Advanced C#

Lecture 8

CS 638 Web Programming

Lecture overview

- Interfaces
- Exceptions
- Operator overloading
  - Indexers
- Delegates
- Partial classes

Interfaces & single inheritance

- Objects from derived classes can be used where objects from a base class are expected
- How can one build objects that can mimic two unrelated base classes?
  - Multiple inheritance (supported in C++)
  - Single inheritance + interfaces (C#)
- Interfaces differ from base classes
  - They do not provide bodies for any method
  - All their members are public
  - By convention the name of all interfaces starts with an "I"

Exceptions

- Using exceptions greatly simplifies code
  - Error handling code separated from code doing main work
  - Code doing main work assumes everything succeeds
- `throw` raises an exception
  - Accepts an exception object as argument
  - With no argument previous exception is re-thrown
  - All exception objects must be instances of a class derived (directly or indirectly) from System.Exception
  - If a method does not handle the exception it generates, it is terminated and the caller receives the exception
    - Unhandled exceptions percolate upward to the framework your code runs in and they may kill your program

Handling exceptions

- Enclose the section of code where an exception can could occur in a `try { }` block
- `catch(`*`ExceptionType`* `ex) { }` handles all exceptions of type *`ExceptionType`* (or derived)
  - Code within this block executed when exception caught
  - Multiple catch clauses allowed, exception handled by the first one matching its type
  - Defining specialized exception classes for specific errors enables specialized handling for those errors
  - Catch most specific exceptions first, most general last
- `finally{ }` encloses code you want run after the `try{ }` block whether exceptions occurred or not

Operator overloading

- Operators are implemented as static methods
  - E.g. "`operator +`", "`operator ==`", etc.
  - Can be overloaded just like other methods
- Operator overloading allows you to manipulate complex objects with the same syntax as for builtin types
  - It can increase legibility of code
  - When misused it can lead to counterintuitive behaviors that make code harder to understand

Indexers

- Used for overloading the `[ ]` operator
  - Syntax similar to properties (with accessors)
  - Not static
- Can use various types for the index between `[ ]`
  - E.g. strings
- Particularly useful for all types of collections

C# delegates

- They allow programs to manipulate references to methods
  - They are the equivalent of function pointers in other languages
- The `delegate` keyword is used for giving a name to a class of methods with the same return value and parameter list
- Delegates are used extensively in event-driven programming to specify event handlers

Manipulating delegates

- Composing delegates with the `+` operator builds a new delegate with all the methods of the delegates
  - Multicasting: when this delegate is called all methods of the old delegates are called in sequence
- Similarly one can remove a method from a delegate using the `–` operator
- Operators `+=` and `-=` also defined

Publishers and subscribers

- Event-driven programs are typically structured into publishers and subscribers of events
- Publisher classes define event handler delegates
- Publisher objects store an instance of the handler
- Subscriber objects register actual methods to handle the events defined by publisher objects
  - A subscriber's delegate can hold multiple handler methods
- When the publisher object notices a change it "generates an event" and invokes the handler(s)

Partial classes

- C# allows you to split a class among many source files by declaring it partial
- Useful feature when you want to give others the ability to add new fields and methods to your class
  - Different from adding derived class with more fields and methods
- They allow ASP.NET to separate "boilerplate" code generated automatically for new classes representing web page elements from code the user writes explicitly for those same web page elements