What Is .NET?

Overview

- Types of software systems
- The .NET architecture
- Managed code and CLR
- Visual Studio .NET
- Types of applications

References

- http://www.devhood.com/training_modules/
Types of Software Systems

- **Desktop solutions**
  - Desktop software is the most commonly deployed “service solution” in existence.
  - It is a chameleon that takes many shapes, colors, and sizes depending on the service it is aimed to deliver.

- **Client/server solutions**
  - Desktop applications are broken into logical pieces and distributed throughout a network of computers.

- **Internet solutions**
  - The Internet is the computer.
  - A Web browser is the user interface or client, the Internet is the server.
Challenges of Software

What is the right operating system?
- Why a target platform?
- Why not the best platform for the solution (such as the one that offers the most complementary or required features)?

The operating system limits choice of development languages and tools.
- .NET is not available on Linux/Solaris platforms.

Are you experienced? (Language and tool experience)
Once you have made your choice there is no turning back!
Why .NET?

- The .NET framework is Internet-oriented.
- The .NET architecture is based on the Internet and for the Internet and can therefore be considered as first-class Internet citizen.
Internet Distributed Software

Second-Tier (database or other server)

Web Server

Client

Second-Tier

Web Server

Client

Second-Tier

Com S 430
Supported Standards

- Hypertext Transfer Protocol (HTTP)
- eXtensible Markup Language (XML)
- Simple Object Access Protocol (SOAP)
- HyperText Markup Language (HTML)
- TCP/IP
- Universal Description Discovery Integration (UDDI)
- Web Service Description Language (WSDL)
- Extensible Stylesheet Language (XSL)
Defining the .NET Framework

The .NET Framework is

- A software development environment
- A runtime engine for *Managed Code*
- A platform designed for Internet-Distributed software

The .NET Framework is an exciting new computing platform.
Abstract View of .NET

Windows/FreeBSD

.NET Framework

Common Language Runtime

Common Type System

Runtime

Languages: C#, Visual Basic, Perl

.NET Applications

Web Services

Building blocks

Service: .NET & COM+

Enterprise Servers

SQL Server

...
.NET Components

In the .NET framework, the unit of deployment is an assembly.

An assembly contains:

- Code that is executed on a virtual machine for the common language runtime (CLR),
- Metadata,
- Additional resources.
A First Look at Managed Code

HelloGUI.cs:

```csharp
using System.Windows.Forms;
using System.Drawing;

class MyForm : Form{
    public static void Main(){
        Application.Run(new MyForm());
    }

    protected override
    void OnPaint(PaintEventArgs e){
        e.Graphics.DrawString("Hello World!",
            new Font("Arial", 35),
            Brushes.Blue, 10, 100);
    }
}
```

program entry point

new paint method
Compile & Run

c: \> csc /target:winexe HelloGui.cs

Hello World!
Managed Code and the CLR

- The Common Language Runtime (CLR) is a runtime engine
  - Manages .NET Code (such as C# applications)
  - Provides features such as memory management, thread management, object type safety, security, etc.
  - Is a part of the .NET Framework

- Managed code
  - Code that targets the CLR
  - Any .NET Language, including C#, Visual Basic, C++, Java, Cobol, etc.
The CLR and Managed Code

Legacy Software
(unmanaged code)

Common Language Runtime
(JIT compilation, memory management, etc.)

Windows
(or other operating system)

Managed Executable

Reusable Managed Components
IL and Metadata

- All Managed executables consist of
  - Intermediate Language (IL) instructions
  - Metadata

IL
- CPU independent machine language
- Just-in-time compiled at runtime

Metadata
- Structured information
- describes programming constructs including
  - Classes definitions, field and method definitions, parameter lists, return types, etc.
From Source Code to Managed .exe

HelloGui.cs

```csharp
using System.Windows.Forms;
using System.Drawing;

class MyForm: Form{
    public static void Main(){
        Application.Run(new MyForm());
    }
}

protected override
    void OnPaint(PaintEventArgs e){
        e.Graphics.DrawString("Hello World!",
            new Font("Arial", 35),
            Brushes.Blue, 10, 100);
    }
```

C# Compiler

HelloGui.exe

A Managed Application
Just-in-Time Compiling

- All managed code runs in native machine language

- However, all managed code is made up of IL and metadata

- The CLR JIT-compiles the IL and metadata
  - At execution time
  - Executed directly by CPU

- Allows for the best of both worlds
  - Code management features
  - Performance of full-speed execution
Executing a Managed Application

At execution time the IL and Metadata are JIT compiled

The CPU executes the JIT-compiled machine code directly
Automatic Memory Management

- The CLR manages memory for managed code
  - All allocations of objects and buffers made from a *Managed Heap*
  - Unused objects and buffers are cleaned up automatically through *Garbage Collection*

- Some of the worst bugs in software development are not possible with managed code
  - Leaked memory or objects
  - References to freed or non-existent objects
  - Reading of un-initialized variables

- Pointer-less environment
Cross Language Support

- The .NET Framework supports many languages.
- Any compiler can be modified to emit managed executables.
  - IL and metadata
- Languages that target the .NET Framework
  - C#, Visual Basic, C++, Java, PERL, COBOL, SmallTalk
  - Dozens more existing and on the way
- Regardless of source language, all managed code can
  - Use the same tools
  - Use the same reusable components
  - Take advantage of features of the CLR
- Developers use their language of choice.
Visual Studio .NET

- A Integrated Development Environment
  - To develop managed and unmanaged applications
  - Supports C#, C++, Visual Basic, JavaScript
  - Many useful tools and wizards

- Not part of the .NET Framework
  - Not necessary to build or run managed code
  - The .NET Framework SDK includes command line compilers
Creating a New Project

Microsoft Development Environment (design) - Start Page

Start Page

Get Started
What’s New
Online Community
Headlines
Search Online
Downloads
XML Web Services
Web Hosting
My Profile

Project

Name
Server
Gen
Cons
Act

Template

Visual C# Projects
Setup and Deployment Projects
Other Projects
Visual Studio SOLUTIONS

Windows Application
Class Library
Windows Control Library
ASP.NET Web Application
ASP.NET Web Service

A project for creating an application with a Windows user interface

Name: HelloGUI
Location: C:\Courses\Fall2002\Com430\Examples\HelloGUI
Project will be created at C:\Courses\Fall2002\Com430\Examples\HelloGUI.

More    OK    Cancel    Help

Com S 430
First Look
Running the Project

Hello World!
The Framework Class Library

- Included with the .NET Framework is a set of .NET Framework Class Library (FCL) assemblies that contains several thousand type definitions.
  - Classes, interfaces, enumerations and structures
- Types for
  - Common tasks such as collections, file IO, memory and thread management
  - GUI and window manipulation
  - Web form and web service applications
- Totally object oriented toolbox for developers
  - Ships as part of the .NET Framework
Kinds of Applications

The CLR and the FCL allow developers to build the following kinds of applications:

- XML Web services
- Web Forms
- Windows Forms
- Windows console applications
- Windows services
- Component library
ASP.NET XML Web Services

XML Web services are methods that can be accessed over the Internet.

XML Web services consist of two parts:

- The XML Web service entry point and
- The code that implements the XML Web service functionality.
ASP.NET Web Forms

Web Forms are HTML-based applications (Web sites).

Typically, Web Forms applications will make database queries and Web service calls, combine and filter the returned information, and then present that information in a browser using a rich HTML-user interface.
Windows Forms

- Windows Forms are used to develop rich Windows GUI applications.

- Windows Forms applications are in fact classic Win32 desktop applications, the kinds that were traditionally developed in Visual Basic, Visual C++, Delphi, etc.

- Examples include drawing or graphics applications, data-entry systems, point-of-sale systems, and games.
Windows Console Applications

- Applications with very simple user interface demands.

- Compilers, utilities, and tools are typical applications that are implemented as console applications (see csc).
Windows Services

Windows Services are service applications controllable via the Windows Service Control Manager (SCM).

Settings…
Administrative Tools…
Services…
Component Library

The .NET Framework allows you to build stand-alone components (types) that can be easily incorporated into any other .NET application.
Managed Applications - Summary

- **Common application**
  - GUI Windows applications (Windows Forms)
  - Console applications or command line applications

- **Web applications (ASP.NET)**
  - Web Forms applications
  - Web services

- **Scripted code**
  - Applications like word processors or database servers can host the CLR
  - Use any managed language (C#, VB, etc.) as a script or macro language

- **Wherever you use managed code.**
Open Source C# IDE

#develop (short for SharpDevelop) is a free IDE for C# and VB.NET projects on Microsoft's .NET platform.

It is open-source (GPL), and you can download both sourcecode and executables from http://www.icsharpcode.net.