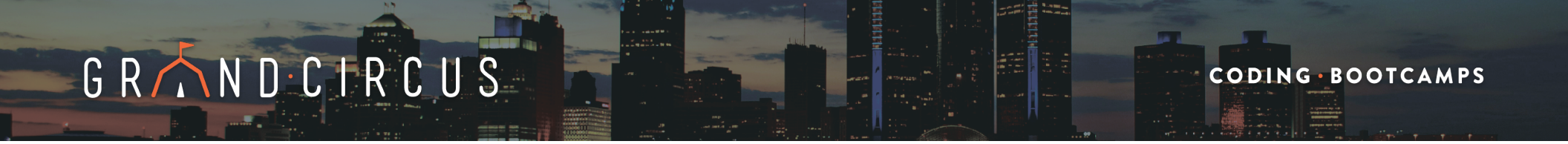


C#

BOOTCAMP

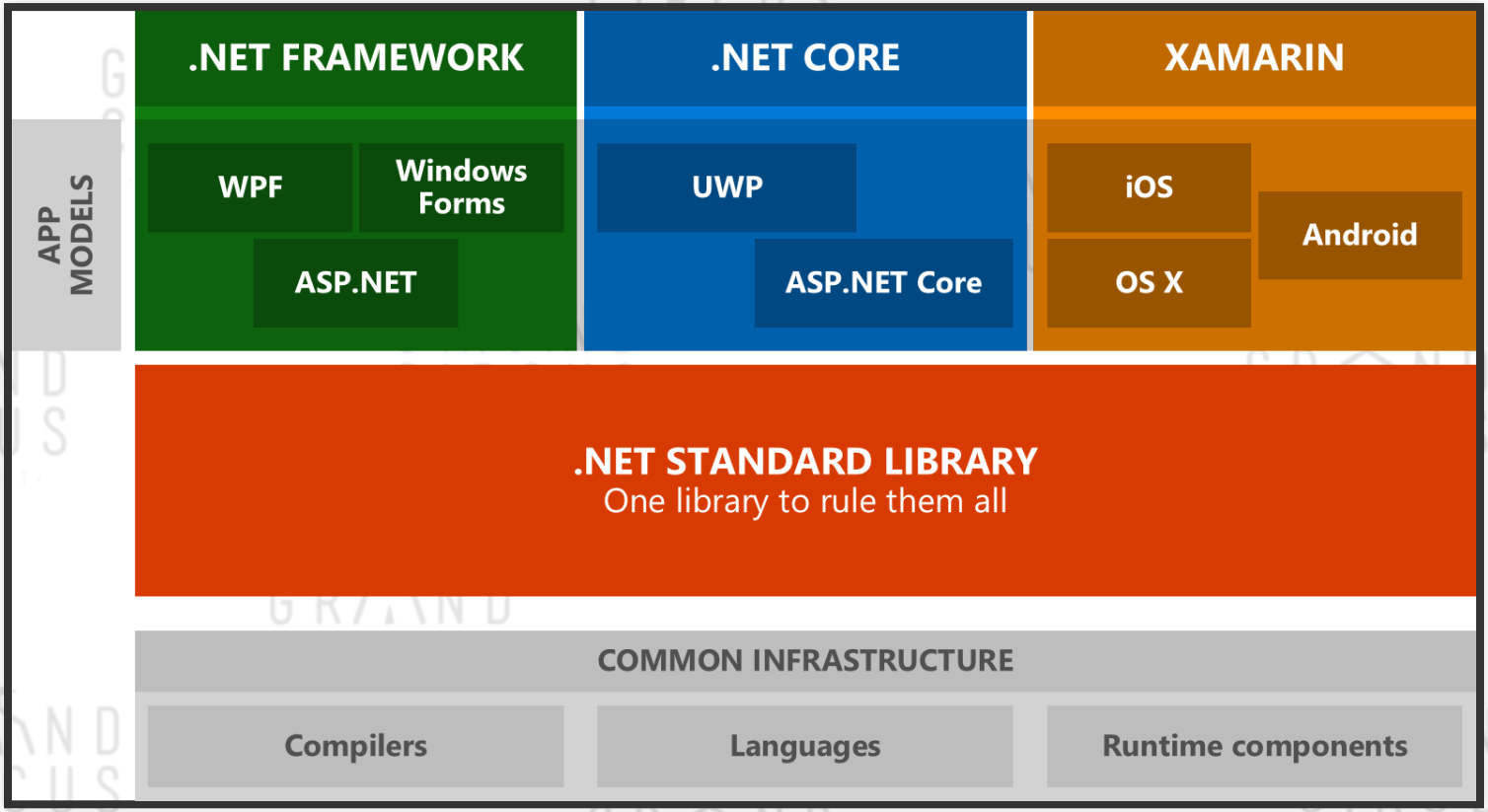
INTRO TO C#

.NET FRAMEWORK

.NET framework is a software framework that is developed by Microsoft.

.NET is used to develop many applications including Web, Desktop, and Mobile Applications.

.NET FRAMEWORK



ABOUT C#

C# .NET

- Introduced by Microsoft back in 2000
- Object-Oriented
- Derives much of its syntax from C and C++, but has fewer low-level facilities than either of them
- Most recent version is C# 7

C# COMPARED TO JAVA AND C++

- C# is very similar to C++ and JAVA in syntax.
- C# and JAVA have a similar compilation architecture.

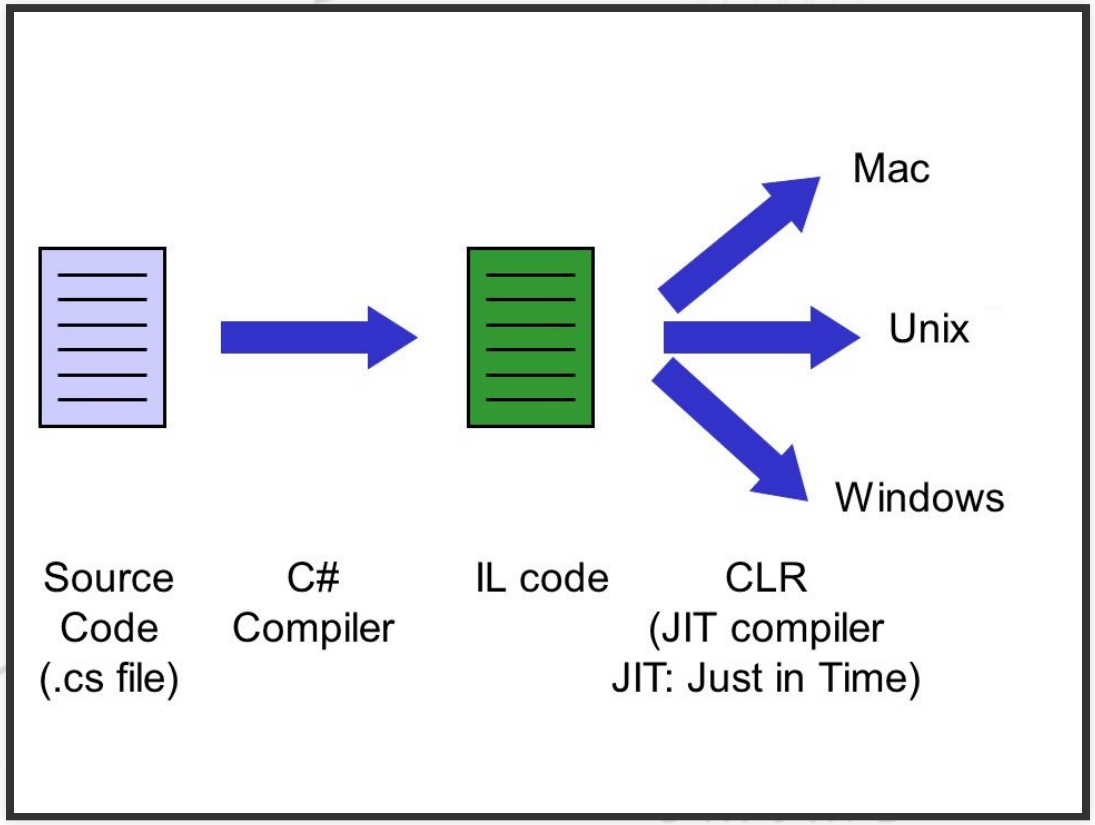
C# COMPARED TO JAVA AND C++

- C++ and C# run faster than Java, but Java is getting faster with each new version.
- Both Java and C# handle most memory operations automatically, while C++ programmers must write code that manages memory.

OPERATING SYSTEMS SUPPORTED BY C# .NET

- Windows
- Linux and OS X with the .NET MONO

HOW C# COMPILES AND INTERPRETS CODE



INTEGRATED DEVELOPMENT ENVIRONMENTS

Visual Studio is the Most Commonly used IDE for developing C#

Other IDEs include: SharpDevelop and MONODevelop

Intro to C#

Declaring the Main Method

```
public static void main(String[] args) {
//Statements
}
```

PRINTING DATA TO THE CONSOLE

We can use the Console class to do output:

Example:

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

GETTING INPUT FROM THE CONSOLE

Getting input from the console is also done using the Console class, but we need to parse the input into the proper type.

Example:

```
String input = Console.ReadLine();
int x = int.Parse(input);
```

STATEMENTS

Statements direct the operation of the program.

Statements end with a semicolon (;), but blocks of code between curly braces ({}) simply end with the right curly brace.

It is best practice to use indentation and spacing to align statements and blocks of code. This makes your code much easier to read.

COMMENTS

Comments are used to document what the code does. This is useful not only for other programmers who may need to maintain your code, but future you as well!

A single-line comment begins with two slashes (//).

A block comment starts with a slash and an asterisk (/*) and ends with the same two symbols in reverse (*/).

IDENTIFIERS

Any name created in C# is called an identifier. Identifiers may be used to name classes, methods, variables, and so on.

A keyword is one that is reserved by the language. It may not be used as an identifier.

NAMING AN IDENTIFIER

Start each identifier with a letter or underscore. Use letters, underscores, or digits for subsequent characters.

Don’t use keywords!

DECLARING AND INITIALIZING VARIABLES

We use variables to store data. For each variable we use, we must declare it (specifying its data type):

```
type variableName;
```

In order to initialize a variable, we have to assign it a value:

```
variableName = value;
```

CONSTANTS

A constant stores a value that cannot change as the program executes.

Declare a constant by preceding the normal initialization with the keyword const and capitalizing all letters in the name of the variable.

DATA TYPES

VALUE TYPES:

- Variables have a value
- Space allocated on stack
- Examples: int, double, float.

DATA TYPES

REFERENCE TYPES:

- Variable is just a reference allocated on the stack
- Reference is a “type-safe” pointer
- Data space allocated on heap
- Examples: string, arrays.

DATA TYPES

C# Type Alias	CLS Type	Size (bits)	Suffix	Description	Range
sbyte	SByte	8		signed byte	–128 to 127
byte	Byte	8		unsigned byte	0 to 255
short	Int16	16		short integer	-32,768 to 32,767
ushort	UInt16	16		unsigned short integer	0 to 65535
int	Int32	32		integer	-2,147,483,648 to 2,147,483,647
uint	UInt32	32	u	unsigned integer	0 to 4,294,967,295
long	Int64	64	L	long integer	–9,223,372,036,854,775,808 to 9,223,372,036,854,775,807
ulong	UInt64	64		unsigned long integer	0 to 18,446,744,073,709,551,615
char	Char	16		Unicode character	any valid character (e.g., 'a' , '*', '\x0058' [hexadecimal], or '\u0058' [Unicode])
float	Single	32	F	floating-point number	$\pm 1.5 \times 10^{-45}$ to $\pm 3.4 \times 10^{38}$
double	Double	64	d	double floating-point number	Range $\pm 5.0 \times 10^{-324}$ to $\pm 1.7 \times 10^{308}$
bool	Boolean	1		logical true/false value	true/false
decimal	Decimal	128	m	used for financial and monetary calculations	from approximately 1.0×10^{-28} to 7.9×10^{28} with 28 to 29 significant digits
bool	Boolean	true or false		used to represent true or false values	

ARITHMETIC EXPRESSIONS

ORDER OF PRECEDENCE

- Increment and decrement
- Positive and negative
- Multiplication, division, and remainder
- Addition and subtraction

ARITHMETIC OPERATORS

Operator	Name
+	Addition
-	Subtraction
*	Multiplication
/	Division
%	Modulus
++	Increment
--	Decrement
+	Positive sign
-	Negative sign

ASSIGNMENT STATEMENTS

An assignment statement consists of a variable, an equals sign, and an expression. When the assignment statement is executed, the value of the expression is determined and the result is stored in the variable.

ASSIGNMENT OPERATORS

Operator	Name
=	Assignment
+=	Addition
-=	Subtraction
*=	Multiplication
/=	Division
%=	Modulus

CASTING

IMPLICIT CASTING

Assigning a less precise data type to a more precise data type will cause C# to automatically convert the less precise data type to the more precise data type. This is also called a widening conversion.

CASTING

EXPLICIT CASTING

We can code an assignment statement that assigns a more precise data type to a less precise data type. In this case, we must use parentheses to specify the less precise data type. This is also called a narrowing conversion.

RECAP

WHAT YOU SHOULD KNOW AT THIS POINT:

- C# Importance and History
- How C# is compared to other programming languages.
- Types of applications developed using C#.
- Know common C# IDEs
- How to write a Hello World program using C#.
- Data types, statements, and variables in C#.