Module 1 Day 2

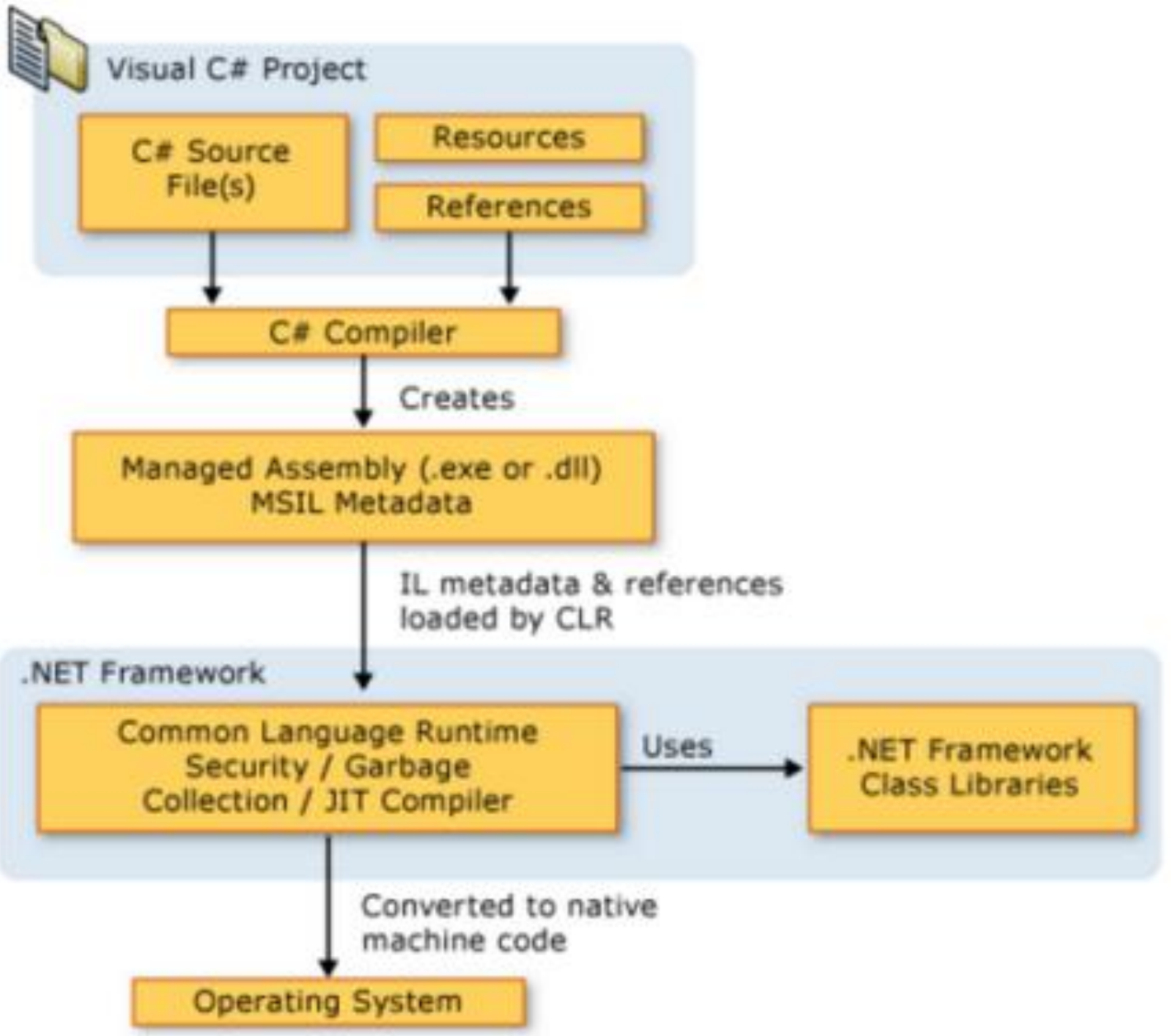
Variables and Data Types

C# and Microsoft .Net

- C# : a modern language derived from C and C++
- Managed Memory (garbage collection) is the big win over C/C++
- C# is compiled first into MSIL (intermediate language), then into machine code
- C# is one of a number of languages that can be compiled into MSIL, thus the runtime is called the Common-Language Runtime (CLR)
- The .Net Framework provides tons of added functionality from collection classes to security to data access
- Circa 2001

C# .Net Architecture

- Source-code is complied into “intermediate language” (MSIL) by the developer
- At runtime, MSIL is “just-in-time compiled” (JITted) into machine code by the Common Language Runtime (CLR)
- .Net Framework Class Libraries provide loads of functionality



Visual Studio 2017

- Full-featured Integrated Development Environment (IDE)
- Write code, compile run, test, debug
- Pull, push, merge and diff code to and from Git repos
- Project: creates a single binary “assembly” (.dll, .exe)
- Solution: a collection of related projects. The “top level” element in the VS IDE
- Let’s use the famous Hello World! To take a tour
  - Create a project and solution
  - Write code
  - Build and Run code

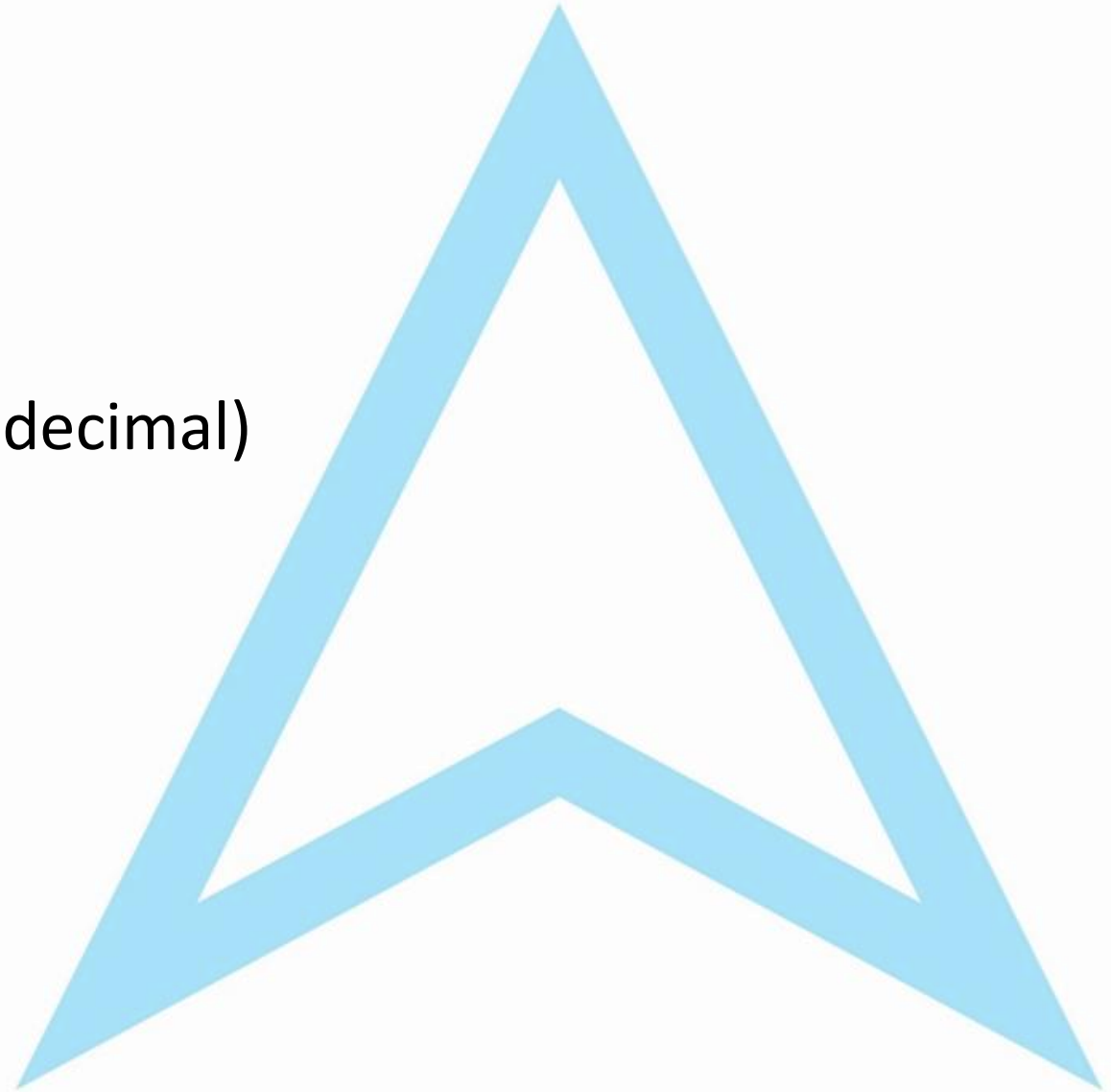
Variables

- A name for a location in memory
- Must be declared before it is used
- Type must be specified
  - http://book.techelevator.com/.net/05-introduction-to-programming/variables/05-data-types.html
- Declared only once; assigned multiple times
- Assigning a variable
  - Assignment statement
  - Assignment at declaration time
  - Const
- Variable is a “container”. The value is the “contents of the container”.

Computer Memory, Part 1

How Stuff is Stored

- Binary representation
- Whole numbers (byte, int, long)
  - Logical (bool: True = 1, False = 0)
- Fractional numbers (float, double, decimal)
- Characters (char)



Computer Memory, Part 2

Where Stuff is Stored

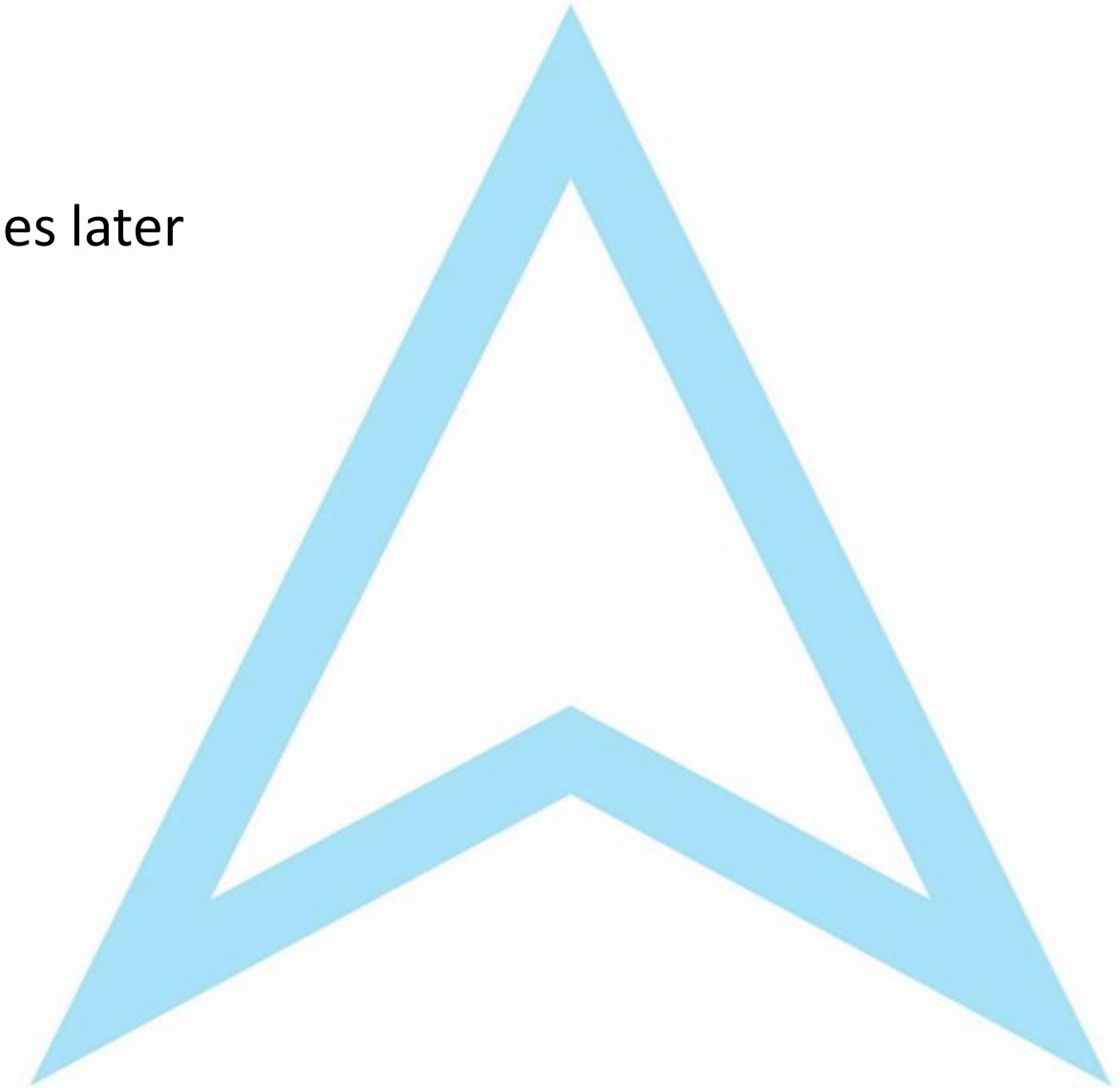
- Stack
  - Static memory allocation (variable size known at compile-time)
  - One “stack frame” per function call
  - Destroyed when function exits
  - Fast access
  - LIFO
- Heap
  - Dynamic memory allocation (determined at run-time)
  - Global in scope
  - Slower access as it can fragment
- C# Value types: on the stack
- C# Reference types: on the heap

C# Value Data Types

Reserved Word	.NET Type	Type	Size (bits)	Range (values)
byte	Byte	Unsigned integer	8	0 to 255
sbyte	SByte	Signed integer	8	-128 to 127
short	Int16	Signed integer	16	-32,768 to 32,767
ushort	UInt16	Unsigned integer	16	0 to 65,535
int	Int32	Signed integer	32	-2,147,483,648 to 2,147,483,647
uint	UInt32	Unsigned integer	32	0 to 4294967295
long	Int64	Signed integer	64	-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807
ulong	UInt64	Unsigned integer	64	0 to 18,446,744,073,709,551,615
float	Single	Single-precision floating point type	32	-3.402823e38 to 3.402823e38
double	Double	Double-precision floating point type	64	-1.79769313486232e308 to 1.79769313486232e308
decimal	Decimal	Precise fractional or integral type that can represent decimal numbers with 29 significant digits	128	(+ or -)1.0 x 10e-28 to 7.9 x 10e28
char	Char	A single Unicode character	16	Unicode symbols used in text
bool	Boolean	Logical Boolean type	8	True or False

Strings

- Reference Type (vs. Value type)
  - We'll talk more about Reference types later
- + operator concatenates strings
  - (this is called "operator overload")
- *** Code (1-9)



Expressions

- A construct that gets *evaluated* to a single value
- That value can be *assigned* to a variable
- Arithmetic expressions
  - +, -, *, /, %
- Precedence
  - *, /, %
  - +, -
  - Use () to impose precedence
- Arithmetic Shortcuts: +=, -=, /=, %=
- *** Code (10-15)

Data Type Conversion

- Implicit conversion
  - Done by the compiler
  - Type-safe
  - Smaller to larger values
- Explicit conversion
  - Could be dangerous, so compiler won’t do it without being told
  - Must be specified by the programmer
  - This is called Casting
- *** Code (16+)