10/2/2019

.NET

When you compile you are putting into a common intermediate languages (.dll)

Common Language runtime is used to convert (.dll)

Mono was the port from .NET Framework to linux

Unity – Game engine

.NET Standard – interface that can be used in both Framework and Core CLR environments

CIL –Common Intermediate Language(.dll libraries)

JIT – Just in time compiler, compiles (.dll) into machine code to be run by cpu and provides multithreading, cross-platform, exceptions, garbage collection/memory management, inheritance relationships between classes and is interoperable between source language(can have different languages working together)

CLI – Common Language infrastructure is specification you need to follow to write your own .net infrastructure

VES - Virtual Execution system, like CLR. Needs JIT and compiler to IL

BCL – Base Class Library

CTS – Common type system

Managed Code – Code that runs in the CLR(.NET’s Framework version of VES)

Benefits:

* Multi-threading
* X-platform
* Exceptions
* Garbage collection/memory management
* Inheritance, OOP
* Interoperable

Reference Types

* Any instance of a class
* Like a pointer but you can’t get the address(physical data)

Value Types

* any numeric type(int, bool, double, char) <=structs

In C# you can make your own reference type(struct dog instead of class dog)

Object

Classes

List, Task, Exception, String

System.ValueType => reference type behavior

All value types derive from this => value type behavior(int, double, user defined structs, enums)

Unified Type system so everything is an object and prevents having to box and unbox

But if you do box in c#, an int that is called an object and treats it like a reference type

Int a = 3; =>boxing

Object o = a;

Int b = (int)o; => unboxing

Git Branching

Git pull is 2 part command

1. git fetch(update all remote tracking branches AKA origin/anything)
2. git merge(it merges the origin version of the branch into your version of the brach)

git push --set-upstream origin colton

Clone that repo from our batch GitHub org.  
Make a new branch named with your first name, all lowercase, without accents.  
   To do this, we use git checkout -b nick to create a new branch called  
       nick starting at the current commit.  
   We can create the branch without switching to it by running git branch nick.  
   We can switch branches freely (if there are no un-committed changes) with  
       git checkout nick, git checkout master, etc.  
Push your code to that branch on GitHub.  
   git push already operates on just the current branch.Remember to add a proper gitignore first.  
Using the dotnet CLI, make a new project named FizzBuzz in a folder named FizzBuzz.  
Write your code in the Program.cs class file.  
(So the path will be: <repo>/FizzBuzz/Program.cs).

Applications that you run HAS to target a .NET Core specific API

Libraries can use .NET standard to be later used for various API

From cmd line – how to make folder directory

1. mkdir ShapesApp
2. cd ShapesApp/
3. dotnet new sln (solution)
4. dotnet sln add ShapesApp.App/
5. dotnet new classlib –o ShapesApp.Library
6. in the library lets make a new file

cd ShapesApp/ShapesApp.App

dotnet add reference ../ShapesApp.Library/

Out parameters cannot have a variable before you pass it, the method gets that exact variable and fills in its value

Extension Method – just a simple syntax(as though I added a method to Rectangle but not really)

Interfaces – type that does not have implementation of it’s methods.

* Can put methods and properties. A contract that classes can declare themselves as following

Internal cant be seen by other projects

Virtual methods can be overridden So use override before method name and virtual on base classes method

Non virtual means it cannot be overridden, you would make a method virtual if you plan on overriding it

Method hiding is possible on non virtual methods but need to use new modifier before method name

Static Class - cannot be made into an object Cannot do inheritance with static classes.

Static methods can only see other static members within their class.

Static members create only 1 instance of themselves and not copies.

Immutable - Object whose data doesn’t change

Const – has to be assigned inline

Sealed – preventing any inheritance

Testing

1. Automated
2. Unit-Testing
   1. Syntax?
   2. What is it?
   3. Why do we need it? -
   4. Technogy? xUnit, MSTest, NUnit

Constructors are not inherited, every subclass constructor calls some parent class constructor first