/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*

\* Part 10: DelegatesAndEvents

\*

\* Topics: 1. Implement a custom EventArgs class.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

usingSystem**;**

usingSystem.Collections.Generic**;**

usingSystem.Text**;**

namespaceEventsDemo

**{**

// This class inherits from the base class EventArgs. This class will

// get the current system time and store it. If the event handler wishes

// to know the exact time the event fired, it can call the FireTime

// public property which will format the current time into readable text.

internalclassTimerEventArgs **:** EventArgs

**{**

privateDateTime\_currentTime**;**

internalTimerEventArgs**()**

**{**

\_currentTime=DateTime.Now**;**

**}**

internalstringAlarmTime

**{**

get

**{**

stringtime**;**

time=\_currentTime.Hour.ToString**(**"00"**)** +":"+

\_currentTime.Minute.ToString**(**"00"**)** +":"+

\_currentTime.Second.ToString**(**"00"**)** +"."+

\_currentTime.Millisecond.ToString**(**"000"**);**

returntime**;**

**}**

**}**

**}**

internalclassTimerClass

**{**

// This is the delegate that encapsulates the target methods

// in other objects. When an object wishes to subscribe to the

// event in this class, this delegate will be used to call the

// other objects' event handler methods when the event is raised.

internaldelegate

voidTimerExpiredHandler**(**objectsender**,** TimerEventArgse**);**

// This is the event (aka the multicast delegate) that will be

// raised. The event that can be subscribed to is OnAdvanceTimer.

internaleventTimerExpiredHandlerOnTimerExpired**;**

// This method will run until the stop flag is set to true.

internalvoidRun**(**intnumberOfSeconds**)**

**{**

intelapsedSeconds=0**;**

while **(**elapsedSeconds<numberOfSeconds**)**

**{**

// Sleep for a 10th of a second.

System.Threading.Thread.Sleep**(**1000**);**

// Increment the number of elapsed seconds;

elapsedSeconds++**;**

**}**

// Now the the timer has expired, it's time to

// raise the alarm.

RaiseAlarmEvent**();**

**}**

// Internal helper method that checks to see if any

// objects are subscribed to the event. If any are,

// their event handler method will be called through

// the multicast delegate.

privatevoidRaiseAlarmEvent**()**

**{**

// Make sure to check that there is at least one subscriber!

if **(**OnTimerExpired!=null**)**

**{**

// Note that before the event fires, an object of

// type TimerEventArgs is created. Then, when the

// event is fired, the TimerEventArgs object is

// passed along with the sender.

TimerEventArgst=newTimerEventArgs**();**

OnTimerExpired**(**this**,** t**);**

**}**

**}**

**}**

classEventTestClass

**{**

privatevoidDisplayAlarmMessage**(**objectsender**,** TimerEventArgse**)**

**{**

// Save the current color of the characters.

ConsoleColororiginalForeGroundColor=Console.ForegroundColor**;**

// Change the color of the characters to Red.

Console.ForegroundColor=ConsoleColor.Red**;**

for **(**inti=0**;** i<5**;** i++**)**

**{**

Console.WriteLine**(**"\n{0}: WAKE UP!!!!!!!!!!!"**,**

e.AlarmTime**);**

Console.Beep**(**5000**,** 500**);**

System.Threading.Thread.Sleep**(**1000**);**

**}**

// Reset the color of the characters to their original color.

Console.ForegroundColor=originalForeGroundColor**;**

**}**

staticvoidMain**()**

**{**

// Clear the console window.

Console.Clear**();**

// Get the number of seconds to wait.

Console.Write**(**"Number of seconds to put on the clock: "**);**

stringseconds=Console.ReadLine**();**

intnumberOfSeconds=0**;**

// Attempt to convert the string input into integer.

if **(**int.TryParse**(**seconds**,** outnumberOfSeconds**))**

**{**

// Create an instance of our first test class.

EventTestClassfirstTestClass=newEventTestClass**();**

// Create an instance of our second test class.

AnotherEventTestClasssecondTestClass=

newAnotherEventTestClass**();**

// Create an instance of the timer class.

TimerClassalarmClock=newTimerClass**();**

// Subscribe the first test class to the event in

// the timer class.

alarmClock.OnTimerExpired+=

newTimerClass.TimerExpiredHandler

**(**firstTestClass.DisplayAlarmMessage**);**

// Subscribe the second test class to the event in

// the timer class.

alarmClock.OnTimerExpired+=

newTimerClass.TimerExpiredHandler

**(**secondTestClass.ShowMessage**);**

// Turn on the alarm clock.

alarmClock.Run**(**numberOfSeconds**);**

**}**

Console.Write**(**"\n\nPress <ENTER> to end: "**);**

Console.ReadLine**();**

**}**

**}**

classAnotherEventTestClass

**{**

internalvoidShowMessage**(**objectsender**,** TimerEventArgse**)**

**{**

// Save the current color of the characters.

ConsoleColororiginalForeGroundColor=Console.ForegroundColor**;**

// Change the color of the characters to Magenta.

Console.ForegroundColor=ConsoleColor.Magenta**;**

for **(**inti=0**;** i<3**;** i++**)**

**{**

Console.WriteLine**(**"\n{0}: This is your wakeup call."**,**

e.AlarmTime**);**

Console.Beep**(**3500**,** 400**);**

System.Threading.Thread.Sleep**(**750**);**

**}**

// Reset the color of the characters to their original color.

Console.ForegroundColor=originalForeGroundColor**;**

**}**

**}**

**}**