Week 2

# Class 1 Monday August 26th - Classes Method’s & Namespaces

## NAMESPACES

* Namespace are like chunks of code… Name space generally follows the folder structure of your application
* Namespace should be named the same as the folder it’s in
* Dashes, spaces and underscore are a no go.
* Name folders like ThisIsMyFolder

## CLASSES

* Classes are like templates, or a Blueprint for the thing you actually want to use later. Almost like a function You create a class file, and then use it later
* You can technically have more then one class per c# file, but generally you want to keep 1 class per file
* Name of file === name of class
* Classes are similar to MODUALS
* PRIVATE means only accessible inside of this class

##### Properties

* Properties Start with a Capital
* Get/set are just for properties

public class Name

{

// get allows you to get/retrieve/use the value of FullName

// set allows you to set/change the value of FullName

// You don't need both, but you need at least one

public String Fullname { get; set; }

* }
* If you want to make it accessible outside of this class then user PUBLIC
* If a property has only a get; then the only place it can be set is in the constructor,

##### Fields

* Should generally Keep Fields Private
* When naming a field always start with an underscore followed by camel case i.e.
* int \_thisIsAField;
* string \_name;
* public string Name{ get {return \_name;} set { \_name = vaue;} }

## METHODS

* All methods are Functions, but not all Functions are Methods
* Constructor vs Method
* A constructor can require you to use something to make a “new” class
* You use Constructors to predetermine states of the class
* i.e. public Person(string Name, int Age) { Name = “Keith” }
* When calling a new Person… Name would automatically be Keith, and you would have to assign an age

## Other

* Fields are usual settable, and gettable
* Classes are made up of Properties, Fields
* Properties are a concept
* Class=> Console.Writeline(); <- Method on the class
* private string FullName { get; set; } //This is a Property... an Auto Property
* Static // Void
* Void means you don’t return a value
* Static means

### Object Initializers

* Var name = new Name();
* Name.FullName = “Keith”
* Var name = new Name { FullName = input }
* Allows you to set the property of a class when you initialize it
* Internal vs private/public
* All classes by default are internal

# Class 2 Tuesday August 27th

* Boxing

1. Properties
2. Fields
3. Lists
4. methods

## Methods

* const
  + Cannot be changed, and cannot be something that comes from outside the class non deterministic.
* readonly
  + you can set once in constructor, and then never again
* Value Types
  + HAVE DEFAULT VALUE
  + Int-0
* Reference Types
  + DO NOT HAVE A DEFAULT VALUE
  + Default value = null