Getting Started with Microsoft Visual Studio, .NET and C#

Today's learning objectives

- To be introduced to the .NET framework, including Visual Studio and C#
- To be able to describe characteristics of the Common Language Runtime
- To be able to describe the general characteristics of some important project templates
  - Console application
  - Windows Forms application
  - ASP MVC Web application

Visual Studio (VS)

- What is Visual Studio?
  - Visual Studio is an example of an "Integrated Development Environment" (IDE)
  - Think of an IDE as an application that combines code editor, compiler, debugging and other tools that make the design, maintenance, and documentation of large projects easier
- Why an IDE?
  - Maybe you've had the experience of how Java projects consist of multiple files for different classes
  - This is characteristic for large projects
  - In MVC applications we'll see that we have many types of files (for 'M', 'V', and 'C', and various aspects of the website template, system configuration and helper functions)
- VS lets us manage these features and content in a convenient way

Visual Studio and C#

- Helpful background reading
  - Visual Studio from the Microsoft Developer Network (MSDN) https://msdn.microsoft.com/en-us/library/dd831853(v=vs.120).aspx (particularly note the Visual Studio IDE User Guide and the Application Development in Visual Studio links).
  - C# introduction
    - http://msdn.microsoft.com/en-us/library/aa645597(v=vs.71).aspx
- VS supports multiple languages (C#, C++, Visual Basic, J#) in the one IDE
  - All with nearly identical functionality and performance because the use a common core
- VS produces many types of applications
  - It can also integrate with Microsoft Office applications (Word, Excel, etc.)

.NET Languages

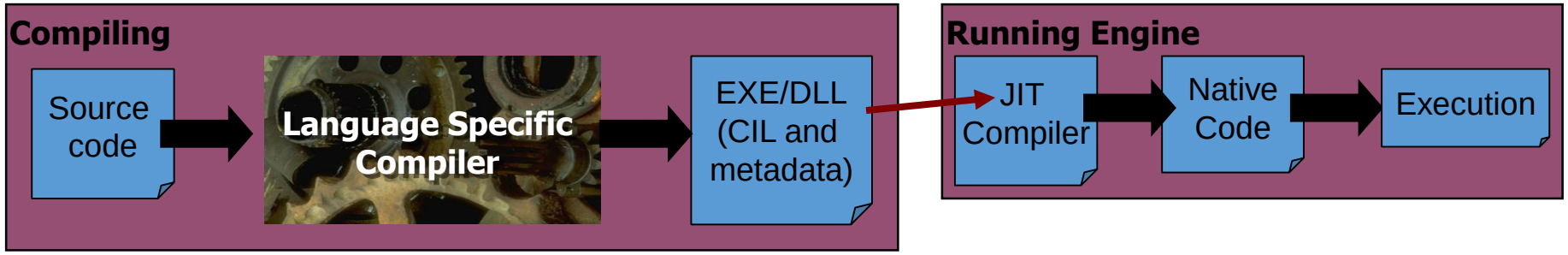
- All Visual Studio .NET languages are object-oriented
  - True inheritance and polymorphism (ability to redefine methods for derived classes) are supported
- No matter the language, all programs have a similar structure
  - Note that if you've done some kind of BASIC (e.g. VB 6) in the past - the language has changed considerably!
- C# (C-sharp) is relatively a new language
  - With syntax similar to C++, but also borrowing a lot of ideas from Java
- Visual J# is also a new language
  - with syntax similar to Java
- File structure is similar for all languages
  - Files are grouped into projects
- All programs compiled into Common Intermediate Language (CIL)
  - Also known as Microsoft Intermediate Language (MSIL)

The .NET Framework

- The .NET Framework define the rules for language interoperability and how an application is compiled into executable code.
- It is also responsible for managing the execution of an application created in any VS .NET language.
- The .NET Framework has two main components: the common language runtime (CLR) and the .NET Framework class library.
  - CLR
    - Manages memory, thread execution, code execution, code safety verification, compilation, and other system services
      - Memory allocation, starting up and stopping processes
  - The .NET Framework class library
    - Provides developers with a unified, object-oriented, hierarchical and extensible set of class libraries (‘application programmer interfaces’, APIs)

Execution Management

- The managed execution process includes the following steps:
  - Choosing a compiler
  - Compiling, code -> CIL/MSIL
    - Compiling translates the source code into CIL and generates the required metadata (this package is an 'assembly' (e.g. a DLL is an assembly)
      - The format is similar to assembly language but is hardware independent
  - Compiling, CIL -> native code
    - A just-in-time (JIT) compiler translates the assembly into native code (or runs it in a virtual machine)
    - Conceptually, the process is similar to the Java Virtual Machine
  - Running code
    - The CLR provides the infrastructure that enables managed execution to take place as well as a variety of services that can be used during execution.



Garbage Collection

- The CLR performs memory management
  - It manages the allocation and release of memory for an application
    - Automatic memory management can eliminate common problems,
      - such as forgetting to free an object and causing a memory leak, (common problems in C and C++ that lack garbage collection!) or
      - attempting to access memory for an object that has already been freed.
  - A contiguous area of memory allocated to a process is called the managed heap
    - Reference types are allocated from the managed heap
  - The CLR reclaims memory using the Garbage Collector (GC)
    - The GC examines variables to reclaim unused memory
      - It looks for memory without a corresponding variable (root)

Namespace

- Physical assemblies are organized into logical components called namespaces
- Namespaces are organized into a hierarchy
- Microsoft has divided VS .NET into two primary namespaces:
  - The **System** namespace contains components developed by the .NET team
  - The **Microsoft** namespace contains components developed by Microsoft but outside of the .NET development team
- Common Namespaces:
  - The ‘System’ namespace contains fundamental classes
  - System.Data namespace contains classes supplying data access capabilities
  - You’ll create namespaces for your own content (e.g. the data ‘model’ in your MVC application)

Creating a Console Application with VS

- Open Visual Studio
  - To create a new project called HelloWorldApp
    - Choose File->New Project
    - Select the project type: Visual C#
    - Select the project template: Console Application
    - Enter a name, Select a save Location and enter solution Name
    - Select Create directory for solution
    - Click OK – a program skeleton with a Main method that appears in the editor
- Insert the following code:

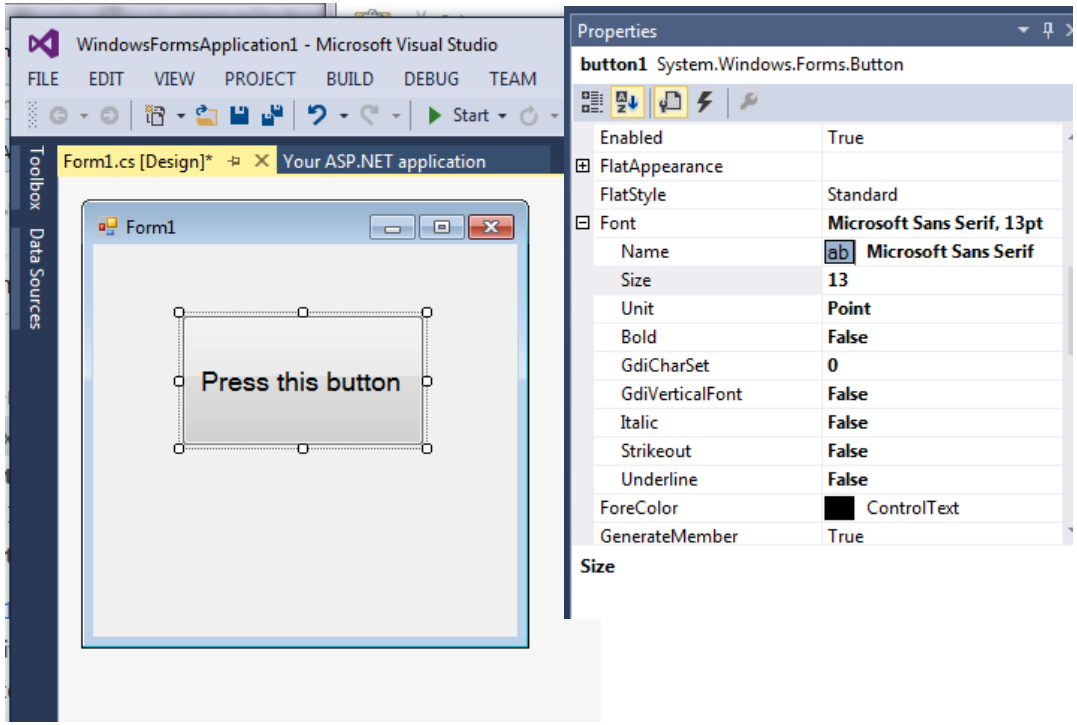
```
Console.WriteLine("Hello World");
Console.ReadLine();
```

  - Note:
    - The Main method is the entry point of your program, where the program control starts and ends
    - Insert Console.ReadLine() to the Main method which causes the program to pause until ENTER is pressed
- To run your application
  - Press F5 to run the application, or
  - Click the 'Start' button (with the green triangle ['play'] icon)

Creating a Windows Forms application

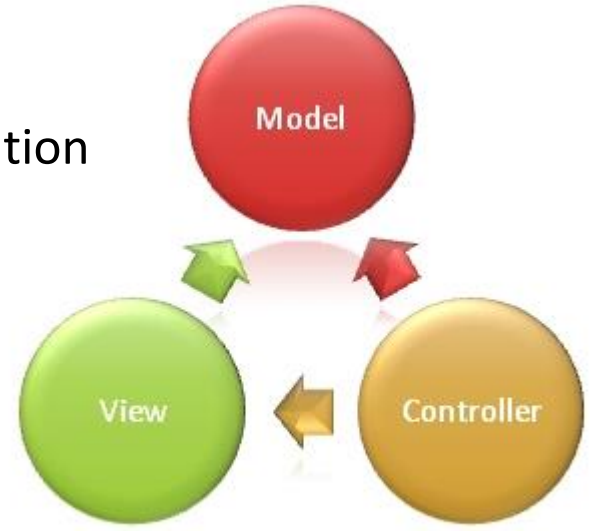
- This type of project starts with a blank form
- You can drag and drop useful components onto the form from the 'Toolbox' (available under the View menu)

- Use the Properties window to edit attributes of the selected control
- Double-click a control to add code into its event handler
  - E.g. MessageBox.Show("Hello, world!");



The Model-View-Controller approach

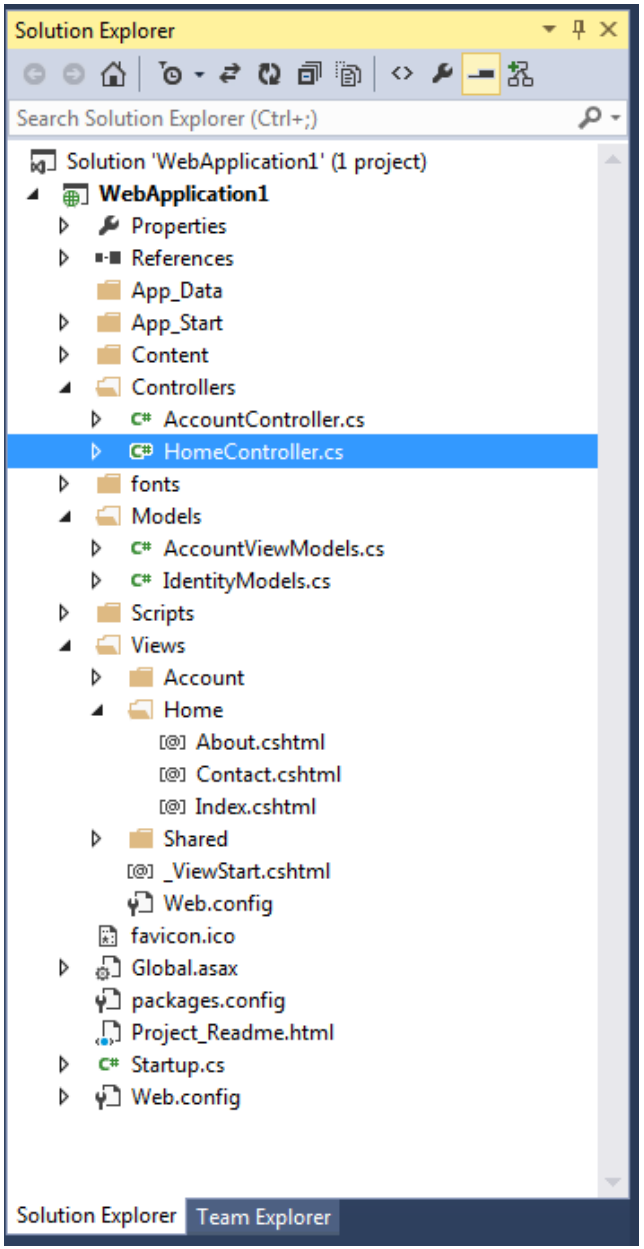
- MVC is an architectural pattern
  - Well, it's 'architectural' when you make a project with directories for the 3 areas of concern; conceptually, it's a *design* pattern
- Separates responsibilities in the application
  - Model – the data (including the connection and mapping to the DBMS) and its integrity constraints (e.g. legal values of variables in terms of the domain logic)
  - View – the rendering. What it looks like to the user and the detail of how they interact with the application
  - Controller – Handles and responds to user interaction. Uses the model and selects the next view to offer the user.



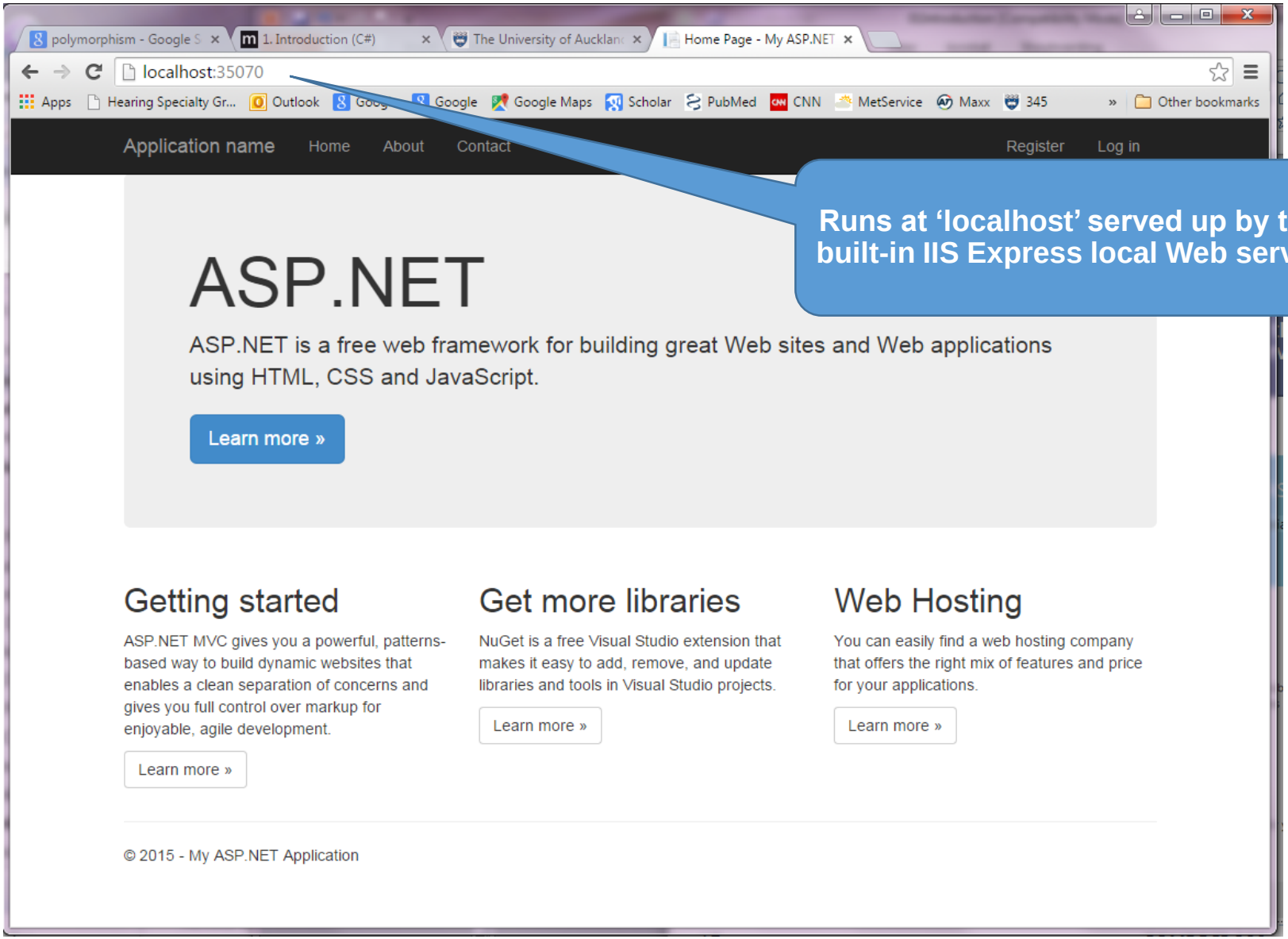
See http://www.asp.net/mvc/tutorials/older-versions/overview/asp-net-mvc-overview

An MVC Project in VS

- Open VS
  - File/New Project… select Visual C# -> Web -> ASP .NET Web Application and the MVC template
  - Give it a name and location
  - Press OK and listen to the hard drive grind
  - Solution Explorer (righthand side of screen in IDE) produces folders for the Controllers, Models and Views (alphabetical order)
    - The default template includes account management (login, new user) MVC elements



MVC application from the template



Conclusion

- .NET is a framework for creating applications featuring
  - Common language run-time (CLR)
  - .NET Framework class library
- The CLR provides execution management
  - Just-in-time compiling
  - Memory management (via garbage collection)
- Visual Studio (VS) is an integrated development environment (IDE) for .NET
  - Allows you develop in any of several languages, including VB and C#
- VS provides a number of application templates
  - Can create terminal, Windows or Web applications
- Next – we'll get into C# as a language…