Microsoft .NET

• what is .NET?

• "framework" for supporting web-based services
  - single run-time environment for programs written in a variety of languages
  - web forms for interfaces on web pages
  - XML, SOAP, WSDL, UDDI, etc., for web services

• development platform
  - single intermediate language as target for all languages
  - common type system
    all languages produce interoperable objects and types
  - common language runtime environment
    base class libraries accessible to all languages
    just in time compilation
  - control of deployment and versioning
    the end of DLL hell?
  - IDE for writing programs
  - significant new language, C#
  - evolution of Visual Basic and other languages

Why bother / who cares?

• primary focus of Microsoft software development
  - next stage after COM
  - likely to have major impact on how computing is done
certainly in Microsoft world

• interesting comparisons and contrasts with Java and J2EE

• ties in with other topics of 333
  - evolution of C, C++, Java -> C#
  - object-oriented programming
  - component-based software development
  - Visual Basic, user interfaces
  - web services
  - politics and economics of software
Java model

- Java language
  - derivative of C and C++
  - strictly object-oriented
  - garbage collection
- compiled into intermediate language ("byte code")
  - result stored in .class files
  - packages and JAR files for larger collections
- interpreted by Java Virtual Machine on host
  - local services provided by host system
  - enormous set of libraries in JRE
  - can be compiled into native instructions either ahead of time or "just in time"
- largely portable
  - types completely specified
  - main problems come from making use of services of host environment
  - "write once, run anywhere" is partially true
- applets for running code in web pages
- Java Server Pages (JSP) for server-based web transactions

.NET model

- multiple languages: C#, VB, C++, Jscript, ...
  - C# is a derivative of C, C++ and Java
  - VB.net is a significantly different version of VB
  - "managed extensions" for C++ that permit safe computation, garbage collection, etc.
- all are object-oriented
- all languages compile into same intermediate language ("MSIL")
  - types completely specified by Common Type System (CTS)
  - objects can interoperate if they conform to Common Language Specification (CLS) [a subset of CTS]
- IL compiled into native machine instructions
  - just in time compilation: no interpretation
  - local services provided by host system (Win 2K/XP)
  - enormous set of libraries
- not portable
  - tightly integrated into Windows environment
- web forms for GUI components on web pages
- ASP.NET for server-based web transactions
Common Language Runtime (CLR)

- all languages compile into IL that uses CLR
- common services:
  - memory management / garbage collection
  - exceptions
  - security
  - debugging, profiling
- access to underlying operating system

C# programming language

- based on C, C++ and Java
  - Microsoft does not stress the Java contribution
  - "An evolution of Microsoft C and Microsoft C++" (Visual Studio.NET documentation)

- "C# has a high degree of fidelity to C and C++"
  - everything is a class object (Java)
  - no global functions, variables, constants
  - garbage collection; destructors called implicitly (Java)
  - arrays are managed types (Java)
  - updated primitive types (Java)
    - char is Unicode character; string is a basic type
  - single inheritance and interfaces (Java)
  - ref, out parameter modifiers
  - try-catch-finally (Java)
  - delegate type (roughly, function pointers)
  - unsafe mode (pointers permitted)
  - some syntax changes:
    - `.` instead of `->` and `::`, switches don't fall through,
      `foreach` statement
  - no need for forward declarations (Java)
  - no headers or `#include` (Java)
  - `///` documentation comments (Java)
Separated at birth?

```java
public class hello {
    public static void main(String[] args) {
        System.out.println("hello, world");
    }
}
```

```csharp
using System;
public class hello {
    public static void Main(string[] args) {
        System.Console.WriteLine("hello, world");
    }
}
```

"echo" in Java and C#

```java
public class echo {
    public static void main(String[] args) {
        for (int i = 0; i < args.length; i++)
            System.out.println("Arg[" + i + "] = [" + args[i] + "]");
    }
}
```

```csharp
using System;
public class echo {
    public static void Main(string[] args) {
        for (int i = 0; i < args.Length; i++)
            Console.WriteLine("Arg[{0}] = [{1}]", i, args[i]);
    }
}
```
fmt in Java

```java
import java.io.*;
import java.util.*;

public class f {
    String line = "";
    String space = " ";
    int maxlen = 60;

    public static void main(String args[]) {
        f t = new f();
        t.runf();
    }

    public void runf() {
        String s;
        try {
            BufferedInputStream bin =
                new BufferedInputStream(System.in);
            DataInputStream in = new DataInputStream(bin);
            while ((s = in.readLine()) != null) {
                StringTokenizer st = new StringTokenizer(s);
                while (st.hasMoreTokens())
                    addword(st.nextToken());
            }
        } catch (Exception e) {
            System.err.println(e); // eof
        }
        printline();
    }

    public void addword(String w) {
        if (line.length() + w.length() > maxlen)
            printline();
        line += space + w;
        space = " ";
    }

    public void printline() {
        if (line.length() > 0)
            System.out.println(line);
        line = "";
        space = " ";
    }
}
```

fmt in C#

```csharp
using System;
using System.IO;

namespace fmtcs
{
    class fmt {
        int maxlen = 60;
        string line = "";

        static void Main(string[] args) {
            new fmt(args[0]);
        }

        fmt(string f) {
            string inline;
            Stream fin = File.OpenRead(f);
            StreamReader sr = new StreamReader(fin);
            for (inline = sr.ReadLine(); inline != null;
                inline = sr.ReadLine()) {
                string[] inwords = inline.Split(null);
                for (int i = 0; i < inwords.Length; i++)
                    addword(inwords[i]);
            }
            printline();
        }

        void addword(string w) {
            if (line.Length + w.Length > maxlen)
                printline();
            if (line.Length > 0)
                line += " ";
            line += w;
        }

        void printline() {
            if (line.Length > 0) {
                Console.WriteLine(line);
                line = "";
            }
        }
    }
}
```
Accessors (get/set members)

• syntax looks like public class variables
• semantics defined by calling get and set methods

class Thing {
    static bool fldstate;

    public static bool fldok {
        get { return fldstate; }
        set { fldstate = value; }
    }
}

Thing v;

if (v.fldok)
    v.fldok = false;

Indexers (get/set [] members)

• syntax looks like array access (v[i])
• semantics defined by calling get and set members with a subscript

class Awkarray {
    public Hashtable ht = new Hashtable();
    public Awk this[string name] {
        get {
            if (!ht.Contains(name))
                ht.Add(name, new Awk());
            return (Awk) ht[name];
        }
        set { ht.Add(name, value); }
    }
}

Awkarray aa = new Awkarray();

if (aa["whatever"] != null)
    aa["whatever"] = "a string";
Visual Studio.NET: the IDE

Visual Studio.NET

- handles multiple languages
- completely integrated with languages and run-time environment
  - can run compilers, etc., from command line too
- extensive online help
VB example

```vbnet
Module Module1
    Dim line As String
    Sub Main(ByVal args As String())
        Dim inline As String, words As String()
        Dim i As Integer
        line = ""
        FileOpen(1, args(0), OpenMode.Input)
        While Not EOF(1)
            inline = LineInput(1)
            words = inline.Split(Nothing)
            For i = 0 To words.Length - 1
                addword(words(i))
            Next i
        End While
        FileClose(1)
        printline()
    End Sub
    Sub addword(ByVal w As String)
        If line.Length + w.Length > 60 Then
            printline()
        End If
        If line.Length > 0 Then
            line = line & " "
        End If
        line = line & w
    End Sub
    Sub printline()
        If line.Length > 0 Then
            Console.WriteLine(line)
            line = ""
        End If
    End Sub
End Module
```

fmt in VB.NET

```vbnet
Module Module1
    Dim line As String
    Sub Main(ByVal args As String())
        Dim inline As String, words As String()
        Dim i As Integer
        line = ""
        FileOpen(1, args(0), OpenMode.Input)
        While Not EOF(1)
            inline = LineInput(1)
            words = inline.Split(Nothing)
            For i = 0 To words.Length - 1
                addword(words(i))
            Next i
        End While
        FileClose(1)
        printline()
    End Sub
    Sub addword(ByVal w As String)
        If line.Length + w.Length > 60 Then
            printline()
        End If
        If line.Length > 0 Then
            line = line & " "
        End If
        line = line & w
    End Sub
    Sub printline()
        If line.Length > 0 Then
            Console.WriteLine(line)
            line = ""
        End If
    End Sub
End Module
```
Other languages

• VB changes
  - now object-oriented
  - some obsolete features finally deleted (GOSUB)
  - library changes
  - arrays now origin 0, not 1 (upper limit is n, not n-1)
  - wizard to upgrade from previous version

• managed extensions for C++
  - garbage collected classes
    ```cpp
    __gc class M { public: int i; }
    int main() {
      while (true)
        M *m = new M;
        // runs forever without exhausting heap
    }
    ```
  - __gc pointers point to managed items only
  - __value classes for small items with short lifetimes
  - System::String type: `"this is a string"
  - etc.

Other worlds

• access to COM object from .NET client
  - .NET client calls COM object through a wrapper
    `Runtime Callable Wrapper`
  - callable at runtime (no prearrangement needed)
  - wrapper makes COM object look like it is a .NET object
  - and makes .NET client look like a COM client

• access to .NET components from COM
  - less common case, probably
  - COM object calls .NET object through a wrapper
    `COM Callable Wrapper`
  - makes .NET object look like a COM object
Assemblies

- "fundamental unit of deployment, version control, reuse, activation scoping, and security permissions for a .NET-based application"
- collection of type and resource info
- (usually? always?) packaged as .exe or .dll
  - may contain other files, including .exe and .dll
  - executable parts are in MSIL, not native code
- each assembly contains a "manifest" with
  - name, version of the assembly
  - file table: other files in the assembly
  - external dependencies
- greatly reduce need for Windows registry
  - program and components self-contained
  - can often remove an application just by removing the files

Deployment, versioning

- prior to .NET, installing an application requires
  - copying files to multiple directories
  - making entries in registry
  - adding shortcuts to desktop and menus
- backing up, moving, removing an application requires an installer program
- "DLL Hell": shared libraries get out of sync with apps that need them
  - new installation breaks existing programs that rely on properties of old DLL
  - new installation overwrites newer DLL with older one
- assemblies provide strong internal naming/typing
  - ensure that the right library is being used
  - assembly can specify versions of external references that it needs to work properly
  - CLR loads proper one
  - can have old and new versions working side by side
**J2EE (Java 2 Enterprise Edition)**

- **Java comes in 3 editions**
  - J2SE standard edition (what we all use)
  - J2ME embedded edition (phones, PDAs, ...)
  - J2EE enterprise edition (big systems)
  - same language but different libraries and programming models

- **J2EE aimed at e-commerce**
  - browse through offerings
  - select item, gather billing & shipping info
  - check inventory (maybe trigger supply chain)
  - validate financial info
  - arrange shipping, get tracking number
  - ...

- **usually complicated multi-tier structures**
  - need toolkit of subsystems for building system
    - naming & directory services
    - distributed objects
    - database access, concurrency control, transaction integrity
    - security
  - need help in integrating and packaging components
    (Java components called "beans")

**J2SE/J2EE vs .NET**

- **technical**
  - trying to solve similar problems
  - Java is a single language solution
  - .NET supports multiple languages
  - Java builds on existing environments; portable
  - .NET deeply embedded in Windows, only runs there
  - JSP similar to ASP
  - creating web services more integrated in .NET:
    every program is potentially a web service

- **non-technical**
  - monopoly vs. benevolent dictatorship?
  - Sun is concerned that Microsoft will cut the ground out from under it as an enterprise software system
    lawsuit charges anti-trust violations, unfair competition
    that tries to damage Java (filed March 2002)
  - April 2004: Sun & Microsoft settle all legal issues,
    Microsoft pays Sun $1.6B
How I spent my summer vacation...

• install Visual Studio.NET

• learn C# by creating pathetic version of an AWK to C# translator

• upgrade existing VB programs to VB.NET

• do some simple performance trials

• create a visual interface to AMPL using pipes and sockets

• “supervise” two students re-doing a COS 333 project in ASP.NET

• report on what happened

Tentative conclusions

• C# is a reasonable language
  - easy to pick up basics if know Java
  - easy to convert Java statements to C#
  - batch mode compilation is easy

• VB.NET is too complicated
  - each new release has made it more complicated
  - wizard helps upgrade process but can’t handle lots of things

• C/C++ are not much changed
  - some minor problems compiling old programs

• Visual Studio.NET feels smoother and easier than Visual Studio 6
  - all languages are handled in a uniform way
  - good integration of visual and textual
  - some remarkable omissions (layout managers!)

• likely to be too hard to adapt or upgrade most existing programs to .NET
  - they may not port to older versions of Windows

• a reasonable choice for brand new implementations