**Задание.**

* Разработать таймер, реализующий возможность по истечении назначенного времени (время задаётся пользователем) передавать сообщение любому подписавшемуся на событие типу(классу).
* Для создания эффекта ожидания использовать метод Thread.Sleep пространства имен System.Threading.
* Пользователь может задавать имя таймера.
* Доработать тип делегата, чтобы получатель события мог определить «источник события» и вывести «имя таймера».
* Добавить событие с отсчётом времени, в котором будет передаваться «сколько секунд осталось».
* Предусмотреть возможность подписки на событие в нескольких классах.
* Создайте три(\*) пользовательских класса явно реализующие интерфейс ICutDownNotifier с методами Init() (подписывается на событие «таймера») и Run (запускает «таймер»)
  + Один обрабатывает события с помощью методов;
* Второй обрабатывает события с помощью анонимных делегатов;
* **\***Третий обрабатывает события с помощью лямбда выражений.
* В итого каждый из пользовательских классов должен обрабатывать три события:
  + Старт обратного отсчёта;
  + Осталось N-секунд;
  + Обратный отсчёт завершён.
* Каждый обработчик должен выводить «имя таймера» в обработчиках события.
* Приложение должно одновременно настроить и запустить все три таймера с названиями ««Чтение задание», «Выполнение задания» и «Проверка задания перед отправкой».
  + При этом надо работать с массивом объектов типа ICutDownNotifier.
* В качестве UI-интерфейса использовать консольное приложение с интерфейсом командной строки, WinForms или WPF-приложение.

**Важно:**

* Материалы для использования

<http://www.rsdn.ru/article/mag/200401/codestyle.xml#EVOAC>

* How to: Publish Events that Conform to .NET Framework Guidelines (C# Programming Guide)

<http://msdn.microsoft.com/en-us/library/w369ty8x.aspx>

* Framework Design Guidelines: Events and Callbacks

<http://msdn.microsoft.com/en-us/library/ms229041.aspx>

* Action<T> Delegate

<http://msdn.microsoft.com/ru-ru/library/018hxwa8.aspx>