Module 1 Day 2

Variables and Data Types

What makes an application?

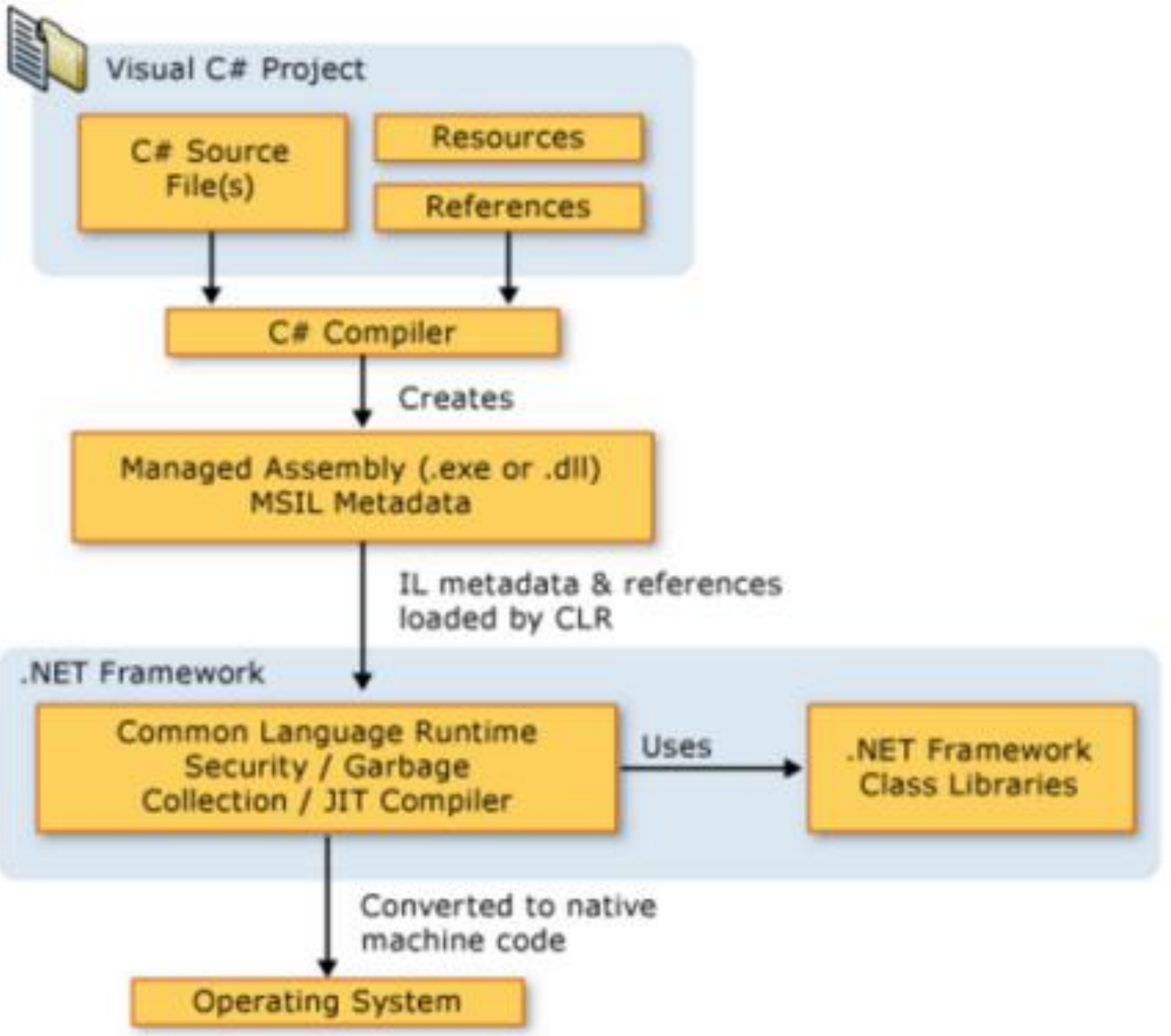
- Program Data
  - ➢ Variables & .NET Data Types
  - ❑ Arrays
  - ❑ More Collections (list, dictionary, stack, queue)
  - ❑ Classes and objects (OOP)
- Program Logic
  - ❑ Statements and expressions
  - ❑ Conditional logic (if)
  - ❑ Repeating logic (for, foreach, do, while)
  - ❑ Methods (functions / procedures)
  - ❑ Classes and objects (OOP principles)
  - ❑ Frameworks (MVC)
- Input / Output
  - User
    - ❑ Console read / write
    - ❑ HTML / CSS
    - ❑ Front-end frameworks (HTML / CSS / JavaScript)
  - Storage
    - ❑ File I/O
    - ❑ Relational database
    - ❑ APIs

C# and Microsoft .NET

- C# : a modern language derived from C and C++
- Circa 2001
- 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

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

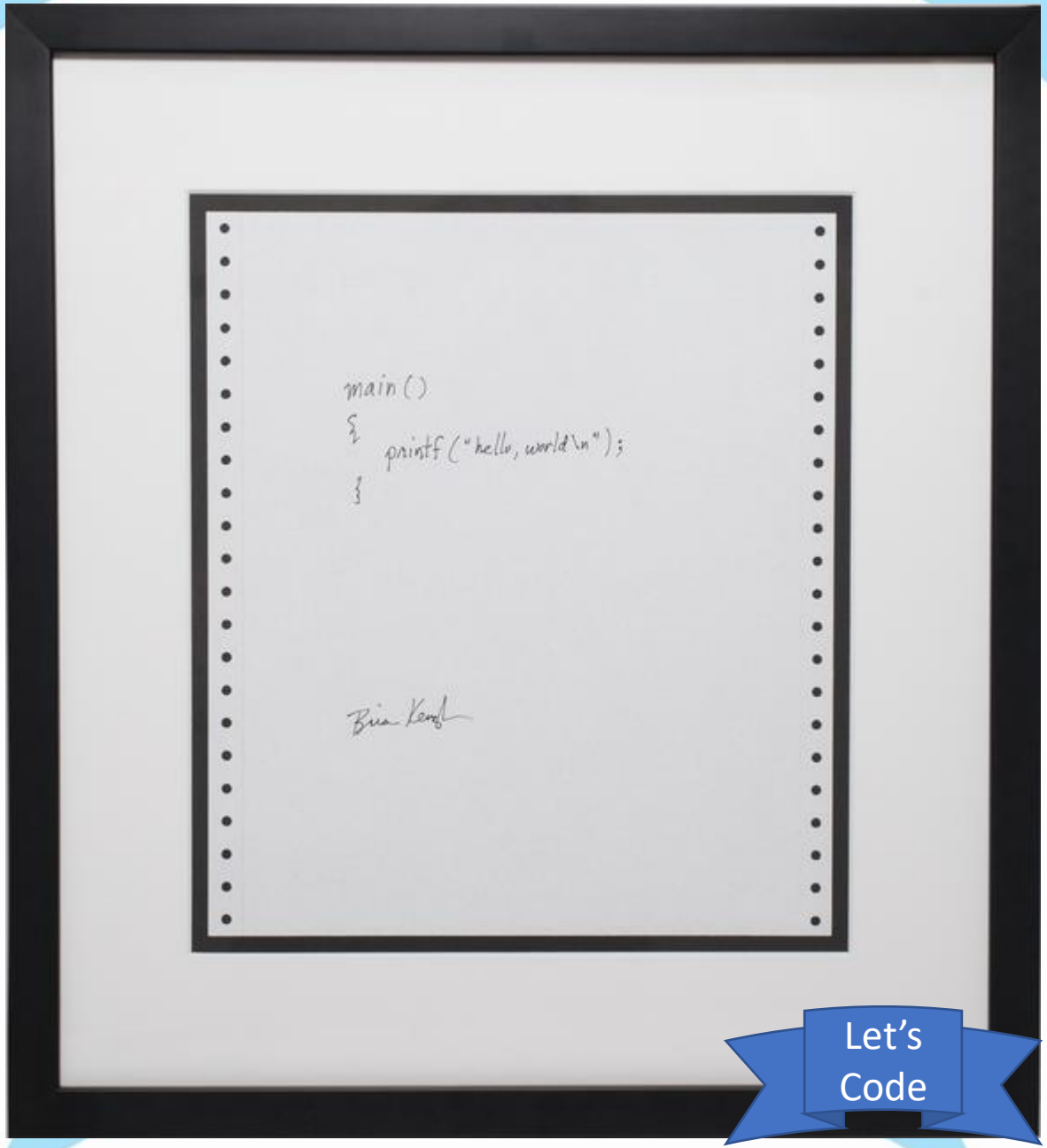


Visual Studio 2019

- 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

Visual Studio 2019

- Let's use the famous Hello World! To take a tour
  - Create a project and solution
  - Write code
  - Build and Run code



Variable

- A name for a location in memory
- Stores "data" inside

Variables – Declaring and Assigning

- A name for a location in memory
- Must be declared before it is used
- Type must be specified
  - https://v2-2-techelevator-book.netlify.app/content/introduction-to-programming-ool.html#common-data-types
  - 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”.
- 3 steps to using a variable: declare – allocate – assign

```
// Declare/allocate, then assign
int age;
age = 25;

// Declare, allocate and assign
int height = 71;

// Value cannot be changed
const int daysInWeek = 7;
```

C# Value Data Types

Let's Code

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

Lecture Code (1-9)

Value types vs. Reference type

- Value types
  - Generally take up a small amount of memory (a few bytes)
  - Size can always be determined at compile-time
  - Int, long, bool, char, float, double, decimal
- Reference types
  - Generally have the potential to require more space
  - Compiler may not be able to determine the amount of space need (done at runtime)
  - String (more to come later ☺)

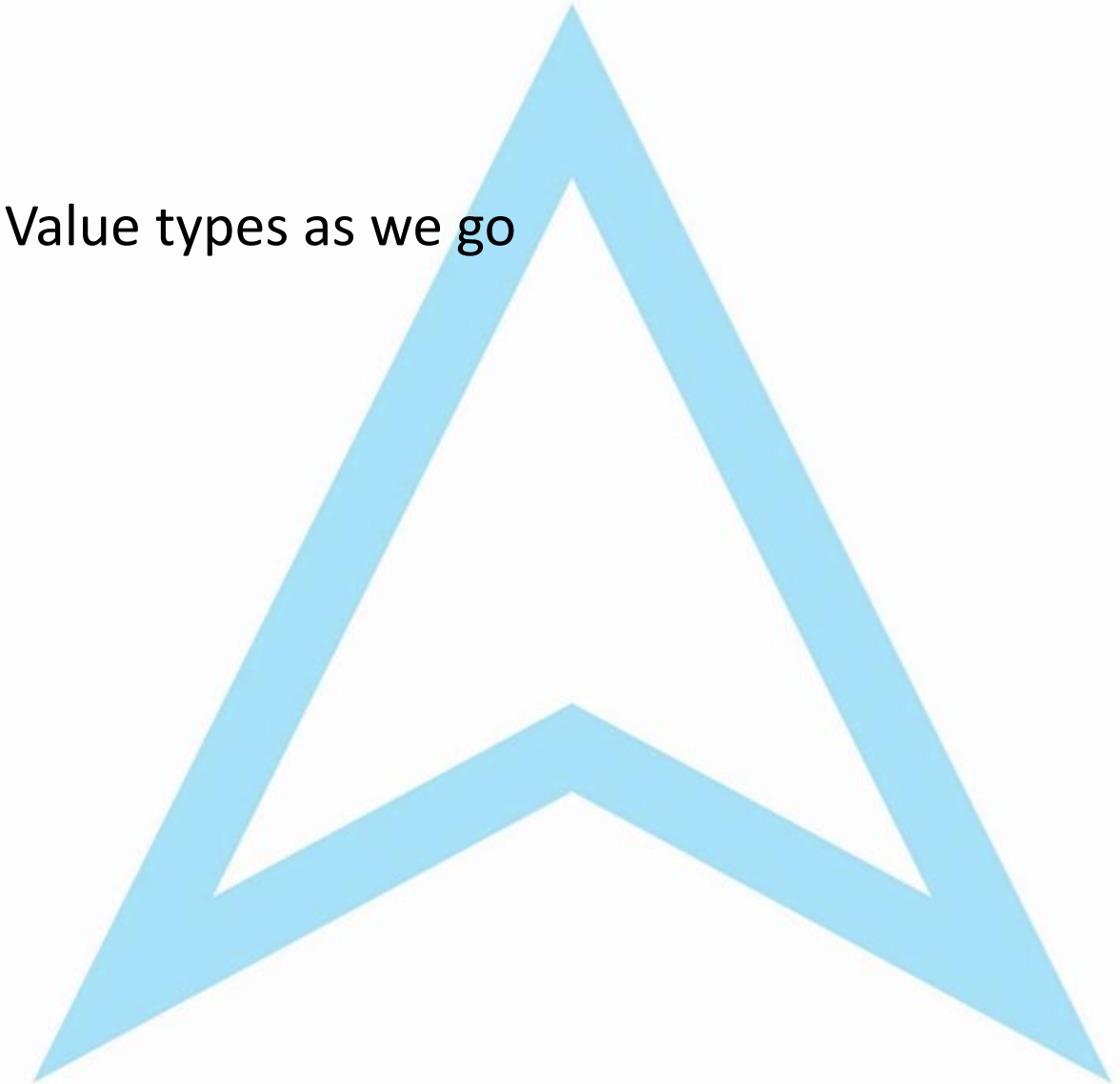
Binary Representation of Data

- Whole numbers (byte, int, long)
  - 00000000 00000000 00000000 00001110 = 14
- Logical (bool: True = 1, False = 0)
  - 0000001 = True
- Fractional numbers (float, double, decimal)
  - Float: 1bit sign, 8bits exponent, 23bits mantissa $(-1)^s * m * 2^{e-127}$
  - 00111101 10101110 00010100 01111011 = 0.085
- Characters (char)
  - 00000000 01000001 = 'A'

- What does 01000010 equal?

Strings

- Reference Type
  - We'll talk more about Reference vs. Value types as we go
- + operator concatenates strings
  - (this is called "operator overload")



Expressions

- A construct that gets *evaluated* to a single value
- That value can be *assigned* to a variable
- Arithmetic expressions
  - +, -, *, /
  - % (modulo)
    - If a number "n" is even, then (n % 2) is 0
    - If a number "n" is odd, then (n % 2) is 1
    - 35 % 10 is 5 (quotient is 3, remainder 5)
- Precedence
  - *, /, %
    - 4 + 5 * 10 equals 54
  - +, -
  - Use () to impose precedence
- Arithmetic Shortcuts: +=, -=, /=, %=

Lecture 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
- Real Type Literals
  - Any real literal (e.g. 3.14) is presumed to be Double
  - You can designate a literal to be Decimal with 'M', Float with 'F'

Lecture Code (16+)