# Проверяемое задание № 4

**Тема 4. Делегаты и события**

Лекция 4.1. Делегаты и события

**Цель работы**

Сформировать понимание назначения делегатов и навыки их использования для отделения операций от метода реализации. Научиться использовать лямбда-выражения для определения анонимного метода, объяснять назначение событий и уметь их использовать в приложениях.

**«Задание**

1. Разработать класс для имитации часов с обратным отсчетом (таймер), реализующий возможность по истечении назначенного времени (количество секунд ожидания проставляется пользователем) передавать сообщение любому подписавшемуся на событие типу.
2. Для создания эффекта ожидания использовать метод Thread.Sleep пространства имен System.Threading.
3. Добавить возможность для пользователя задавать имя таймера.
4. Доработать тип делегата, чтобы получатель события мог определить «источник события» и вывести «имя таймера».
5. Добавить событие с отсчётом времени, в котором будет передаваться «сколько секунд осталось».
6. Предусмотреть возможность подписки на событие в нескольких классах.
7. Создать три пользовательских класса, явно реализующих интерфейс ICutDownNotifier с методами Init() (подписывается на событие «таймера») и Run (запускает «таймер»):
   * первый обрабатывает события с помощью методов;

* второй обрабатывает события с помощью анонимных делегатов;
* третий обрабатывает события с помощью лямбда-выражений». [1]

1. «В итого каждый из пользовательских классов должен обрабатывать три события:
   * «старт обратного отсчёта»;
   * «осталось N секунд»;
   * «обратный отсчёт завершён».
2. Каждый обработчик должен выводить «имя таймера» в обработчиках события.
3. Создать делегат «Началось время выполнения задания» с параметрами «Название задачи», «Сколько было отведено времени».
4. Конструктор каждого из пользовательских классов, реализующих ICutDownNotifier, должен принимать два параметра:
   * созданный делегат «Началось время выполнения задания»;
   * делегат «Закончилось время выполнения задания» на основе стандартного типа Action<T> с параметрами «Название задачи», «Сколько было отведено времени».
5. Эти делегаты нужно вызывать в соответствующее время». [1]
6. «Создать в приложении метод, который будет информировать о начале и завершении времени выполнения заданий, и передать их в конструкторы «пользовательских классов».
7. 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.
8. В качестве UI-интерфейса использовать консольное приложение с интерфейсом командной строки, WinForms или WPF-приложение». [1]

**Важные материалы для выполнения задания (!)**

* Правила оформления событий –

<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>

**Требования к предоставлению результатов (!)**

**К отчету следует приложить:**

* скриншоты и краткое описание этапов работы;
* листинг программ, написанных на языке программирования C#.