C# For Developers

Course Syllabus

Companion Textbook:
C# 6 for Programmers
by Paul Deitel and Harvey Deitel

by Kenneth Brady
for Perscholas
Draft 1   January 27, 2017

Introductions

1. Introductions

Hellos, Names, and Why are we Here?

Why write code?
- Vocation
- Personal use / curiosity

What developers do (in the workplace):

- Walking through “other people's code”
- Debugging code
- Research (books & online)
- Rewriting code (“refactoring”)
- Communicating with your team – the Scrum
- Merging code
- Documenting code
- Testing code
    - Functionality (unit tests)
    - Reliability
    - Performance

Writing (new) code

Daily Course Outline

Scrum: Updating your team (~30 min)

Progress report on your class project
Things you've learned outside of class
Things you've attempted and need help with

New topics in C#: Review and discuss reading assignments (~1.5 h)

Break

Deep Dive: Walk-through a working application (~1 h)

Hands-on: Labs and course projects (~2 h)

1. Introduction to Programming

From Turing Machines to Tablets

Bits, Bytes, and Silicon: Machine Memory

Types of Computer Languages
- Rendered
- Interpreted
- Data-Defining / Storing / Modifying
- Compiled
- Hybrids

The Integrated Development Environment (IDE)
- Code Editors
- File Management
- Compilers
- Debuggers

Visual Studio
- Supports all .NET languages (C#, VB.NET, C++, F#), SQL, javascript, python
- Builds Windows, cloud and web applications
- Builds Apps for iOS and Android platforms (via Extensions)
- Community edition is free.

Student Presentations - Discussion

Lab 1a:	Installing VS 2015 Community Edition
Lab 1b:	Accessing GitHub
Lab 1c:	Uses of Version Control (Subversion)

2. Object-Oriented Programming

- Classes and Instances
- Encapsulation, Inheritance, and Polymorphism

C# and the .NET Framework

- Objects, Events, and Generics
- Framework Classes
- Common Language Runtime

Types of Applications

- GUIs and Apps
- Web Sites and Services
- Background Services
- Cloud Applications

Navigating Visual Studio

- Customizing Visual Studio
- Solutions and Projects
- Solution Explorer
- Projects
  - Properties, References, Resources, and Code-files
  - Designers and Code-behinds
- Debugging applications

Student Presentations

Lab 2: An implementation of a Hex-Dec-Oct-Bin Converter on 4 platforms:

- Console
- WPF / XAML
- WinForms
- ASP.NET

3. C# Programming Basics

- Code File Structure
  - Namespaces
  - Classes / Structs
    - Methods
    - Properties
    - Events
  - Comments

- The Console
  - The System Namespace
  - Executing and Debugging a Console Application
  - Compiler Errors

- Declaring, Initializing, and Using Variables
  - Common Types: *int*, *double*, *string*, *bool*, *decimal*
  - Operators
    - Arithmetic
    - Logical
    - String
  - C# Data Types
  - Type Aliases
  - Scope

Lab 4a:	Exception-handling in NumericConverter
Lab 4b:	Exception-handling and the Call Stack

4. C# Classes

- Declaring Classes
  - Members
    - Fields
    - Methods
    - Properties
    - Events
    - Access Modifiers
  - Nested Classes
  - *partial* Classes

- Naming Conventions
  - Pascal-casing (Classes, Properties, Methods, Events)
  - Camel-casing (method parameters and local variables)

- Field Declarations
- Method Implementations
  - Constructors
- Property Implementations
- Event Implementations

- Instantiating Classes
  - The *new* keyword
  - Constructors
  - The Stack and the Heap

- *static* Members

- Student Presentations

Lab 4a: Visualizing information in the UI (WPF)
Debugging programs
Visual Studio Extensions: The Magical C# Debugger

5. Workflow and Control Statements I

Types & Operators

Structured Programming vs. *goto*

if and *if else*

The *?:* ternary operator

while Loops and sentinels

Counter-controlled loops

Type conversions
  implicit
  explicit – the cast operator

++ and -- operators: prefix and postfix

Compound assignment operators: += -= *= /= %=

Lab 5: A Wave Generator

6. Workflow and Control Statements II

for loops

do … while loops

switch statements
- break statements
- default case

loop-termination and short-circuiting
- break
- continue

Boolean operators: == != && || ^ &

Lab 6: Reading a MIDI file

7. C# Methods

.NET Framework Class Library

System.Math

Method Parameters
- Multiple Parameters
- Overloading
- Argument promotion and casting
- Optional parameters
- *params* argument
- *ref* and *out* parameters

Expression-bodied methods and properties

The Call Stack
- Recursions
- The StackOverflow Exception

Value Types vs. Reference Types
- The Stack and the Heap
- Garbage Collection

A Tour of the .NET Framework Class Library

Using System.Random

Enumerations

Lab 7a: Pulling data from a web page or a REST service
Lab 7b: Displaying a histogram

8. Data Structures I: Arrays, Collections and Lists

Introduction to Generics and LINQ

Declaring and Creating Arrays

Indexing Arrays

Multidimensional and jagged arrays

System.Array Methods

Enumerating arrays: *foreach*
- IEnumerable and IEnumerable<T>
- System.Object

Querying an array with LINQ
- Extension Methods

System.Collections.Generic.List<T>
- List<T> Declaration, Properties, and Methods
- Implementing List<T>.Sort
- Algorithmic efficiency of Methods
  - Contains
  - Find
  - IndexOf
  - Add
  - Insert
  - Remove
  - Sort
  - BinarySearch

Lab 8: Displaying Filtered Information

9. Defining Classes and Structs

Introduction to Design Patterns

The Purpose of a Class
- Implementation goals:
  - fulfill purpose
  - correctness
  - simplicity
  - efficiency

Throwing Exceptions

The current instance: *this*

Constructors
- implicit
- overloaded
- construction logic

Composition
- Bringing capabilities together
- Examples

Destructors and Garbage Collections
- Memory Leaks
- IDisposable

static Class Members

readonly and *const* Fields

Object initializers (default constructor + public Properties)

Structs
- Choosing between class and struct
- Reference types and Value types
- Boxing of Value types

Extension Methods

Lab 9: Reading data from a Database
Introduction to SQL

10. Inheritance

- Base Classes and Derived Class
    - General → Specific
    - Examples from .NET Framework

- *protected* Members

- *virtual* and *override*

- Calling *base* Methods and Constructors

- Refactoring Code
    - Minimizing Code Duplication

- Object Methods
    - virtual bool Equals(Object obj)
    - static bool Equals(Object objA, Object objB)
    - virtual int GetHashCode()
    - static bool ReferenceEquals(Object objA, Object objB)
    - virtual string ToString()

Lab 10: Parsing Mathematical Expressions

11. Interfaces and Polymorphism

Subclasses and Superclasses

The “is a” relationship

The *is* operator

The *as* operator

The cast operator (<type>)

virtual and *override* properties and methods

abstract properties, methods and classes

sealed methods and classes

Interfaces

Lab 11: Polymorphism in the UI

12. Events

Files and Streams

delegate and *event* keywords

declaring and raising events

handling events

Event-driven software

The System.IO Namespaces
- File and Directory
- FileInfo and DirectoryInfo
- Stream and FileStream
- Path

Other Streams:
- MemoryStream
- NetworkStream

Serializing Objects
- ISerializable
- BinaryReader and BinaryWriter

Lab 12: Saving and Restoring Objects

13. Exception Handling

try, catch, and finally

throwing exceptions

The *using* statement and IDisposable.

The Exception class

The null-conditional operator *.?*

Nullable value-types

Exception filters

Lab 13a: Exceptions in the Debugger
Lab 13b: Exceptions in UI Applications

14. Working with Strings

What are C# strings?
- Escape characters
- Verbatim strings

The String class
- The indexer []
- String Comparisons
- Search for and creating substrings

The StringBuilder Class

The Character Class

Regular Expressions
- The Regex Class

Working with XML

Lab 14: Scraping a Web Page

15. Threads and Asynchronous Programming

CPUs, Threads and Processes
- When to use Threads

Communication between Threads
- Mutexes, WaitHandles and Semaphores
- Accessing data shared between threads
- Race conditions

The ThreadPool

System.ComponentModel.BackgroundWorker

async, *await* and the Task<> Class

Lab 15a: Accessing a WebService using a Thread
Lab 15b: Accessing a WebService using *async* and *await*

16. Programming in the Workplace

Product Design and Prototyping

Source Control
- Subversion
- GitHub

Agile Development, Iterations, and the Scrum

Visual Studio Extensions and 3d-Party Tools

Quality Assurance Testing

Production Release Cycles

Lab 16: Introduction to Cloud Computing and Google OCR

Web Resources

.NET Framework Class Library: https://msdn.microsoft.com/en-us/library/gg145045(v=vs.110).aspx

Classes: https://msdn.microsoft.com/en-us/library/x9afc042.aspx

Nested Types: https://msdn.microsoft.com/en-us/library/ms173120.aspx

Naming Conventions: https://msdn.microsoft.com/en-us/library/ms229045(v=vs.110).aspx

Get Involved

NYC .NET Developers:	https://www.meetup.com/NYC-NET-Developers
Newsletters:	https://www.simple-talk.com/