C# Fundamentals for Absolute Beginners

Microsoft Virtual Academy Course by Bob Tabor

Creating Your First C# Program 1

Variables 2

Decisions 3

Operators, Expressions & Statements 4-5

For Iteration 6

Understanding Arrays 7-8

Simple Methods and Helper Methods 9-10

While Iteration 11-12

Working with Strings 13

Dates and Times 14

Simple Classes 15

Object Life Time 16-17

Understanding Scope 18

Assemblies and Namespaces 19

Working with Collections 20-21

Understanding LINQ 22-23

Enums and Switch 24-25

Handling Exceptions 26

Event Driven Programming (Timer Example) 27

Certificate of Completion 28

Creating Your First C# Program

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace HelloWorld

{

class Program

{

static void Main(string[] args)

{

Console.WriteLine("Hello World");

Console.ReadLine();

}

}

}

Variables

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Variables

{

class Program

{

static void Main(string[] args)

{

/\*

int x;

int y;

x = 7;

y = x + 3;

Console.WriteLine(y);

Console.ReadLine();

\*/

Console.WriteLine("what is your name?");

Console.Write("Type your first name: ");

string myFirstName;

myFirstName = Console.ReadLine();

//string myLastName;

//Console.Write("Type your last name: ");

//myLastName = Console.ReadLine();

Console.WriteLine("Type your last name: ");

string myLastName = Console.ReadLine();

Console.WriteLine("Hello, " + myFirstName + " " + myLastName);

Console.ReadLine();

}

}

}

Decisions

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Decisions

{

class Program

{

static void Main(string[] args)

{

/\*

Console.WriteLine("Bob's Big Giveaway");

Console.Write("Choose a door: 1, 2 or 3: ");

string userValue = Console.ReadLine();

string message = "";

if (userValue == "1")

message = "You won a new car!";

else if(userValue == "2")

message = "You won a new boat!";

else if(userValue =="3")

message = "You won a new cat!";

else

{message = "Sorry, we didn't understand. ";

//message = message + "You Lose.";

message += "You Lose.";

}

Console.WriteLine(message);

Console.ReadLine();

\*/

Console.WriteLine("Bob's Big Giveaway");

Console.Write("Choose a door: 1, 2 or 3: ");

string userValue = Console.ReadLine();

string message = (userValue == "1") ? "boat" : "strand of lint";

//Console.Write("You won a ");

//Console.Write(message);

//Console.Write(".");

//Console.WriteLine("You won a {0}.", message);

Console.WriteLine("You entered: {0}, therefore you won a {1}.", userValue, message);

Console.ReadLine();

}

}

}

Operators, Expressions & Statements

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace OperatorsExpressionsStatements

{

class Program

{

static void Main(string[] args)

{

// Variable declaration

int x, y, a, b;

// Assignment operator

x = 3;

y = 2;

a = 1;

b = 0;

// There are many mathematical operators ...

// Addition operator

x = 3 + 4;

// Subtraction operator

x = 4 - 3;

// Multiplication operator

x = 10 \* 5;

// Division operator

x = 10 / 5;

// Order of operations using parenthesis

x = (x + y) \* (a - b);

// There are many operators used to evaluate values ...

// Equality operator

if (x == y)

{

}

// Greater than operator

if (x > y)

{

}

// Less than operator

if (x < y)

{

}

// Greater or equal to operator

if (x >= y)

{

}

// Less than or equal to operator

if (x <= y)

{

}

// There are two "conditional" operators as well that can

// be used to expand / enhance an evaluation ...

// ... and they can be combined together multiple times.

// Conditional AND operator …

if ((x > y) && (a > b))

{

}

// Conditional OR operator …

if ((x > y) || (a > b))

{

}

// Also, here's the in-line conditional operator we

// learned about in the previous lesson ...

string message = (x == 1) ? "Car" : "Boat";

// Member access and Method invocation

Console.WriteLine("Hi");

//x + y;

}

}

}

For Iteration

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ForIteration

{

class Program

{

static void Main(string[] args)

{

for (int i = 0; i < 10; i++)

{

//Console.WriteLine(i);

if (i == 7)

{

Console.WriteLine("Found seven!");

break;

}

}

for (int myValue = 0; myValue < 12; myValue++)

{

Console.WriteLine(myValue);

}

Console.ReadLine();

}

}

}

Understanding Arrays

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace UnderstandingArrays

{

class Program

{

static void Main(string[] args)

{

/\*

int number1 = 4;

int number2 = 8;

int number3 = 15;

int number4 = 16;

int number5 = 23;

if (number1 == 16)

{

}

else if (number2 == 16)

{

}

else if (number3 == 16)

{

}

\*/

/\*

int[] numbers = new int[5];

numbers[0] = 4;

numbers[1] = 8;

numbers[2] = 15;

numbers[3] = 16;

numbers[4] = 23;

//numbers[5] = 42;

//Console.WriteLine(numbers[1]);

Console.WriteLine(numbers.Length);

Console.ReadLine();

\*/

//int[] numbers = new int[] { 4, 8, 15, 16, 23, 42 };

string[] names = new string[] { "Eddie", "Alex", "Michael", "David Lee" };

/\*

for (int i = 0; i < names.Length; i++)

{

Console.WriteLine(names[i]);

}

Console.ReadLine();

\*/

/\*foreach (string name in names)

{

Console.WriteLine(name);

}

Console.ReadLine();

\*/

string zig = "You can get what you want out of life " +

" if you help enough people get what they want.";

char[] charArray = zig.ToCharArray();

Array.Reverse(charArray);

foreach (char zigChar in charArray)

{

Console.Write(zigChar);

}

Console.ReadLine();

}

}

}

Simple Methods and Helper Methods

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace SimpleMethod

{

class Program

{

static void Main(string[] args)

{

HelloWorld();

Console.ReadLine();

}

private static void HelloWorld()

{

Console.WriteLine("Hello world!");

}

}

}

namespace HelperMethods

{

class Program

{

static void Main(string[] args)

{

Console.WriteLine("The Name Game");

Console.Write("What's your first name? ");

string firstName = Console.ReadLine();

Console.Write("What's your last name? ");

string lastName = Console.ReadLine();

Console.Write("In what city were you born? ");

string city = Console.ReadLine();

DisplayResult(ReverseString(firstName),

ReverseString(lastName),

ReverseString(city));

Console.ReadLine();

}

private static string ReverseString(string message)

{

char[] messageArray = message.ToCharArray();

Array.Reverse(messageArray);

return String.Concat(messageArray);

}

private static void DisplayResult(

string reversedFirstName,

string reversedLastName,

string reversedCity)

{

Console.WriteLine("Results: ");

Console.Write(String.Format("{0} {1} {2}",

reversedFirstName,

reversedLastName,

reversedCity));

}

private static void DisplayResult(string message)

{

Console.WriteLine("Results: ");

Console.Write(message);

}

}

}

While Iteration

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace WhileIteration

{

class Program

{

static void Main(string[] args)

{

bool displayMenu = true;

while (displayMenu == true)

{

displayMenu = MainMenu();

}

//MainMenu();

}

private static bool MainMenu()

{

Console.Clear();

Console.WriteLine("Choose an option:");

Console.WriteLine("1) Print Numbers");

Console.WriteLine("2) Guessing Game");

Console.WriteLine("3) Exit");

string result = Console.ReadLine();

if (result == "1")

{

PrintNumbers();

return true;

}

else if (result == "2")

{

GuessingGame();

return true;

}

else if (result == "3")

{

return false;

}

else

{

return true;

}

}

private static void PrintNumbers()

{

Console.Clear();

Console.WriteLine("Print numbers!");

Console.Write("Type a number: ");

int result = int.Parse(Console.ReadLine());

int counter = 1;

while (counter < result + 1)

{

Console.Write(counter);

Console.Write("-");

counter++;

}

Console.ReadLine();

}

private static void GuessingGame()

{

Console.Clear();

Console.WriteLine("Guessing game!");

Random myRandom = new Random();

int randomNumber = myRandom.Next(1, 11);

int guesses = 0;

bool incorrect = true;

do

{

Console.WriteLine("Guess a number between 1 and 10: ");

string result = Console.ReadLine();

guesses++;

if (result == randomNumber.ToString())

incorrect = false;

else

Console.WriteLine("Wrong!");

} while (incorrect);

Console.WriteLine("Correct! It took you {0} guesses.", guesses);

Console.ReadLine();

}

}

}

Working with Strings

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace WorkingWithStrings

{

class Program

{

static void Main(string[] args)

{

//string myString = "My \"so-called\" life";

//string myString = "What if I need a \nnew line?";

//string myString = "Go to your c:\\ drive";

//or

//string myString = @"Go to your c:\ drive";

//string myString = String.Format("{0} = {1}", "First", "Second");

//string myString = string.Format("{0:C}", 123.45);

//string myString = string.Format("{0:N}", 1234567890);

//string myString = string.Format("Percentage: {0:P}", .123);

//string myString = string.Format("Phone Number: {0:(###) ###-####}", 1234567890);

//string myString = " That summer we took threes across the board ";

//myString = myString.Substring(6, 14);

//myString = myString.ToUpper();

//myString = myString.Replace(" ", "--");

//myString = myString.Remove(6, 14);

/\*

myString = String.Format("Length before: {0} -- Length after: {1}",

myString.Length,

myString.Trim().Length);

\*/

/\*

string myString = "";

for (int i = 0; i < 100; i++)

{

myString += "--" + i.ToString();

}

\*/

StringBuilder myString = new StringBuilder();

for (int i = 0; i < 100; i++)

{

myString.Append("--");

myString.Append(i);

}

Console.WriteLine(myString);

Console.ReadLine();

}

}

}

Dates and Times

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace DatesAndTimes

{

class Program

{

static void Main(string[] args)

{

DateTime myValue = DateTime.Now;

//Console.WriteLine(myValue.ToString());

//Console.WriteLine(myValue.ToShortDateString());

//Console.WriteLine(myValue.ToShortTimeString());

//Console.WriteLine(myValue.ToLongDateString());

//Console.WriteLine(myValue.ToLongTimeString());

//Console.WriteLine(myValue.AddDays(3).ToLongDateString());

//Console.WriteLine(myValue.AddHours(3).ToLongTimeString());

//Console.WriteLine(myValue.AddDays(-3).ToLongDateString());

//Console.WriteLine(myValue.Month);

//DateTime myBirthday = new DateTime(1987, 05, 25);

//Console.WriteLine(myBirthday.ToShortDateString());

DateTime myBirthday = DateTime.Parse("25/05/1987");

TimeSpan myAge = DateTime.Now.Subtract(myBirthday);

Console.WriteLine(myAge.TotalDays);

Console.ReadLine();

}

}

}

Simple Classes

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace SimpleClasses

{

class Program

{

static void Main(string[] args)

{

Car myCar = new Car();

myCar.Make = "Vauxwagon";

myCar.Model = "Golf";

myCar.Year = 2007;

myCar.Color = "Black";

Console.WriteLine("{0} {1} {2} {3}",

myCar.Make,

myCar.Model,

myCar.Year,

myCar.Color);

//decimal value = DetermineMarketValue(myCar);

//Console.WriteLine("{0:C}", value);

Console.WriteLine("{0:C}", myCar.DetermineMarketValue());

Console.ReadLine();

}

private static decimal DetermineMarketValue(Car car)

{

decimal carValue = 100.0M;

//Someday I might look up the car

//online to get a more accurate value

return carValue;

}

}

class Car

{

public string Make { get; set; }

public string Model { get; set; }

public int Year { get; set; }

public string Color { get; set; }

public decimal DetermineMarketValue()

{

decimal carValue;

if (Year > 2008)

carValue = 10000;

else

carValue = 2000;

return carValue;

}

}

}

Object Life Time

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ObjectLifeTime

{

class Program

{

static void Main(string[] args)

{

Car myCar = new Car();

Car.MyMethod();

/\*

myCar.Make = "Vauxwagon";

myCar.Model = "Golf";

myCar.Year = 2007;

myCar.Color = "Silver";

\*/

//Car myThirdCar = new Car("Vauxhall", "Corsa", "2000", "Silver");

/\*

Car myOtherCar;

myOtherCar = myCar;

Console.WriteLine("{0} {1} {2} {3}",

myOtherCar.Make,

myOtherCar.Model,

myOtherCar.Year,

myOtherCar.Color);

myOtherCar.Model = "Polo";

Console.WriteLine("{0} {1} {2} {3}",

myOtherCar.Make,

myOtherCar.Model,

myOtherCar.Year,

myOtherCar.Color);

myOtherCar = null;

Console.WriteLine("{0} {1} {2} {3}",

myOtherCar.Make,

myOtherCar.Model,

myOtherCar.Year,

myOtherCar.Color);

myCar = null;

\*/

Console.ReadLine();

}

}

class Car

{

public string Make { get; set; }

public string Model { get; set; }

public int Year { get; set; }

public string Color { get; set; }

/\*public Car()

{

// You could load from a configuration file,

// a database, etc.

Make = "Mazda";

}

public Car(string make, string model, int year, string color)

{

Make = make;

Model = model;

Year = year;

Color = color;

}\*/

public static void MyMethod()

{

Console.WriteLine("Called the static MyMethod");

}

}

}

Understanding Scope

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace UnderstandingScope

{

class Program

{

private static string k = "";

static void Main(string[] args)

{

string j = "";

for (int i = 0; i < 10; i++)

{

j = i.ToString();

k = i.ToString();

Console.WriteLine(i);

if (i == 9)

{

string l = i.ToString();

}

//Console.WriteLine(l);

}

//Console.WriteLine(i);

Console.WriteLine("Outside of the for: " + j);

Console.WriteLine("Outside of the for: " + k);

HelperMethod();

Car myCar = new Car();

myCar.DoSomething();

Console.ReadLine();

}

static void HelperMethod()

{

Console.WriteLine("Value of k from the HelperMethod(): " + k);

}

}

class Car

{

public void DoSomething()

{

Console.WriteLine(helperMethod());

}

private string helperMethod()

{

return "Hello World!";

}

}

}

Assemblies and Namespaces

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.IO;

using System.Net;

namespace AssembliesAndNamespaces

{

class Program

{

static void Main(string[] args)

{

WebClient client = new WebClient();

string reply = client.DownloadString("http://microsoft.com");

Console.WriteLine(reply);

//File.WriteAllText(@"C:\Lesson17\WriteText.txt", reply);

Console.ReadLine();

}

}

}

Working with Collections

using System;

using System.Collections;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace WorkingWithCollections

{

class Program

{

static void Main(string[] args)

{

/\*

Car car1 = new Car();

car1.Make = "Vauxwagon";

car1.Model = "Golf";

car1.VIN = "A1";

Car car2 = new Car();

car2.Make = "Vauxhall";

car2.Model = "Corsa";

car2.VIN = "B2";

Book b1 = new Book();

b1.Author = "Stuart Bird";

b1.Title = "Microsoft .NET XML Web Services";

b1.ISBN = "0-000-00000-0";

\*/

/\*

//ArrayLists are dynamically sized,

//cool features sorting, remove items

ArrayList myArrayList = new ArrayList();

myArrayList.Add(car1);

myArrayList.Add(car2);

myArrayList.Add(b1);

myArrayList.Remove(b1);

foreach (Car car in myArrayList)

{

Console.WriteLine(car.Make);

}

\*/

/\*

//List<T>

List<Car> myList = new List<Car>();

myList.Add(car1);

myList.Add(car2);

//myList.Add(b1);

foreach (Car car in myList)

{

Console.WriteLine(car.Model);

}

\*/

/\*

//Dictionary<TKey, TValue>

Dictionary<string, Car> myDictionary

= new Dictionary<string, Car>();

myDictionary.Add(car1.VIN, car1);

myDictionary.Add(car2.VIN, car2);

Console.WriteLine(myDictionary["B2"].Make);

\*/

//string[] names = { "Bob", "Steve", "Brian", "Chuck"};

//Object initialiser

//No need for a constructor

//Car car1 = new Car() { Make = "BMW", Model = "M3", VIN = "C3" };

//Car car2 = new Car() { Make = "Mercades", Model = "A series", VIN = "D4" };

//Collection initialiser

List<Car> myList = new List<Car>()

{

new Car{ Make = "Jaguar", Model = "E type", VIN = "E5" },

new Car{ Make = "Rolls Royce", Model = "Phantom", VIN = "F6" }

};

Console.ReadLine();

}

}

class Car

{

public string VIN { get; set; }

public string Make { get; set; }

public string Model { get; set; }

}

class Book

{

public string Title { get; set; }

public string Author { get; set; }

public string ISBN { get; set; }

}

}

Understanding LINQ

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace UnderstandingLINQ

{

class Program

{

static void Main(string[] args)

{

//LINQ = LANGUAGE INTEGRATED QUERY syntax

List<Car> myCars = new List<Car>() {

new Car() { VIN="A1", Make = "BMW", Model= "550i", StickerPrice=55000, Year=2009},

new Car() { VIN="B2", Make="Toyota", Model="4Runner", StickerPrice=35000, Year=2010},

new Car() { VIN="C3", Make="BMW", Model = "745li", StickerPrice=75000, Year=2008},

new Car() { VIN="D4", Make="Ford", Model="Escape", StickerPrice=25000, Year=2008},

new Car() { VIN="E5", Make="BMW", Model="55i", StickerPrice=57000, Year=2010}

};

// LINQ query

/\*

var bmws = from car in myCars

where car.Make == "BMW"

&& car.Year == 2010

select car;

\*/

/\*

var orderedCars = from car in myCars

orderby car.Year descending

select car;

\*/

// LINQ method

//var bmws = myCars.Where(p => p.Make == "BMW" && p.Year == 2010);

//var orderedCars = myCars.OrderByDescending(p => p.Year);

/\*

var firstBMW = myCars.OrderByDescending(p => p.Year).First(p => p.Make == "BMW");

Console.WriteLine(firstBMW.VIN);

\*/

//Console.WriteLine(myCars.TrueForAll(p => p.Year > 2007));

//myCars.ForEach(p => p.StickerPrice -= 3000);

//myCars.ForEach(p => Console.WriteLine("{0} {1:C}", p.VIN, p.StickerPrice));

//Console.WriteLine(myCars.Exists(p => p.Model == "745li"));

//Console.WriteLine(myCars.Sum(p => p.StickerPrice));

/\*

foreach (var car in orderedCars)

{

Console.WriteLine("{0} {1}", car.Year, car.Model, car.VIN);

}

\*/

Console.WriteLine(myCars.GetType());

var orderedCars = myCars.OrderByDescending(p => p.Year);

Console.WriteLine(orderedCars.GetType());

var bmws = myCars.Where(p => p.Make == "BMW" && p.Year == 2010);

Console.WriteLine(bmws.GetType());

var newCars = from car in myCars

where car.Make == "BMW"

&& car.Year == 2010

select new { car.Make, car.Model };

Console.WriteLine(newCars.GetType());

Console.ReadLine();

}

}

class Car

{

public string VIN { get; set; }

public string Make { get; set; }

public string Model { get; set; }

public int Year { get; set; }

public double StickerPrice { get; set; }

}

}

Enums and Switch

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace EnumsAndSwitch

{

class Program

{

static void Main(string[] args)

{

List<Todo> todos = new List<Todo>()

{

new Todo { Description = "Task 1", EstimatedHours = 6, Status = Status.Completed },

new Todo { Description = "Task 2", EstimatedHours = 2, Status = Status.InProgress },

new Todo { Description = "Task 3", EstimatedHours = 8, Status = Status.NotStarted },

new Todo { Description = "Task 4", EstimatedHours = 12, Status = Status.Deleted },

new Todo { Description = "Task 5", EstimatedHours = 6, Status = Status.InProgress },

new Todo { Description = "Task 6", EstimatedHours = 2, Status = Status.NotStarted },

new Todo { Description = "Task 7", EstimatedHours = 14, Status = Status.NotStarted },

new Todo { Description = "Task 8", EstimatedHours = 8, Status = Status.Completed },

new Todo { Description = "Task 9", EstimatedHours = 8, Status = Status.InProgress },

new Todo { Description = "Task 10", EstimatedHours = 8, Status = Status.Completed },

new Todo { Description = "Task 11", EstimatedHours = 4, Status = Status.NotStarted },

new Todo { Description = "Task 12", EstimatedHours = 10, Status = Status.Completed },

new Todo { Description = "Task 13", EstimatedHours = 12, Status = Status.Deleted },

new Todo { Description = "Task 14", EstimatedHours = 6, Status = Status.Completed }

};

Console.ForegroundColor = ConsoleColor.Blue;

PrintAssessment(todos);

Console.ReadLine();

}

private static void PrintAssessment(List<Todo> todos)

{

foreach (var todo in todos)

{

{

}

}

}

}

class Todo

{

public string Description { get; set; }

public int EstimatedHours { get; set; }

public Status Status { get; set; }

}

enum Status

{

NotStarted,

InProgress,

OnHold,

Completed,

Deleted

}

}

Handling Exceptions

using System;

using System.Collections.Generic;

using System.IO;

using System.Linq;

using System.Net;

using System.Text;

using System.Threading.Tasks;

namespace HandlingExceptions

{

class Program

{

static void Main(string[] args)

{

try

{

string content = File.ReadAllText(@"C:\Lesson22\Exampl.txt");

Console.WriteLine(content);

//string content = File.ReadAllText(@"C:\Users\Stuart\Documents\Visual Studio 2017\Projects\HandlingExceptions");

//Console.WriteLine(content);

//Above gives access denied result

}

catch (FileNotFoundException ex)

{

Console.WriteLine("There was a problem!");

Console.WriteLine("Make sure the name of the file is named correctly: Exampl.txt");

}

catch (DirectoryNotFoundException ex)

{

Console.WriteLine("There was a problem!");

Console.WriteLine(@"Make sure the directory C:\Lesson22 exists.");

}

catch (Exception ex)

{

Console.WriteLine("There was a problem!");

Console.WriteLine(ex.Message);

}

finally

{

// Code to finalise, Setting objects to null

// Closing database connections

Console.WriteLine("Closing application now ...");

}

Console.ReadLine();

}

}

}

Event Driven Programming (Timer Example)

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Timers;

namespace TimerExample

{

class Program

{

static void Main(string[] args)

{

Timer myTimer = new Timer(2000);

myTimer.Elapsed += MyTimer\_Elapsed;

myTimer.Elapsed += MyTimer\_Elapsed1;

myTimer.Start();

Console.WriteLine("Press enter to remove the red event.");

Console.ReadLine();

myTimer.Elapsed -= MyTimer\_Elapsed1;

Console.ReadLine();

}

private static void MyTimer\_Elapsed1(object sender, ElapsedEventArgs e)

{

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine("Elapsed1: {0:HH:mm:ss.fff}", e.SignalTime);

}

private static void MyTimer\_Elapsed(object sender, ElapsedEventArgs e)

{

Console.ForegroundColor = ConsoleColor.White;

Console.WriteLine("Elapsed: {0:HH:mm:ss.fff}", e.SignalTime);

}

}

}

