private Sub PasteToolStripMenuItem\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles PasteToolStripMenuItem.Click

txt2.Paste()

End Sub

Private Sub ChangeFontToolStripMenuItem\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles ChangeFontToolStripMenuItem.Click

If (FontDialog1.ShowDialog <> Windows.Forms.DialogResult.Cancel) Then

txt1.Font = FontDialog1.Font

End If

End Sub

Private Sub ColorToolStripMenuItem\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles ColorToolStripMenuItem.Click

If (ColorDialog1.ShowDialog <> Windows.Forms.DialogResult.Cancel) Then

txt2.ForeColor = ColorDialog1.Color

End If

End Sub

Private Sub ToolStripButton1\_Click\_1(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles ToolStripButton1.Click

pb1.Visible = True

Dim tFileName As String

OpenFileDialog1.InitialDirectory = "C:\"

OpenFileDialog1.Filter = "jpeg|\*.jpg|gifs|\*.gif|Bitmap|\*.bmp"

Dim d As Integer = OpenFileDialog1.ShowDialog()

If d <> DialogResult.Cancel Then

tFileName = OpenFileDialog1.FileName

pb1.Image = Image.FromFile(tFileName)

OpenFileDialog1.Reset()

End If

End Sub

Private Sub ToolStripComboBox1\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles con1.Click

Dim s As String

s = con1.Text

Select Case (s)

Case "Textbox"

TextBox1.Visible = True

Exit Sub

Case "RadioButton"

RadioButton1.Visible = True

Exit Sub

Case "Checkbox"

CheckBox1.Visible = True

Exit Sub

Case "Button"

Button1.Visible = True

Exit Sub

Case "default"

RadioButton1.Visible = False

CheckBox1.Visible = False

TextBox1.Visible = False

Button1.Visible = False

End Select

End Sub

Private Sub PrintToolStripMenuItem\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles PrintToolStripMenuItem.Click

PrintDialog1.Document = PrintDocument1

PrintDialog1.PrinterSettings = PrintDocument1.PrinterSettings

PrintDialog1.AllowSomePages = True

If (PrintDialog1.ShowDialog = Windows.Forms.DialogResult.OK) Then

PrintDocument1.PrinterSettings = PrintDialog1.PrinterSettings

PrintDocument1.Print()

End If

End Sub

Private Sub PrintPreviewToolStripMenuItem\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles PrintPreviewToolStripMenuItem.Click

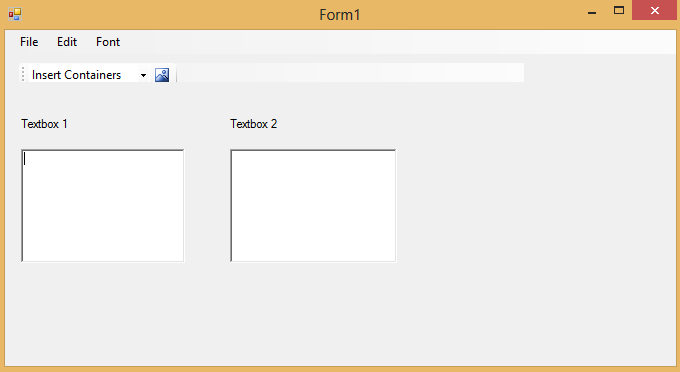
PrintPreviewDialog1.ShowDialog()

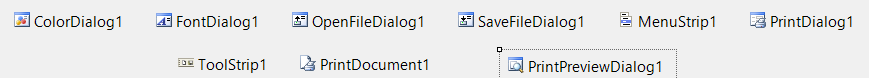
End Sub

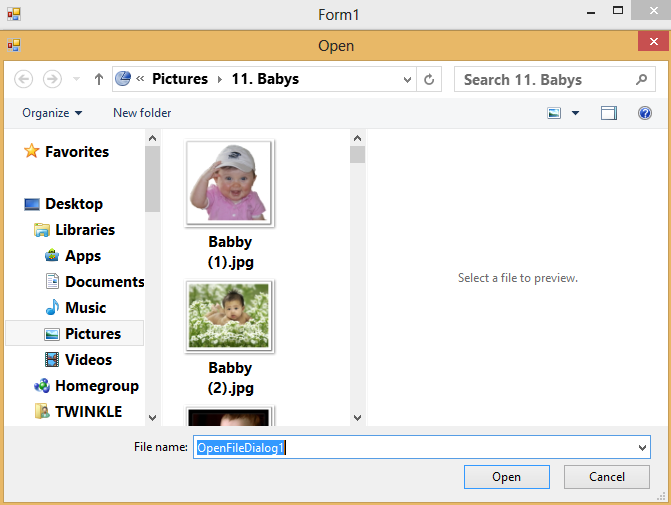
End Class

OUTPUT :

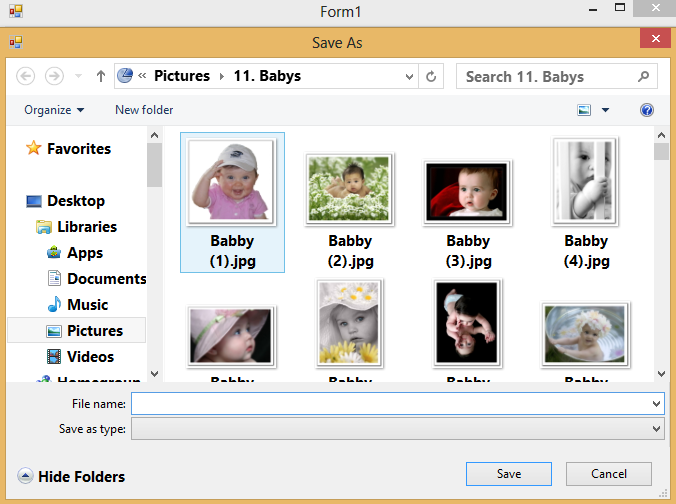
* INTERFACE



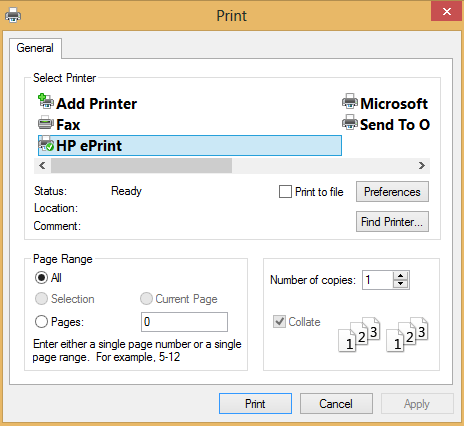
* Controls added
* File -> open



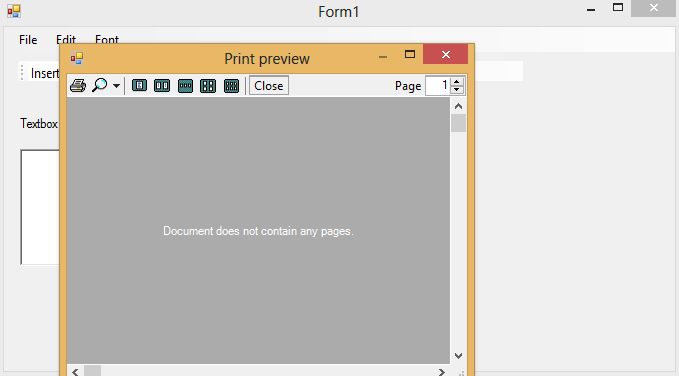
* File -> save and save as



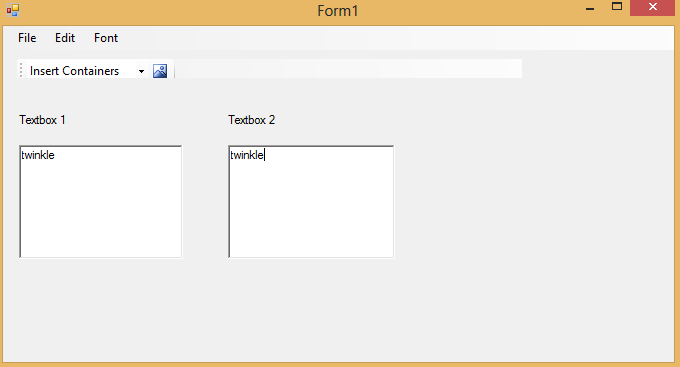
* File -> print



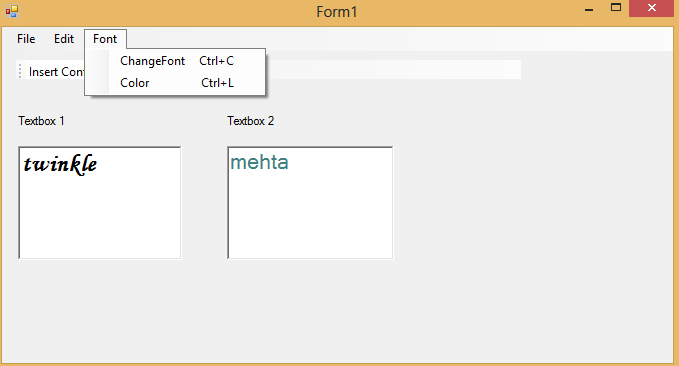
* File -> print preview



* Edit -> cut ,copy ,paste



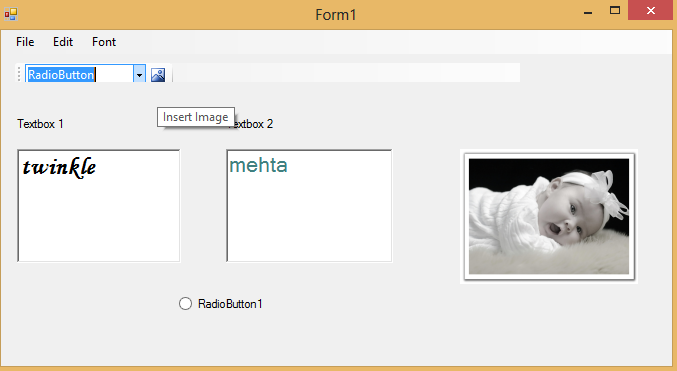
* Font-> changefont and color



* Toolstrip “insert container”



* Toolstrip “insert image”



PRACTICAL : 07

Implement MASTER FORM with Windows Application.

Program:

Public Class Form1

Private Sub NewToolStripMenuItem\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles NewToolStripMenuItem.Click

Dim frm As New Form

frm.MdiParent = Me

frm.Show()

End Sub

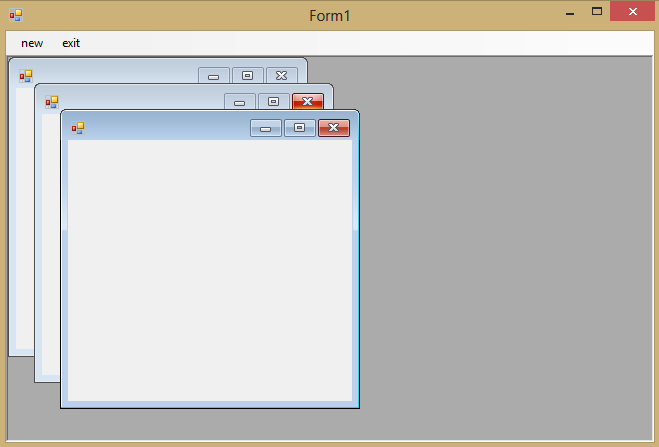
Private Sub ExitToolStripMenuItem\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles ExitToolStripMenuItem.Click

End

End Sub

End Class

OUTPUT :



PRACTICAL 08:

Implement Overloading,Overriding,Constructor and Destructor.

Program :

Overloading:

Imports System

Module Module1

Public Class MySampleClass

Public Sub New()

End Sub

Public Sub New(ByVal Name As String) 'Constructor with One Parameter

Console.WriteLine("Name is:" & Name)

Console.WriteLine("--------------")

End Sub

Public Sub New(ByVal Name As String, ByVal Age As Integer)

'Constructor with two Parameters()

Console.WriteLine("Name is:" & Name)

Console.WriteLine("Age is:" & Age)

End Sub

End Class

Sub Main()

Dim Obj1 As MySampleClass = New MySampleClass() 'No Parameter

Dim Obj2 As MySampleClass = New MySampleClass("Twinkle") 'One Parameter

Dim Obj3 As MySampleClass = New MySampleClass("Twinkle", 20) 'Two Parameters

Console.ReadLine()

End Sub

End Module

Output :

Program:

Overriding :

Module Module1

Class Over

Public Overridable Function add(ByVal x As Integer, ByVal y As Integer)

Console.WriteLine("Function Inside Base Class")

Return (x + y)

End Function

End Class

Class DerOver

Inherits Over

Public Overrides Function add(ByVal x As Integer, ByVal y As Integer)

Console.WriteLine(MyBase.add(120, 100))

Console.WriteLine("Function Inside Derived Class")

Return (x + y)

End Function

End Class

Sub Main()

Dim obj As New DerOver

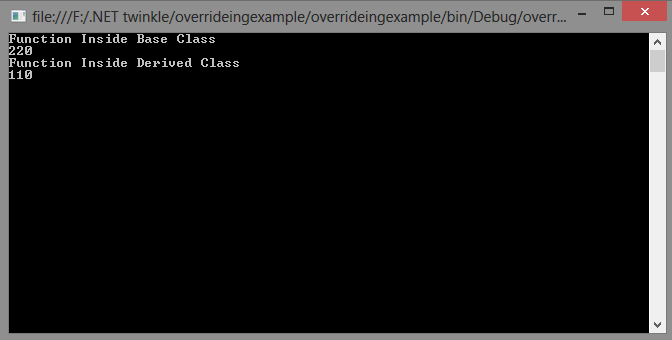
Console.WriteLine(obj.add(10, 100))

Console.Read()

End Sub

End Module

Output :



PRACTICAL 09

Write a program for events and delegates.

Program :

using System;

using System.Collections.Generic;

using System.Text;

public delegate void EventHandler();

class Program

{

public static event EventHandler ev;

static void Main()

{

ev += new EventHandler(Dog);

ev += new EventHandler(Cat);

ev += new EventHandler(Mouse);

ev += new EventHandler(Mouse);

ev.Invoke();

Console.Read();

}

static void Cat()

{

Console.WriteLine("This is Cat");

}

static void Dog()

{

Console.WriteLine("This is Dog");

}

static void Mouse()

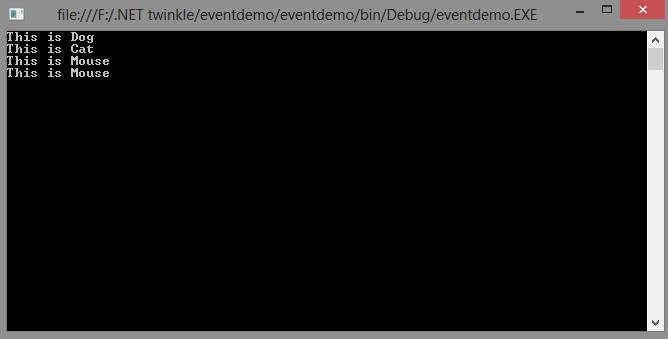
{

Console.WriteLine("This is Mouse");

}

}

Output :



PRACTICAL 10

Implement concepts of Inheritance, visual inheritance and Interface in

Windows application**.**

Program :

Inheritance:

Imports System

Module Module1

Public Class baseclass

Public m As Integer

Public Sub baseclassmethod()

Console.Write("base class method")

End Sub

End Class

Public Class derivedclass

Inherits baseclass

Public Sub derivedclassmethod()

Console.WriteLine(" derived class method")

End Sub

End Class

Sub Main()

Console.WriteLine("accessing base class object:")

Dim bc As baseclass = New baseclass()

bc.m = 10

bc.baseclassmethod()

Console.WriteLine("")

Console.WriteLine("creating object of derived class :")

Dim dc As derivedclass = New derivedclass()

dc.m = 20

dc.baseclassmethod()

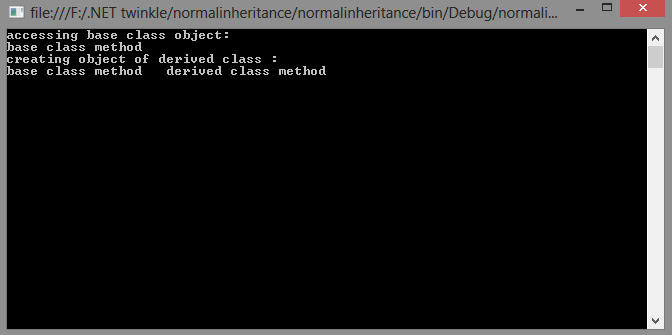
dc.derivedclassmethod()

Console.Read()

End Sub

End Module

Output:



Visual inheritance :

Public Class BaseForm

Public Overridable Sub Add()

EditMode(True)

End Sub

Public Overridable Sub EditMode(ByVal eedit As Boolean)

If eedit = True Then

btnSave.Visible = True

btnCancel.Visible = True

Else

btnSave.Visible = False

btnCancel.Visible = False

End If

End Sub

Public Overridable Sub save()

MsgBox("base form save button clicked")

End Sub

Public Overridable Sub btnSave\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnSave.Click

Me.save()

End Sub

Public Overridable Sub cancel()

MsgBox("base form cancel button clicked")

Me.Close()

End Sub

Public Overridable Sub btnCancel\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnCancel.Click

Me.cancel()

End Sub

Private Sub btnAdd\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnAdd.Click

Me.Add()

InheritedForm.Show()

End Sub

Private Sub BaseForm\_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load

End Sub

End Class

Inherited form:

Public Class InheritedForm

Public Overrides Sub save()

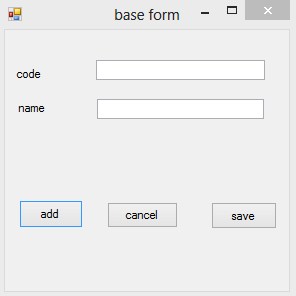
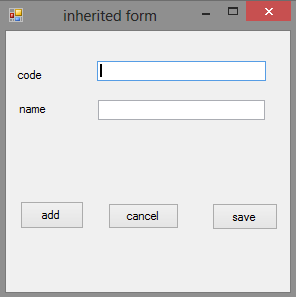
MsgBox("inherited form save button clicked")

MyBase.EditMode(False)

End Sub

End Class

Output :

Interface :

Module Module1

Public Interface Interface1

Function Add(ByVal x As Integer) As Integer

End Interface

Public Class first

Implements Interface1

Public Function Add(ByVal x As Integer) As Integer Implements Interface1.Add

Console.WriteLine("Implementing x+x in first class::" & (x + x))

End Function

End Class

Public Class second

Implements Interface1

Public Function Add(ByVal x As Integer) As Integer Implements Interface1.Add

Console.WriteLine("Implementing x+x+x in second class::" & (x + x + x))

End Function

End Class

Sub Main()

Dim obj1 As New first

Dim obj2 As New second

obj1.Add(10)

obj2.Add(50)

Console.Read()

End Sub

End Module

Output:



PRACTICAL 11

Implement printing of GDI+ with windows application

Program :

Public Class Form1

Private Sub Button1\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click

Dim g As Graphics = Me.CreateGraphics()

Dim pen As Pen = New Pen(Color.BlueViolet, 2)

g.DrawEllipse(pen, 250, 125, 150, 150)

g.DrawEllipse(pen, 260, 135, 110, 110)

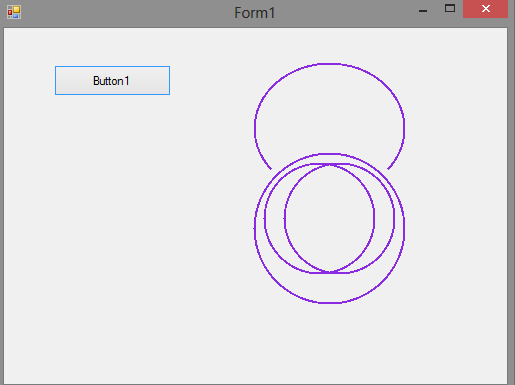
g.DrawEllipse(pen, 280, 135, 110, 110)

g.DrawArc(pen, 250, 35, 150, 130, 145, 250)

End Sub

End Class

Output :

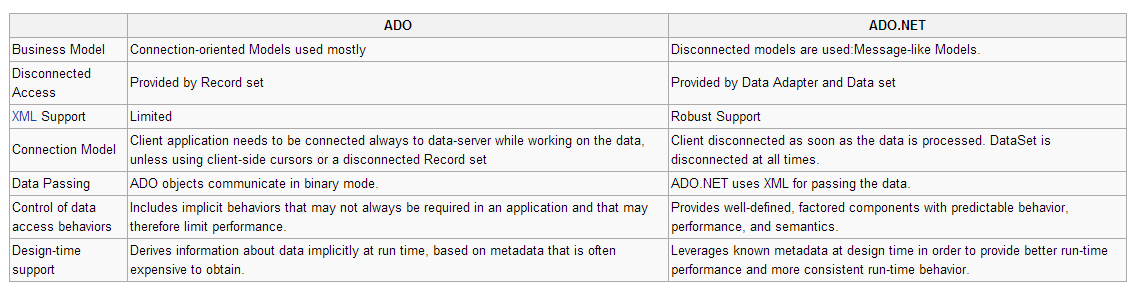


PRACTICAL 12

Study of comparison between ADO.net and basic ADO

* 1. ADO used connected data usage, while ADO.net used disconnected data environment.
  2. ADO used [OLE DB](http://dotnetguts.blogspot.in/2007/07/difference-between-ado-and-adonet.html) to access data and is COM-based, while ADO.net uses XML as the format for transmitting data to and from your database and web application.

* 1. In ADO, Record set, is like a single table or query result, while in ADO.net Dataset, can contain multiple tables from any data source.
  2. In ADO, it is sometime problematic because firewall prohibits many types of request, while in ADO.net there is no such problem because XML is completely firewall-proof.
  3. In ADO you scan sequentially through the rows of the recordset using the ADO **MoveNext** method. In ADO.NET, rows are represented as collections, so you can loop through a table as you would through any collection, or access particular rows via ordinal or primary key index.
  4. In ADO.NET you open connections only long enough to perform a database operation, such as a Select or Update. You can read rows into a dataset and then work with them without staying connected to the data source. In ADO the recordset can provide disconnected access, but ADO is designed primarily for connected access.
  5. Transmitting an ADO.NET dataset between applications is much easier than transmitting an ADO disconnected recordset. To transmit an ADO disconnected recordset from one component to another, you use COM marshalling. To transmit data in ADO.NET, you use a dataset, which can transmit an XML stream.



PRACTICAL 13

Use Dataset, Data Reader, XML Reader & Data Sources (SQL , Object &XML) with Any Windows or Web Application.

Dataset.aspx

<%@ Page Language="C#" AutoEventWireup="true" CodeFile="DataSet.aspx.cs" Inherits="\_Default" %>

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

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

<head runat="server">

<title></title>

<style type="text/css">

.style1

{

width: 100%;

}

.style3

{

}

</style>

</head>

<body>

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

<div>

<table class="style1">

<tr>

<td class="style3">

Courier ID</td>

<td>

<asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>

&nbsp;</td>

</tr>

<tr>

<td class="style3" colspan="2">

<asp:Button ID="Button1" runat="server" onclick="Button1\_Click"

Text="Find The Couriers Issued by the user" />

</td>

</tr>

</table>

</div>

<asp:GridView ID="GridView1" runat="server" AutoGenerateColumns="false">

<Columns>

<asp:TemplateField HeaderText="Courier Names">

<ItemTemplate>

<%# Eval("Courier Company") %>

</ItemTemplate>

</asp:TemplateField>

</Columns>

</asp:GridView>

</form></body></html>

Dataset.aspx.cs

using System;

using System.Collections.Generic;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

using System.Data.SqlClient;

using System.Data;

public partial class \_Default : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e)

{

}

protected void Button1\_Click(object sender, EventArgs e)

{

SqlConnection con=new SqlConnection("Data Source=.\\SQLEXPRESS;AttachDbFilename=|DataDirectory|\\Database.mdf;Integrated Security=True;User Instance=True");

con.Open();

SqlDataAdapter da=new SqlDataAdapter("select Courier\_cmpny from data where Courier\_ID='"+TextBox1.Text+"'",con);

DataSet dt = new DataSet();

da.Fill(dt);

GridView1.DataSource = dt;

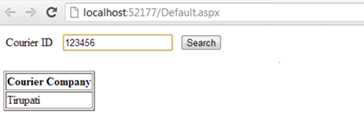
GridView1.DataBind();

con.Close();

}

}

OUTPUT:



Data Reader.aspx

<%@ Page Language="C#" AutoEventWireup="true" CodeFile="DataReader.aspx.cs" Inherits="\_Default" %>

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

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

<body>

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

<table

<tr>

<td >

User ID</td>

<td>

<asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>

&nbsp;</td>

</tr>

<tr>

<td class="style3" colspan="2">

<asp:Button ID="Button1" runat="server" onclick="Button1\_Click"

Text="Find The Couriers Issued by the user" />

</td>

</tr>

</table>

<asp:SqlDataSource ID="SqlDataSource1" runat="server"

ConnectionString="<%$ ConnectionStrings:ConnectionString %>"

SelectCommand="SELECT \* FROM [data]"></asp:SqlDataSource>

<asp:Literal ID="Literal1" runat="server"></asp:Literal>

<asp:GridView ID="GridView1" runat="server" AutoGenerateColumns="false">

<Columns>

<asp:BoundField datafield="Value" headertext="Courier Names" />

</Columns>

</asp:GridView>

</form>

</body>

<html>

DataReader.aspx.cs

using System;

using System.Collections.Generic;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

using System.Data.SqlClient;

using System.Data;

using System.Collections;

public partial class \_Default : System.Web.UI.Page

{

protected void Button1\_Click(object sender, EventArgs e)

{

Int32 i = 0;

SqlConnection con=new SqlConnection("Data Source=.\\SQLEXPRESS;AttachDbFilename=|DataDirectory|\\Database.mdf;Integrated Security=True;User Instance=True");

con.Open();

SqlCommand cmd = new SqlCommand("select Courier\_cmpny from data where Courier\_ID='"+TextBox1.Text+"'",con);

SqlDataReader dr;

Hashtable hs = new Hashtable();

dr=cmd.ExecuteReader();

while (dr.Read())

{

hs.Add(i++,dr[0].ToString());

}

dr.Close();

GridView1.DataSource=hs;

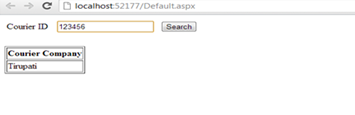
GridView1.DataBind();

con.Close();

}

}

OUTPUT:



XML Reader.aspx.cs:

using System;

using System.Collections.Generic;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

using System.Xml;

public partial class XML : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e)

{

XmlReader reader = XmlReader.Create(Server.MapPath("XMLFile.xml"));

while (reader.Read())

{

if (reader.IsStartElement())

{

// Get element name and switch on it.

switch (reader.Name)

{

case "Library":

// Detect this element.

Response.Write("Start element: Library");

break;

case "Courier":

// Detect this Courier element.

Response.Write("<br/> Courier");

// Search for the attribute name on this current node.

string attribute = reader["name"];

if (attribute != null)

{

Response.Write(":" + attribute);

}

// Next read will contain text.

if (reader.Read())

{

Response.Write(":" + reader.Value.Trim());

}

break;

}

} }

} }

XML File:

<?xml version="1.0" encoding="utf-8" ?>

<Library>

<Courier name="ID">

123456

</Courier>

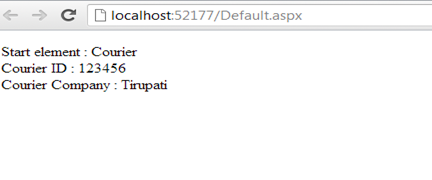
<Courier name="Company">

Tirupati

</Courier>

</Library>

OUTPUT:



PRACTICAL 14

Use Data Controls Like Data list, Grid View,Detail View, Repeater and List Bound Control

DataControls.aspx

<%@ Page Language="C#" AutoEventWireup="true" CodeFile="DataControls.aspx.cs" Inherits="\_Default" %>

<html

<head runat="server">

<title></title>

</head>

<body>

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

<div>

<table class="style1">

<tr>

<td> User ID</td>

<td><asp:TextBox ID="TextBox3" runat="server"></asp:TextBox>

</td>

</tr>

<tr>

<td> Courier ID</td>

<td> <asp:TextBox ID="TextBox4" runat="server"></asp:TextBox>

</td>

</tr>

<tr>

<td colspan="2">

<asp:Button ID="Button1" runat="server" Text="Submit" onclick="Button1\_Click" />

</td>

</tr>

<tr>

<td colspan="2">

<asp:Label ID="Label1" runat="server" Text="Courier issued Successfully"

Visible="False"></asp:Label>

</td>

</tr>

</table>

</div>

<p>

&nbsp;Grid View<asp:GridView ID="GridView1" runat="server" AutoGenerateColumns="false"

DataSourceID="SqlDataSource1">

<Columns>

<asp:BoundField DataField="Courier\_ID" HeaderText="Courier\_ID"

SortExpression="Courier\_ID" />

<asp:BoundField DataField="User\_ID" HeaderText="User\_ID"

SortExpression="User\_ID" />

</Columns>

</asp:GridView>

</p>

<p>

Repeater</p>

<p>

<asp:Repeater ID="Repeater1" runat="server" DataSourceID="SqlDataSource1"

onitemcommand="Repeater1\_ItemCommand">

<ItemTemplate>

Courier Id: <%# Eval("Courier\_ID") %>

User ID: <%# Eval("User\_ID") %>

</ItemTemplate>

</asp:Repeater>

</p>

Data List

<asp:DataList ID="DataList1" runat="server" DataSourceID="SqlDataSource1">

<ItemTemplate>

Courier\_ID:

<asp:Label ID="Courier\_IDLabel" runat="server" Text='<%# Eval("Courier\_ID") %>' />

<br />

User\_ID:

<asp:Label ID="User\_IDLabel" runat="server" Text='<%# Eval("User\_ID") %>' />

<br />

<br />

</ItemTemplate>

</asp:DataList>

List Bound control<asp:ListView ID="ListView1" runat="server" DataSourceID="SqlDataSource1">

<AlternatingItemTemplate>

<span style="background-color: #FFF8DC;">Courier\_ID:

<asp:Label ID="Courier\_IDLabel" runat="server" Text='<%# Eval("Courier\_ID") %>' />

<br />

User\_ID:

<asp:Label ID="User\_IDLabel" runat="server" Text='<%# Eval("User\_ID") %>' />

<br />

<br /></span>

</AlternatingItemTemplate>

<EditItemTemplate>

<span style="background-color: #008A8C;color: #FFFFFF;">Courier\_ID:

<asp:TextBox ID="Courier\_IDTextBox" runat="server" Text='<%# Bind("Courier\_ID") %>' />

<br />

User\_ID:

<asp:TextBox ID="User\_IDTextBox" runat="server" Text='<%# Bind("User\_ID") %>' />

<br />

<asp:Button ID="UpdateButton" runat="server" CommandName="Update"

Text="Update" />

<asp:Button ID="CancelButton" runat="server" CommandName="Cancel"

Text="Cancel" />

<br /><br /></span>

</EditItemTemplate>

<EmptyDataTemplate>

<span>No data was returned.</span>

</EmptyDataTemplate>

<InsertItemTemplate>

<span style="">Courier\_ID:

<asp:TextBox ID="Courier\_IDTextBox" runat="server" Text='<%# Bind("Courier\_ID") %>' />

<br />User\_ID:

<asp:TextBox ID="User\_IDTextBox" runat="server" Text='<%# Bind("User\_ID") %>' />

<br />

<asp:Button ID="InsertButton" runat="server" CommandName="Insert"

Text="Insert" />

<asp:Button ID="CancelButton" runat="server" CommandName="Cancel"

Text="Clear" />

<br /><br /></span>

</InsertItemTemplate>

<ItemTemplate>

<span style="background-color: #DCDCDC;color: #000000;">Courier\_ID:

<asp:Label ID="Courier\_IDLabel" runat="server" Text='<%# Eval("Courier\_ID") %>' />

<br />

User\_ID:

<asp:Label ID="User\_IDLabel" runat="server" Text='<%# Eval("User\_ID") %>' />

<br />

<br /></span>

</ItemTemplate>

<LayoutTemplate>

<div ID="itemPlaceholderContainer" runat="server"

style="font-family: Verdana, Arial, Helvetica, sans-serif;">

<span runat="server" id="itemPlaceholder" />

</div>

<div style="text-align: center;background-color: #CCCCCC;font-family: Verdana, Arial, Helvetica, sans-serif;color: #000000;">

</div>

</LayoutTemplate>

<SelectedItemTemplate>

<span style="background-color: #008A8C;font-weight: bold;color: #FFFFFF;">

Courier\_ID:

<asp:Label ID="Courier\_IDLabel" runat="server" Text='<%# Eval("Courier\_ID") %>' />

<br />

User\_ID:

<asp:Label ID="User\_IDLabel" runat="server" Text='<%# Eval("User\_ID") %>' />

<br />

<br /></span>

</SelectedItemTemplate>

</asp:ListView>

<p>

Detail View<asp:DetailsView ID="DetailsView1" runat="server" AutoGenerateRows="False"

DataSourceID="SqlDataSource1" Height="50px" Width="125px">

<Fields>

<asp:BoundField DataField="Courier\_ID" HeaderText="Courier\_ID"

SortExpression="Courier\_ID" />

<asp:BoundField DataField="User\_ID" HeaderText="User\_ID"

SortExpression="User\_ID" />

</Fields>

</asp:DetailsView>

</p>

</form>

</body>

</html>

DataControls.aspx.cs

using System;

using System.Collections.Generic;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

using System.Data.SqlClient;

public partial class \_Default : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e)

{

if (!Page.IsPostBack)

{

Label1.Visible = false;

}

}

protected void Button1\_Click(object sender, EventArgs e)

{

SqlConnection con = new SqlConnection("Data Source=.\\SQLEXPRESS;AttachDbFilename=|DataDirectory|\\Database.mdf;Integrated Security=True;User Instance=True");

con.Open();

SqlCommand cmd = new SqlCommand("insert into Courier\_Issue values ('"+TextBox4.Text+"','"+TextBox3.Text+"')",con);

cmd.ExecuteNonQuery();

GridView1.DataBind();

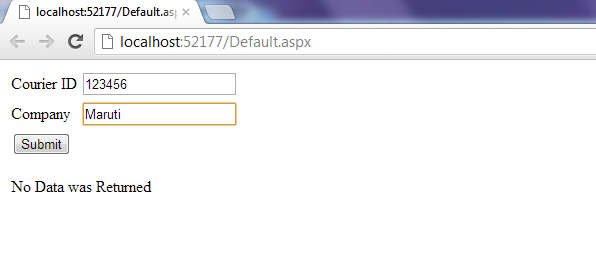
Label1.Visible = true;

con.Close();

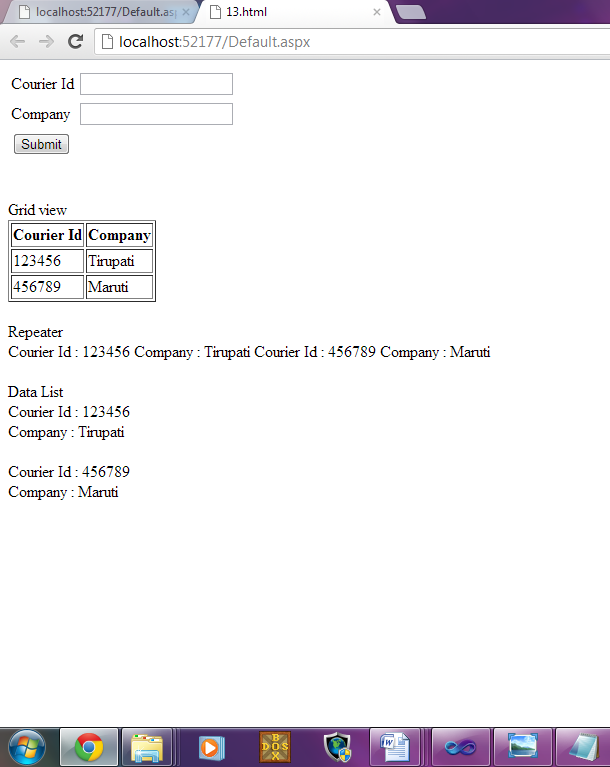
}

}

OUTPUT:



After Adding some Information



PRACTICAL 15

Implement web application using ASP.NET with web control

<%@ Page Language="C#" AutoEventWireup="true" CodeFile="Default4.aspx.cs" Inherits="Default4" %>

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

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

<head id="Head1" runat="server">

<title></title>

</head>

<body>

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

<div align="center">

<table bgcolor="#FFCCFF" border="5" cellpadding="5" cellspacing="5">

<tr>

<td>

Name</td>

<td>

<asp:TextBox ID="txt\_name" runat="server"></asp:TextBox>

</td>

</tr>

<tr>

<td>

Email</td>

<td>

<asp:TextBox ID="txt\_email" runat="server"></asp:TextBox>

</td>

</tr>

<tr>

<td>

Contactno</td>

<td>

<asp:TextBox ID="txt\_contact" runat="server"></asp:TextBox>

</td>

</tr>

<tr>

<td>

City</td>

<td>

<asp:TextBox ID="txt\_city" runat="server"></asp:TextBox>

</td>

</tr>

<tr>

<td>

<asp:Label ID="Label4" runat="server" Text="Date of Birth"></asp:Label>

</td>

<td>

<asp:DropDownList ID="DropDownList1" runat="server">

<asp:ListItem Selected="True">DD</asp:ListItem>

</asp:DropDownList>

<asp:DropDownList ID="DropDownList2" runat="server">

<asp:ListItem Selected="True">MM</asp:ListItem>

</asp:DropDownList>

<asp:DropDownList ID="DropDownList3" runat="server">

<asp:ListItem Selected="True">YYYY</asp:ListItem>

</asp:DropDownList>

</td>

</tr>

<tr>

<td>

<asp:Label ID="Label5" runat="server" Text="Gender"></asp:Label>

</td>

<td>

<asp:RadioButton ID="RadioButton1" runat="server" GroupName="test"

Text="Male" />

<asp:RadioButton ID="RadioButton2"

runat="server" BorderColor="#FFCCFF" GroupName="test" Text="Female" />

</td>

</tr>

<tr>

<td align="center" colspan="2">

<asp:Button ID="btn\_register" Text="Register" runat="server"

onclick="btn\_register\_Click" />

</td>

</tr>

</table>

</div>

<br />

<br />

<br />

<br />

<br />

<table

align="center">

<tr>

<td align=center colspan =2>Registration Details</td>

</tr>

<tr>

<td align=center colspan =2></td>

</tr>

<tr>

<td>

Name

</td>

<td>

<asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>

</td>

</tr>

<tr>

<td>

Email</td>

<td>

<asp:TextBox ID="TextBox2" runat="server"></asp:TextBox>

</td>

</tr>

<tr>

<td>

Contactno</td>

<td>

<asp:TextBox ID="TextBox3" runat="server"></asp:TextBox>

</td>

</tr>

<tr>

<td>

City</td>

<td>

<asp:TextBox ID="TextBox4" runat="server"></asp:TextBox>

</td>

</tr>

<tr>

<td>

<asp:Label ID="Label1" runat="server" Text="Date of Birth"></asp:Label>

</td>

<td>

<asp:TextBox ID="TextBox5" runat="server"></asp:TextBox>

</td>

</tr>

<tr>

<td>

<asp:Label ID="Label2" runat="server" Text="Gender"></asp:Label>

</td>

<td>

<asp:TextBox ID="TextBox6" runat="server"></asp:TextBox>

</td>

</tr>

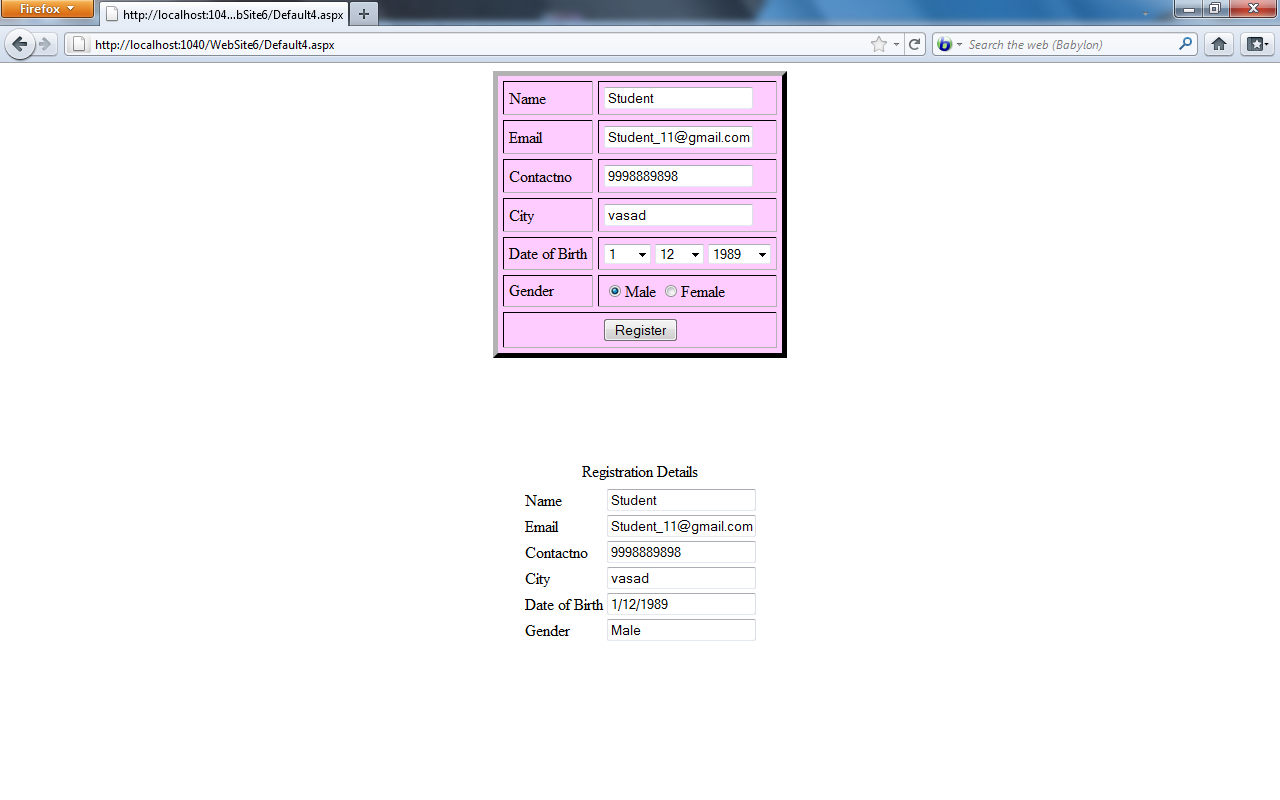
</table>

</form>

</body>

</html>

**OUTPUT:**

****

PRACTICAL 16

Write a code for web application to provide input validations using input valuators

<%@ Page Language="C#" AutoEventWireup="true" CodeFile="Default4.aspx.cs" Inherits="Default4"%>

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

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

<head runat="server">

    <title></title>

</head>

<body>

    <form id="form1" runat="server">

    <div>

    <table bgcolor="#6699FF" border="5" title="Validation Controls">

    <tr>

    <td>

    USER NAME:

    </td>

    <td>

    <asp:TextBox Id="txtusername" runat="server" TabIndex="1" ToolTip="Enter Your Username" ></asp:TextBox>

    <asp:RequiredFieldValidator id="validatetxtusername" runat="server"

            controlToValidate="txtusername" errorMessage="Please Fill Up Username"

            display="static" ForeColor="#CC0000"></asp:RequiredFieldValidator>

</td>

       </tr>

    <tr>

    <td>

    PASSWORD:

    </td>

    <td>

    <asp:TextBox ID="txtpassword" runat="server" TabIndex="2" ToolTip="Enter Your Password" TextMode="Password" ></asp:TextBox>

    <asp:CompareValidator ID="validatetxtpassword" runat="server"

            ControlToValidate="txtpassword" ControlToCompare="txtconfirmpassword"

            ErrorMessage="Password Not Mach" TextMode="Password" ForeColor="#CC0000"></asp:CompareValidator>

    <br />

    </td>

    </tr>

    <tr>

    <td>

    CONFIRM PASSWORD:

    </td>

    <td>

    <asp:TextBox ID="txtconfirmpassword" runat="server" TabIndex="3" ToolTip="Enter Your  Confirm Password" TextMode="Password"></asp:TextBox>

   </td>

    </tr>

    <tr>

    <td>

    EMAIL:

    </td>

    <td>

    <asp:TextBox ID="txtemail" runat="server" TabIndex="4" ToolTip="Enter Email"></asp:TextBox>

    <asp:RegularExpressionValidator ID="validatetxtemail"

            ValidationExpression="\w+([-+.']\w+)\*@\w+([-.]\w+)\*\.\w+([-.]\w+)\*"

            runat="server"  ControlToValidate="txtemail"

            ErrorMessage="Enter Valid Email Address" ForeColor="#CC0000"></asp:RegularExpressionValidator>

            <br />

            </td>

            </tr>

            <tr>

            <td>

            </td>

            <td>

            <asp:Button ID="btn\_submit" Text="Submit" runat="server"

                    onclick="btn\_submit\_Click" />

                </td>

            </tr>

            </table>

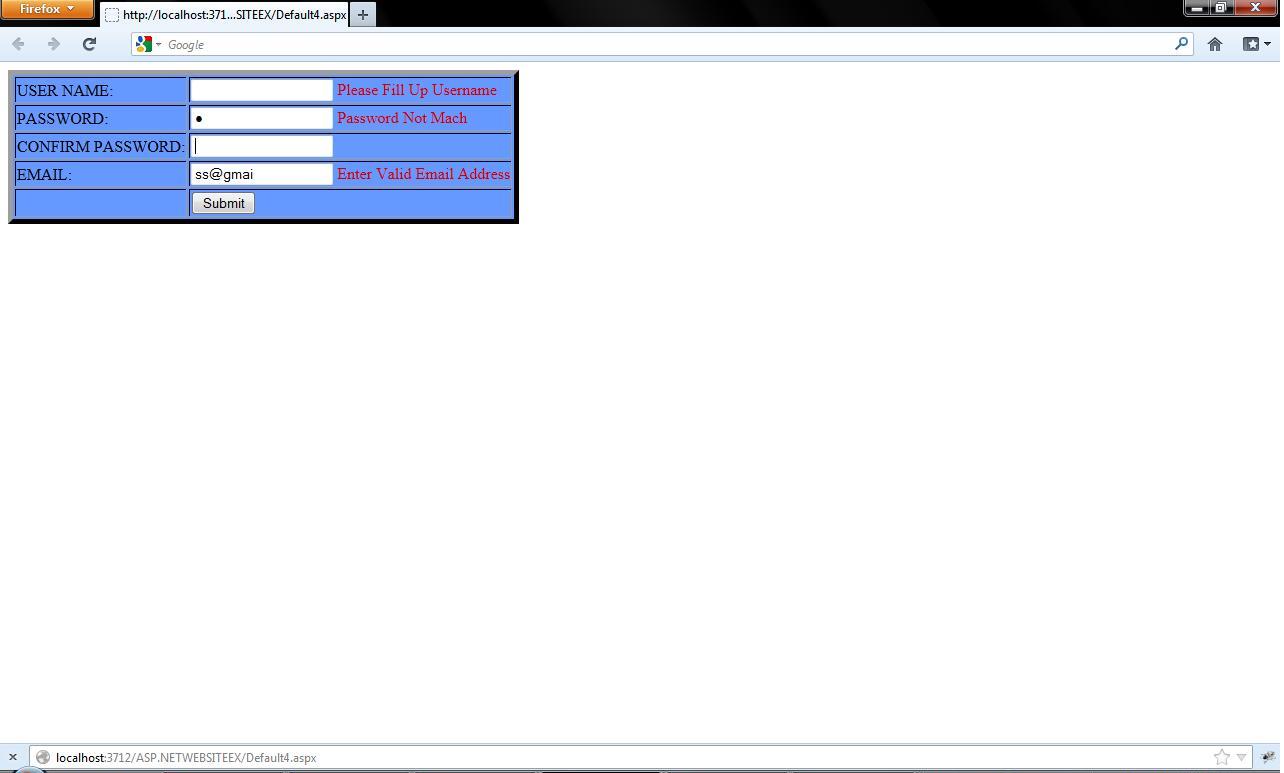
    </div>

    </form>

</body>

</html>

**OUTPUT:**

****

PRACTICAL 17

Create a Web Application that illustrates the use of themes and master pages with site map.

MasterPage.Master

<%@ Master Language="C#" AutoEventWireup="true" CodeFile="MasterPage.master.cs" Inherits="MasterPage" %>

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

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

<head runat="server">

<title></title>

<asp:ContentPlaceHolder id="head" runat="server">

</asp:ContentPlaceHolder>

</head>

<body bgcolor="aqua" >

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

<div>

<asp:Label ID="Label1" runat="server" Text="Welcome To Courier Management System!"></asp:Label>

<asp:ContentPlaceHolder id="ContentPlaceHolder1" runat="server">

</asp:ContentPlaceHolder>

<address>

<a href="Default.aspx">Click here to Search</a>

<br />

<a href="XMLFile.xml">Click here to View XML File</a></address>

</div>

</form>

</body>

</html>

SkinFile.skin

<asp:Label runat="server" ForeColor="#585880"

Font-Size="2em" Font-Names="Verdana"

/>

MasterFile.aspx

<%@ Page Title="" Language="C#" MasterPageFile="~/MasterPage.master" AutoEventWireup="true" CodeFile="MasterUse.aspx.cs" Inherits="MasterUse" %>

<asp:Content ID="Content1" ContentPlaceHolderID="head" Runat="Server">

</asp:Content>

<asp:Content ID="Content2" ContentPlaceHolderID="ContentPlaceHolder1" Runat="Server">

<br />

<b>Site Map:

<asp:Menu ID="Menu1" runat="server" DataSourceID="SiteMapDataSource1">

</asp:Menu>

<asp:SiteMapDataSource ID="SiteMapDataSource1" runat="server" />

<b>End of Site Map</b>

</asp:Content>

SiteMap.aspx

<%@ Page Language="C#" AutoEventWireup="true" CodeFile="sitemap.aspx.cs" Inherits="sitemap" %>

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

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

<head runat="server">

<title></title>

</head>

<body>

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

<div>

<asp:Menu ID="Menu1" runat="server" DataSourceID="SiteMapDataSource1">

</asp:Menu>

<asp:SiteMapDataSource ID="SiteMapDataSource1" runat="server" />

</div>

</form>

</body>

</html>

Web.SiteMap

<?xml version="1.0" encoding="utf-8" ?>

<siteMap xmlns="http://schemas.microsoft.com/AspNet/SiteMap-File-1.0" >

<siteMapNode url="Default.aspx" title="Default page" description="Main page">

<siteMapNode url="XML.aspx" title="XML page" description="Display XML Data" />

<siteMapNode url="XMLFile.xml" title="XML File" description="Display XML File" />

</siteMapNode>

</siteMap>

Output:



PRACTICAL 18

Create a web application in ASP.NET using various CSS

Default.aspx

<%@ Page Language="C#" AutoEventWireup="true" CodeFile="Default.aspx.cs" Inherits="Default" %>

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

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

<head id="Head1" runat="server">

<title></title>

<link href="StyleSheet.css" rel="stylesheet" type="text/css" />

</head>

<body>

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

<div>

</div>

<br />

<br />

<asp:Label ID="Label1" runat="server"

Text="This is a simple Web page."></asp:Label>

<br />

<br />

<asp:TextBox ID="TextBox1" runat="server" Width="269px">this is a text box</asp:TextBox>

<br />

</form>

</body>

</html>

Stylesheet.css

body

{

color:#ff0000;

background-color:#0000ff;

font-family: sans-serif;

}

#Label1

{

font-size:18pt

}

p

{

font-size:12pt }

#form1

{

background-color:#efefef;

border-style:solid;

border-width:2px;

border-color:#999900;

height:200px;

width:600px;

}

#TextBox1

{

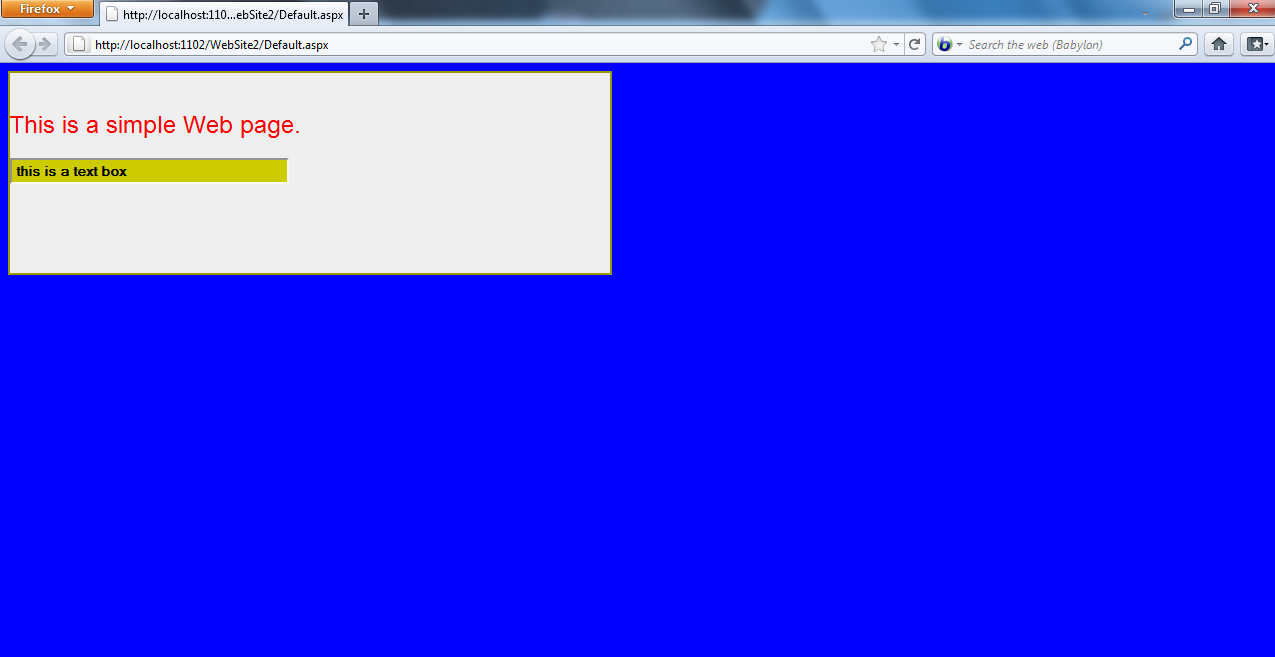
background-color:#cccc00;

font-weight:bold;

padding:3px;

}

OUTPUT:

****

PRACTICAL 19

Implement the concept of state management in a web application.

Default.aspx.cs

using System;

using System.Collections.Generic;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

using System.Data.SqlClient;

using System.Data;

using System.Configuration;

public partial class \_Default : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e)

{

}

protected void Button1\_Click(object sender, EventArgs e)

{

Session["courierid"] = TextBox1.Text;

Response.Redirect("session.aspx");

}

protected void Button2\_Click(object sender, EventArgs e)

{

string url1="qrystring.aspx?id="+TextBox1.Text;

Response.Redirect(url1);

}

}

State.aspx

<%@ Page Language="C#" AutoEventWireup="true" CodeFile="qrystring.aspx.cs" Inherits="qrystring" %>

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

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

<head id="Head1" runat="server">

<title></title>

</head>

<body>

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

<div>

<asp:GridView ID="GridView1" runat="server" AutoGenerateColumns="false">

<Columns>

<asp:TemplateField HeaderText="Courier Names">

<ItemTemplate>

<%# Eval("couriername") %>

</ItemTemplate>

</asp:TemplateField>

</Columns>

</asp:GridView>

</div>

</form>

</body>

</html>

Session.aspx.cs

using System;

using System.Collections.Generic;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

using System.Data.SqlClient;

using System.Data;

using System.Configuration;

public partial class session : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e)

{

SqlConnection con = new SqlConnection("Data Source=.\\SQLEXPRESS;AttachDbFilename=|DataDirectory|\\Database.mdf;Integrated Security=True;User Instance=True");

con.Open();

SqlDataAdapter da = new SqlDataAdapter("select CourierName from CourierInfo where UserID='" + Session["courierid"].ToString() + "'", con);

/\* For qrystring.aspx.cs

SqlDataAdapter da = new SqlDataAdapter("select CourierName from CourierInfo where UserID='" + Request.QueryString["id"].ToString() + "'", con); \*/

DataSet dt = new DataSet();

da.Fill(dt);

GridView1.DataSource = dt;

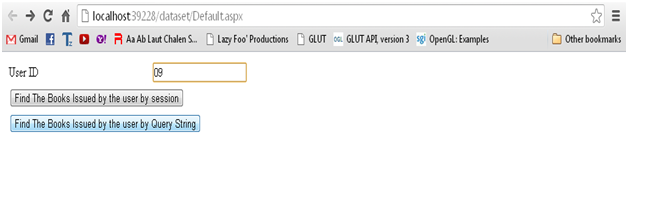
GridView1.DataBind();

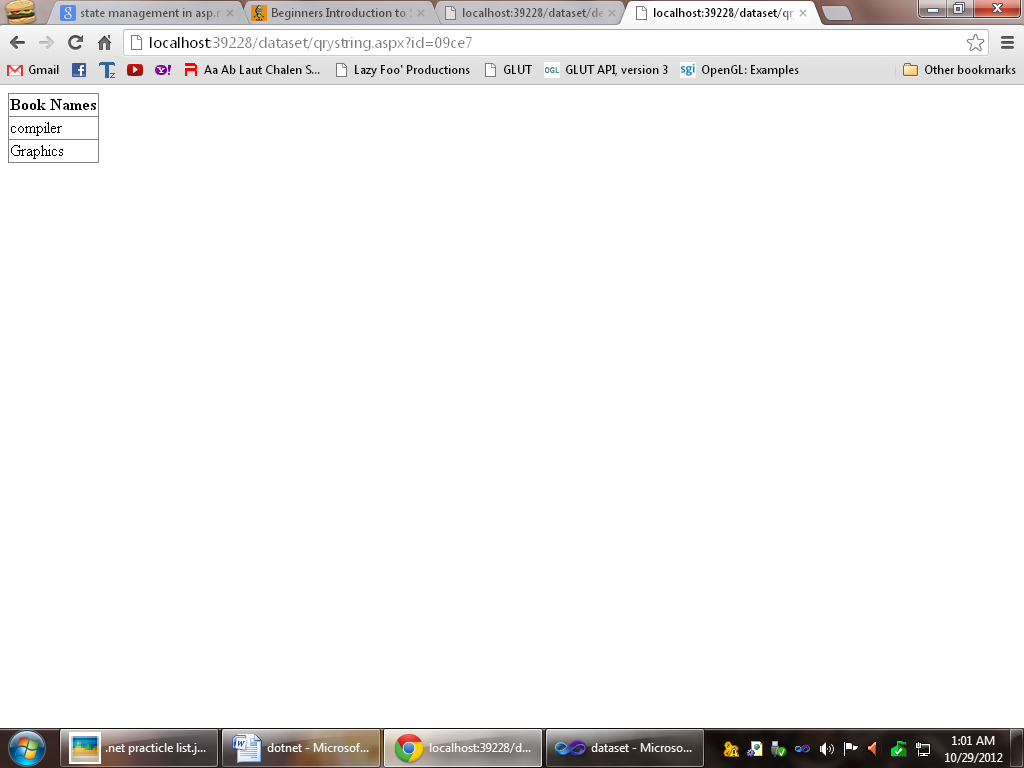
con.Close();

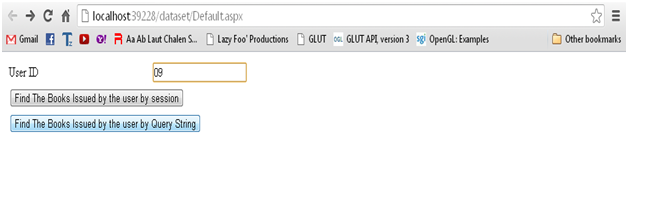
}

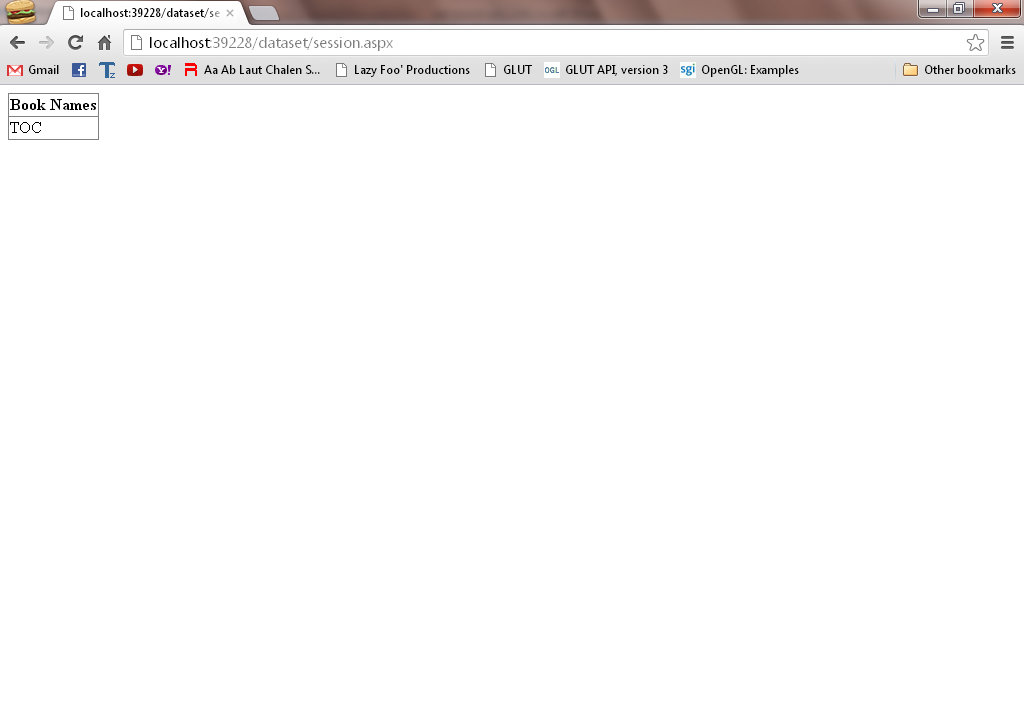
}

Output:









Illustrate The Use of Cookie

Program :

using System;

using System.Collections.Generic;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

public partial class \_Default : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e){ }

protected void Button1\_Click(object sender, EventArgs e)

{

HttpCookie myCookie = new HttpCookie("MyTestCookie");

myCookie.Value = TextBox1.Text .ToString();

// Set the cookie expiration date.

myCookie.Expires = DateTime.Now.AddMinutes(10);

// Add the cookie.

Response.Cookies.Add(myCookie);

Response.Write("<p> The cookie has been written.");

}

protected void Button2\_Click(object sender, EventArgs e)

{

HttpCookie myCookie = new HttpCookie("MyTestCookie");

myCookie = Request.Cookies["MyTestCookie"];

// Read the cookie information and display it.

if (myCookie != null)

{

// Response.Write("<p>" + myCookie.Name + "<p>" + myCookie.Value);

Label1.Text = myCookie.Value;

}

else

Response.Write("not found");

}

}

Output:





PRACTICAL 20

Implement code in Asp.Net that creates and consumes web services.

<%@ Page Title="Home Page" Language="C#" MasterPageFile="~/Site.master" AutoEventWireup="true"

CodeFile="Default.aspx.cs" Inherits="\_Default" %>

<asp:Content ID="HeaderContent" runat="server" ContentPlaceHolderID="HeadContent">

</asp:Content>

<asp:Content ID="BodyContent" runat="server" ContentPlaceHolderID="MainContent">

<asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>

<asp:Button ID="Button" runat="server" onclick="Button1\_Click" Text="Add" />

<asp:TextBox ID="TextBox2" runat="server"></asp:TextBox>

<br />

<asp:Label ID="Label1" runat="server"></asp:Label>

<br />

</asp:Content>

service.cs:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.Services;

/// <summary>

/// Summary description for service

/// </summary>

[WebService(Namespace = "http://tempuri.org/")]

[WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1\_1)]

// To allow this Web Service to be called from script, using ASP.NET AJAX, uncomment the following line.

// [System.Web.Script.Services.ScriptService]

public class service : System.Web.Services.WebService {

public service () {

//Uncomment the following line if using designed components

//InitializeComponent();

}

[WebMethod]

public string HelloWorld(string a) {

return "Hello World"+a;

}

[WebMethod]

public string add(string a, string b)

{

double d = double.Parse(a);

double c = double.Parse(b);

c = c + d;

return c.ToString() ;

}

}

Default.aspx.cs:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

public partial class \_Default : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e)

{

}

protected void Button1\_Click(object sender, EventArgs e)

{

localhost.service web1 = new localhost.service();

Label1.Text = web1.add(TextBox1.Text ,TextBox2.Text );

}

}

PRACTICAL 21

Study of ASP.net Administration and Configuration tool.

The Web Site Administration tool was first introduced with [ASP.NET 2.0](http://en.wikipedia.org/wiki/ASP.NET_2.0) along with ASP.NET Microsoft Management Console (MMC) Snap-in.

[Web Site Configuration](javascript:void(0)):

Web site configuration settings are stored in an XML file named Web.config, which is located in the root folder of the Web site. The Web Site Administration Tool lets you change your site configuration without having to manually edit the Web.config file. The first time that you use the Web Site Administration Tool to administer a specific Web site, if no Web.config file exists, the Web Site Administration Tool creates one. By default, the Web Site Administration Tool also creates a database in the App\_Data folder of the Web site to store application services data, such as membership and roles information. For most settings, changes that are made in the Web Site Administration Tool take effect immediately and are reflected in the Web.config file.

Inherited Settings:

Default settings for a Web site are automatically inherited from any configuration files that exist for the computer or for the Web server as a whole. For example, the Web server might have default settings that apply to all sites on that server. Using the Web Site Administration Tool, you can create and modify settings for your specific Web site that are not inherited, and you can override the inherited settings as allowed by the site-wide settings. If the setting has been inherited and cannot be overridden, it appears dimmed, to indicate that it is disabled, in the Web Site Administration Tool.

[Requirements](javascript:void(0)):

The Web Site Administration Tool is included with the Microsoft Visual Web Developer Web development tool. In order to use the Web Site Administration Tool to administer a Web site, the user credentials for the user account under which you are running Visual Web Developer must have Read and Write permissions to the Web.config file and the App\_Data folder of the application being administered. If you cannot manage the configuration for the Web site using the Web Site Administration Tool, contact the system administrator.

[Features](javascript:void(0)):

The Web Site Administration Tool features a tabbed interface that groups related configuration settings within each tab. The tabs and the configuration settings that the tabs manage are described in the following sections.

Security Tab:

Use the Security tab to manage access rules to help secure specific resources within the Web site and to manage user accounts and roles.

You can specify how the Web site is used—either from the Internet (publicly) or from an intranet (on a local area network). This in turn indicates the type of authentication mode that the Web site will use. Internet Web sites use the ASP.NET membership system, where you define individual user accounts. ASP.NET uses a security system to restrict access to specific user accounts or the roles to which the user accounts belong. Intranet Web sites use Windows authentication, where users are identified by their Windows logon information.

Application Tab:

Use the Application tab to manage a variety of settings related to the Web site, including the following:

* Application settings, which are name/value pairs that you want to store centrally and access in code from anywhere in the Web site.
* SMTP settings, which determine how your site sends e-mail.
* Debug and trace settings.
* Offline and online settings, which take the Web site offline (shut it down) to perform maintenance or to bring a new Microsoft SQL Server Standard edition database online.

Provider Tab:

Use the Provider tab to test or assign providers for membership and role management for the Web site. Database providers are classes that are called to store application data for a particular feature. By default, the Web Site Administration Tool configures and uses a local Microsoft SQL Server Standard Edition database in the App\_Data folder for the Web site. Instead, you can choose to use a different provider, such a remote SQL Server database, to store membership and role management.

[How to Use the Web Site Administration Tool](javascript:void(0))?

Using the Web Site Administration Tool is similar to using other forms-based Web sites. The general procedure is to open the Web Site Administration Tool, select the appropriate tab, and then adjust the settings that are available on that tab. Most changes take effect immediately.

[How to Access the Web Site Administration Tool](javascript:void(0))?

To access the Web Site Administration Tool, on the Website menu, click ASP.Net Configuration.

[**Considerations**](javascript:void(0))**:**

The following sections provide some considerations for working with the Web Site Administration Tool.

Restarting the Application When Saving

Most changes to configuration settings that you make in the Web Site Administration Tool take effect immediately. This requires the Web site to which the change applies to be restarted. Because this will cause currently active sessions in the Web site to be lost, you should make configuration changes to a staged or development version of the Web site before publishing these changes to the production server.

Saving Your Settings:

Most changes to configuration settings that you make in the Web Site Administration Tool take effect immediately. For settings for which the Web Site Administration Tool interface has a dedicated Save button, leaving the Web Site Administration Tool idle or allowing the Web Site Administration Tool to time out before you click Save will cause your configuration settings changes to be lost.

Time Out

As a security measure, the Web Site Administration Tool times out after a period of inactivity. Any settings that did not take effect immediately and were not saved will be lost. If the Web Site Administration Tool has timed out, close your browser, and then reopen the Web Site Administration Tool in a new window.

The Web Site Administration Tool manages only some of the configuration settings that are available to the Web site. Many other settings require direct modification of configuration files either manually, by using the [MMC Snap-In for ASP.NET](http://msdn.microsoft.com/en-us/library/ms186189(v=vs.100).aspx), or programmatically, by using the [ASP.NET Configuration API](http://msdn.microsoft.com/en-us/library/ms178688(v=vs.100).aspx).

Web Site Administration Tool Security Tab

Use the Security tab of the Web Site Administration Tool to manage rules for securing specific resources in the Web application. ASP.NET uses a security system that lets you restrict access to specific user accounts or the roles to which the user accounts belong. With the Security tab, you manage user accounts, roles, and access rules for the Web site. Before using the Security tab for the first time, use the Security Setup Wizard to configure basic security settings for the Web site.

ASP.NET security is based on the concepts of user accounts, roles, and access rules and lets you restrict access to your Web application resources to only the user accounts that you specify. Security settings are established using a combination of configuration settings and data stored in a database (or other data store). User accounts and roles that you create are stored in the database and access rules are stored in the Web.config file.

You can configure your application to use the following types of security, which depend on how the Web site will be used:

* Forms-based authentication (From the Internet)

Forms-based authentication is used for Web sites that are made available to the Internet. Forms-based authentication uses the ASP.NET membership system to manage individual user accounts and groups (roles). User account information is stored in a local database or in a Microsoft SQL Server database. You can use the ASP.NET logon controls to create a logon page where users can enter their credentials.

* Integrated Microsoft Windows authentication (From a local network)

Windows authentication interacts with Windows security, using the logon credentials that users provide when they log on to Windows. Therefore, Windows authentication is suited for intranet scenarios, where users have logged on to a Windows-based network. You do not have to create a logon page, because users are automatically logged on to your application with their Windows credentials.

Use the Users section of the Security tab to complete the following tasks:

* Create, edit, and delete registered user accounts for the Web site.
* View a list of all registered user accounts for the Web site.
* Change the authentication method that is used by the Web site.

Web Site Administration Tool Application Tab:

Use the Application tab of the Web Site Administration Tool (Web Site Administration Tool) to manage commonly used settings related to the Web application and site.

Use the Application tab to manage the following:

* Application settings name/value pairs.

Application settings are name/value pairs that represent configurable values text in the Web application. Use application settings to store custom application configuration information, such as file paths, XML Web service URLs, commonly used text, or any information that you want to maintain in a central location and be able to change easily.

* Simple Mail Transfer Protocol (SMTP) settings.

If the Web site requires the ability to send e-mail (for example, to send users a password), you must specify the SMTP server that your site uses.

* Application status.

You can take your application offline (shut it down) to perform maintenance or to bring a new Microsoft SQL Server Express edition database online.

* Debugging and tracing.

Debugging and tracing let you diagnose and fix problems with the Web application, and are primarily intended to be used during development. Enabling debugging causes the pages in the Web site to be compiled with information that the .NET Framework or the Microsoft Visual Studio debugger can use to step through your code. Enabling tracing causes pages to generate information about individual Web requests, the HTTP headers that are sent with the requests, the state of controls on the page, and other details about page processing.

Web Site Administration Tool Provider Tab:

Use the Provider tab of the Web Site Administration Tool to manage how ASP.NET stores the data for application features, such as user accounts, roles, and other settings.

ASP.NET uses provider classes to manage data storage for various features, such as membership and role management. A provider class is a component that exposes specific ASP.NET functionality, such as managing user accounts in membership. Each of these application features requires data storage. Although each provider class performs the same application features, each provider class can store data in different ways. For example, different provider classes for ASP.NET membership might store user account information in the following different ways:

* In a Microsoft SQL Server database.
* In Microsoft Windows Active Directory directory service.
* In a custom user database.

Each of these provider classes still performs the same core set of tasks for managing user accounts and you interact with the provider classes in your application in the same way.

Web Site Administration Tool Internals:

Describes the internal workings of the Web Site Administration Tool.

The Web Site Administration Tool provides a Web interface for managing the configuration settings of an individual Web site. Configuration settings are stored in Web.config files, which are located in the root folder of a Web site. The Web Site Administration Tool modifies the Web.config file when changes are made to the default configuration. If no Web.config file exists, and changes are made in the Web Site Administration Tool, a new configuration file is created.

Besides the creation and manipulation of Web.config files, the Web Site Administration Tool also configures site provider databases that support certain features.

When you use the Web Site Administration Tool for the first time, it creates a default Web.config file for the application to configure. The Web Site Administration Tool also creates the default site database provider, which is a local database in the App\_Data folder for the Web site.

PRACTICAL 22

Create a Webservice in Visual Studio to convert temparature in Fahrenhit into temperature in Celcious.

Default2.aspx.cs:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

public partial class Default2 : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e)

{

}

protected void Button1\_Click(object sender, EventArgs e)

{

localhost.temprature t = new localhost.temprature();

TextBox2.Text = t.temp(TextBox1 .Text );

}

}

temperature.cs:

[WebService(Namespace = "http://tempuri.org/")]

[WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1\_1)]

// To allow this Web Service to be called from script, using ASP.NET AJAX, uncomment the following line.

// [System.Web.Script.Services.ScriptService]

public class temprature : System.Web.Services.WebService {

public temprature () {

//Uncomment the following line if using designed components

//InitializeComponent();

}

[WebMethod]

public string HelloWorld() {

return "Hello World";

}

[WebMethod]

public String temp(String a) {

Double b = Double.Parse(a);

b = b \* 9 / 5 + 32;

return b.ToString();

}

}



PRACTICAL 24

Create a mobile application using visual studio 2008

Form1.vb

Public Class Form1

Private Sub Button1\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click

MsgBox("Hello World..!!")

End Sub

Private Sub Label1\_ParentChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Label1.ParentChanged

Label1.Text = "Button '" + Button1.Text + "' clicked at " + DateTime.Now.ToLongTimeString()

End Sub

End Class

Design:



Output :

