

Caroline Tries Game Dev

w Unity!!!

CS Capture

November +

December

Learning Notes ↗

# C# BASICS

## C#

- files end in .cs
- 5th most popular language 2019
- part of .NET platform - Microsoft cross-platform developer platform
- must import w/ using System;

## OUTPUT/INPUT

```
Console.WriteLine("Hi");  
String input = Console.ReadLine();
```

## COMMENTS

```
//single line  
/* multi line */
```

## VARIABLES

- int, double, char, bool, string datatypes
- all variables must be assigned type at declaration

## STRING FUNCTIONS

- ToUpper()
- ToLower()
- Substring()
- Length
- "StringEventArgs"

## CONDITIONALS

```
if (condition)  
{  
    //code  
}  
else if ()  
{  
    //code  
}  
else  
{  
}
```

← typical C#  
syntax note  
braces on new  
own lines

switch

switch ( varName )

{

case "one":

// code

break; ← mandatory break required

case "two":

// code

break;

default:

break;

}

ternary operators

String varName = "one" ? <condition> : "two";

↑  
if

↑  
else

METHODS

static <returnType> <name> (<pType parameter>) {

// code

}

expression based methods

static retType Name () => i++;

↑ ↑ no return necessary  
for 1 line functions

default parameters

```
statele int Myfunc(int p1, p2=2, p3=4) {
```

↑  
mandatory parameters come first

out parameters

- take variables as arguments, sets parameters as arrays  
actions performed on out parameters reflect on actual variable outside method

```
statele void Myfunc(out string name, out string color) {
```

name = "C";

color = "purple";

}

string name;

string color;

Myfunc(out name, out color);

(Console.WriteLine(name, color)) → // C, purple

ARRAYS

```
type[] name = { };
```

```
type[] name = new type[size];
```

- Arrays are mutable : size can change

## loops

### for loop

for (int i = 0; i < 10; i++)  
    {  
        Console.WriteLine(i);  
    }

increasement  
semicolon!

exclusive

### foreach loop

foreach ( type itemname in arrayName ) { }

### while loop

while ( var != "value" ) { }

### do while loop

• always runs at least once

do {

// code

} while ( condition ); t semicolon var

### jump statements

• break; continue, return, as usual

LWSE

cross cat

4

public string name

{ get; set; }

public int age

1

get { return age; }

Set {name} = value;

3

```
public Cat (String name, int age)
```

6

two.none = none

two. age = age

3

state (at 0) — state consumers on  
one, when first makes  
(more wrinkles ("first Cat instance created")); created

3

3

```
Cart cart = new Cart("FIFO", 1);
```

## inheritance

class Animal

{

public **virtual** void Noise();

{

c. w. ('noise sound');

}

{

class Cat : Animal

{

public **override** void Noise();

{

Console.WriteLine ("meow");

{

{

• public accessible anywhere

• private only class

• protected only class : classes that inherit from it

must be abstract  
or virtual to be  
overridden

## INTERFACES

- live mandatory blueprints for classes

interface **IAnimal** with I  
name starts with I

{

```
string name { get; }  
int age { get; set; }  
void Noise();
```

}

pure class (at : **IAnimal**) { }

\* inherit live supertypes

all classes using interface MUST have these properties or methods

## LISTS

- host any type, size can change using `System.Collections.Generic`

```
list<T> stuff = new list<T>();
```

- `Count` property size of list
- `Contains("")` true/false
- `Remove(value)`
- `Clear()` remove all values
- `Add(value)` adds to end

# UNITY w/ C#

## IMPORTS

at the top of every file

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using Unity.VismentScripting;
using UnityEngine;
```

using UnityEngine; or only for working w/ ui elements

## STRUCTURE

scripts attach to  
individual objects

```
private class ScriptName : MonoBehaviour
```

```
{
```

// define global vars here

```
void Start()
```

```
{
```

// called before 1st frame update

```
}
```

```
void Update()
```

```
{
```

// called once per frame

```
}
```

// define other methods below

```
3
```

## Scene Management

using `UnityEngine.SceneManagement`; or required import

`SceneManager.LoadScene (SceneManager.GetActiveScene ().buildIndex + 1);`  
advance to the next scene using build index

## Notable methods

`void OnTriggerEnter2D (Collider2D collision)`

`void OnCollisionEnter2D (Collider2D collision)`

`Input.GetButtonDown ("Space")` or key press

`Input.GetAxisRaw ("Vertical")` or mouse  
in direction

## Notable components

Animator

AudioSource

Layman

Rigidbody2D

BoxCollider2D

Sprk Renderer

myComponent = GetComponent < type > ();

## Serialize field

- more available from editor while keeping it private  
[SerializeField] private string varName;
- easier for quick assignment/overwriting

# Hotkeys : Shortcuts

The screenshot shows the Unity Cheat Keys interface. At the top, it says "CHEATKEYS" and "Licensed to John Smith | v1.0.0 beta". The interface is divided into several sections:

- Tools**
  - Q Pan
  - W Move
  - E Rotate
  - R Scale
  - T Rect
  - Y Unified
  - Z Pivot Mode Toggle
  - X Pivot Rotation Toggle
  - Ctrl+LMB** Snap (move/rotate/scale)
  - V Vertex Snap (move/rotate/scale)
- GameObject**
  - Ctrl+Shift+N New Empty GO
  - Alt+Shift+N New Empty Child GO
  - Ctrl+Shift+A Add Component
  - Ctrl+Alt+F Move to View
  - Ctrl+Shift+F Align with View
  - Shift+F** Lock View to Selected
- Scene View**
  - Alt+LMB Orbit
  - Alt+MMB Drag
  - Alt+RMB Zoom
  - Alt+Scroll Zoom to Mouse Cursor
  - F Frame Selected
  - F+F Follow Frame
  - Shift+F Lock View to Selected
  - Ctrl+D Duplicate
  - Ctrl+R Refresh Scene
  - Ctrl+Alt+# Save Selection Set
  - Ctrl+Shift+# Load Selection Set
- Windows**
  - Ctrl+Tab Next Window
  - Ctrl+Shift+Tab Previous Window
  - Ctrl+1 Scene
  - Ctrl+2 Game
  - Ctrl+3 Inspector
  - Ctrl+4 Hierarchy
  - Ctrl+5 Project**
  - Ctrl+6 Animation
  - Ctrl+7 Profiler
  - Ctrl+8 Audio Mixer
  - Ctrl+0 Asset Store
  - Ctrl+Shift+C Console
- File Options**
  - Ctrl+N New Scene
  - Ctrl+O Open Scene
  - Ctrl+S Save Scene
  - Ctrl+Shift+S Save Scene As
  - Ctrl+Shift+B Build Settings**
  - Ctrl+B Build & Run
- Play Mode Controls**
  - Ctrl+P Play Toggle
  - Ctrl+Shift+P Pause
  - Ctrl+Alt+P Step
- Animation Tool**
  - Period Next Frame
  - Alt+Period Next Keyframe
  - Comma Previous Frame
  - Alt+Comma Previous Keyframe
  - K Record Keyframe
  - Space Play Animation

1,222 x 601

L rotate tile right  
J rotate tile left