

Сигнал

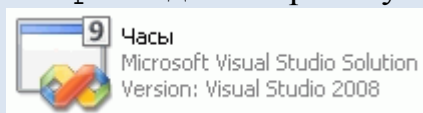
Поставим задачу спроектировать приложение, которое способно подать сигнал при наступлении заранее установленного Вами момента времени. Такое приложение может быть полезно, когда Вы, например, увлеченно работаете на компьютере, не хотите отвлекаться для слежения за временем, но в то же время не хотите пропустить наступление начала некоторого события, например, заранее назначенной деловой встречи.

1. Проектирование интерфейса

Возьмите за основу разработанный Вами предыдущий проект часов.

1. Копируйте в корневой каталог диска *d* свою рабочую папку рабочую папку со своего съемного диска или с диска *o*.

2. Откройте проект Часы. Для этого с помощью программы Мой компьютер войдите в рабочую папку и дважды щелкните на файле решения Часы



. Проект Часы будет открыт.

3. Чтобы убедиться в том, что Ваш проект по-прежнему работает, запустите приложение Часы. Убедились? Закройте приложение.

4. Переименуйте проект. Для этого в окне обозревателя решений выполните правый щелчок на имени проекта Часы. В выпавшем контекстном меню выберите команду Rename (Переименовать). Затем удалите старое имя проекта Часы, введите новое имя проекта Сигнал и завершите эту команду нажатием на клавишу **Enter**.

5. Если окно конструктора форм не открыто, то в окне обозревателя решений выполните двойной щелчок на файле `Form1.vb`.

6. В появившемся окне конструктора форм выделите форму и задайте ее свойству `Text` значение Мой сигнал.

7. Для задания значения часов и минут, определяющих момент времени начала звучания сигнала, примените управляющие элементы `TextBox` (текстовое поле). Текстовое поле позволяет в отличие от надписи редактировать находящийся в нем текст во время выполнения приложения. Согласно рис. 1 разместите на форме под надписью `lbl`. Время два текстовых поля: одно для задания часов, второе для задания минут начала звучания сигнала.

Вам потребуется иметь дело со следующими свойствами этого управляющего элемента:

`Name` – задает имя соответствующего объекта.

`Font` – позволяет задать параметры текста в текстовом поле.

`MaxLength` – максимальное количество символов, которое может быть введено в текстовом поле (по умолчанию 32767).

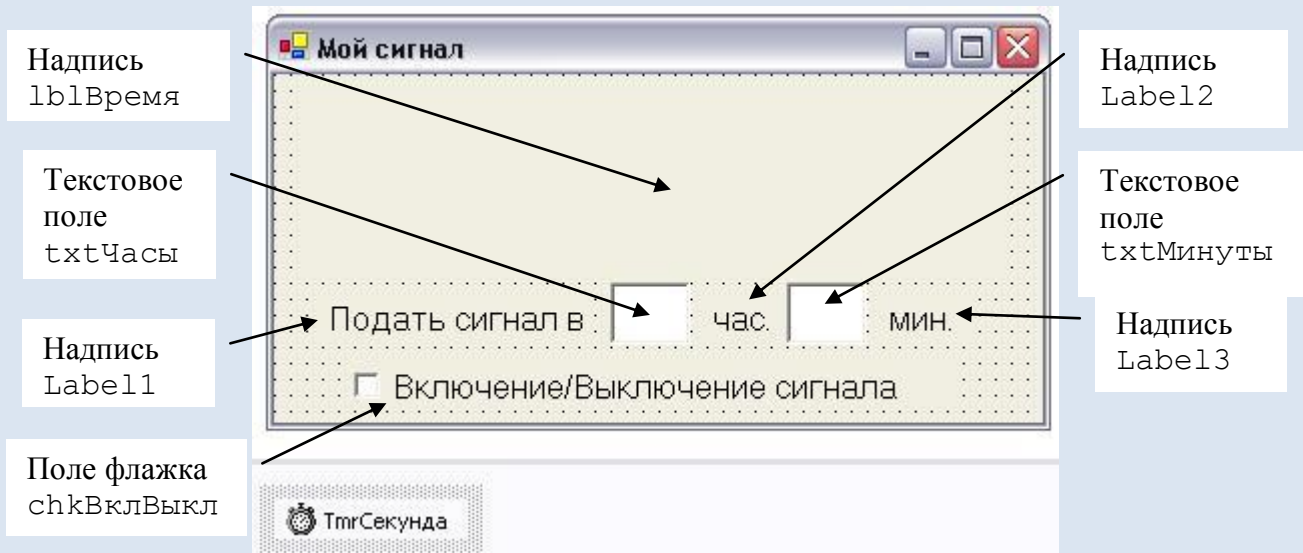


Рис. 1. Итоговый интерфейс проекта

8. Установите, что в текстовые поля можно вводить только по 2 символа (этого хватит для задания часа и минуты подачи сигнала). Для этого задайте свойствам `MaxLength` каждого текстового поля значение 2.

9. Установите размер текста в текстовых полях равным 16 (свойство `Font`).

10. Задайте свойству `Name` текстового поля `TextBox1` значение `txtЧасы`.

11. Задайте свойству `Name` текстового поля `TextBox2` значение `txtМинуты`.

12. Согласно рис. 1 поместите на форме три надписи `Label1`, `Label2` и `Label3`.

13. Установите в этих трех надписях размер текста, равный 12.

14. С помощью свойств `Text` задайте на каждой из трех надписей поясняющий текст в соответствии с рис. 1.

В отличие от простых часов приложение должно работать в одном из двух режимов:

- Сигнал выключен (соответствует работе в режиме обычных часов).
- Сигнал включен.

Переключение режимов должно быть возможно в режиме выполнения приложения. Для реализации этой функциональной возможности может быть применен управляющий элемент `CheckBox` (флажок).

При проектировании интерфейса Вам потребуется пользоваться следующими свойствами управляющего объекта `CheckBox`:

`Name` – имя соответствующего объекта;

`Text` – определяет содержание текста на соответствующем объекте;

`Font` – позволяет задать параметры текста на флажке;

`Checked` – имеет значение `False`, когда флажок не выбран, или значение `True`, когда флажок выбран.

15. Двойным щелчком на управляющем элементе `CheckBox` в окне управляющих элементов поместите флажок на форму.

16. Переместите флажок в нижнюю часть формы так, чтобы он размещался соответственно рис. 1.

17. Выделите объект `CheckBox1`. Задайте свойству `Text` этого объекта значение Включение/Выключение сигнала.

18. Задайте свойству `Name` объекта `CheckBox1` значение `chkВклВыкл`.

19. Задайте для флажка размер текста равным 12 (свойство `Font`). Проверьте, чтобы созданный Вами интерфейс полностью соответствовал рис. 1.

2. Разработка программного кода

Иногда ход вычислительного процесса должен зависеть от некоторого условия. Эту ситуацию, называемую разветвлением, поясняет рис. 2.

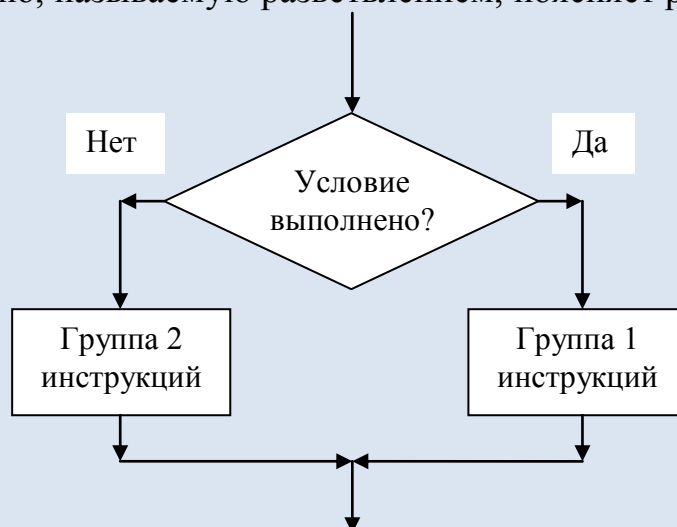


Рис. 2. Разветвление

Если условие выполнено, то работает группа 1 инструкций (ветвь Да), а группа 2 инструкций (ветвь Нет) игнорируется. Если же условие не выполнено, то работает группа 2 инструкций, а группа 1 инструкций пропускается. Как группа 1 инструкций, так и группа 2 инструкций может включать одну инструкцию (действие), несколько инструкций, а также не включить ни одной инструкции. Таким образом условие управляет ходом вычислительного процесса. На двухмерном рисунке это наглядно. Осталось разобраться, как это записать в коде, который является одномерной последовательностью инструкций.

Разветвление реализует условная инструкция `If` (если), которая выглядит так:

```


If ЛогическоеВыражение Then
    Группа1 инструкций
Else
    Группа2 инструкций
End If
  
```

Здесь слова `If`, `Then` (то), `Else` (иначе), `End If` (конец если) – это зарезервированные слова. О логических выражениях речь будет позже. Сейчас же достаточно иметь в виду, что значением логического выражения может быть одно

из двух: True (истина), что соответствует значению Да условия на рис. 2, или False (ложь), что соответствует значению Нет условия на рис. 2.

Итак, ветви Да и Нет размещаются между начальной и конечной строками разветвления (сначала ветвь Да), а границей между ними является строка Else. Строка Else вместе с группой 2 инструкций может отсутствовать. Работает этот код в соответствии с рис. 2. Если логическое выражение имеет значение True, то выполняется группа 1 инструкций, а группа 2 инструкций пропускается. Если же логическое выражение имеет значение False, то выполняется группа 2 инструкций, а группа 1 инструкций пропускается.

20. Откройте программный код проекта. Для этого щелкните на кнопке

 View Code (показать код). Подправьте код в подпрограмме tmrСекунда инструкции, чтобы он соответствовал листингу 1.

Листинг 1. Код подпрограммы tmrСекунда

```
Public Class Form1
    Private Sub tmrСекунда_Tick(ByVal sender As System.Object, _
        ByVal e As System.EventArgs) Handles tmrСекунда.Tick
        Static Сигнал As Boolean
        Сигнал = Not Сигнал
        lblВремя.Text = Now
        If chkВклВыкл.Checked Then
5:             If Format(Now.Hour, "00") = txtЧасы.Text Then
                    If Format(Now.Minute, "00") = txtМинуты.Text Then
                        Beep()
                        If Сигнал Then
10:                            lblВремя.Text = "Сигнал!"
                                End If
                            End If
                        End If
                    End If
        End Sub
End Class
```

Чтобы была понятна работа этого кода, следует пояснить понятие объект, которое является одним из важнейших понятий в объектно-ориентированном языке программирования VB.NET. Под объектом понимают совокупность данных и программ. Объекты обладают свойствами (данными) и методами (программами). Управляющий элемент, помещенный на форму, является объектом.

Для обращения в коде к свойству объекта применяется синтаксис:

ИмяОбъекта.ИмяСвойства

Для обращения в коде к методу применяется синтаксис:

ИмяОбъекта.ИмяМетода

Переменные в VB.NET тоже являются объектами и как объекты обладают свойствами и методами. Функция Now возвращает значение типа дата/время. Данные типа дата/время обладают свойствами Hour (час) и Minute (минута). Значения этих двух свойств являются целочисленными и равны часу (диапазон 0 – 23) и минуте (диапазон 0 – 59) соответствующего значения времени. Следовательно, значение свойства Now.Hour – это час текущего времени (целое число), и значение Now.Minute – это минута текущего времени (тоже целое число).

Перед сравнением этих целых чисел со значениями, записанными в двух позициях соответствующих текстовых полей, эти целые числа следует преобразовать в строки, так как значение свойства `Text` текстового поля является строкой. Для этого преобразования может быть применена функция `Format` (Значение, Формат) которая преобразует аргумент `Значение` в строку в соответствии с аргументом `Формат`.

Значения `Format(Now.Hour, "00")` и `Format(Now.Minute, "00")` – это соответственно тоже час и минута текущего времени, но это уже строки, а не целые числа.

Здесь аргумент `Формат`, равный строке `"00"` означает, что если значение преобразуемого к строке целого числа имеет не два, а один разряд (например, 4), то полученная в результате преобразования строка даже в этом случае будет иметь два символа (в данном примере не `"4"`, а `"04"`).

Строки 4 – 13 программного кода являются разветвлением. Они обеспечивают выполнение строк 5 – 12, если выбран флажок `chkВклВыкл` (сигнал включен).

Строки 5 – 12 в свою очередь являются разветвлением. Они обеспечивают выполнение строк 6 – 11, если час текущего времени равен установленному часу подачи сигнала. Логическим выражением в строке 5 является отношение. Знак равенства в этом отношении означает проверку равенства левой и правой частей этого отношения. Если равенство имеет место, то это отношение имеет значение `True`. Если же равенства нет, то это отношение имеет значение `False`.

Строки 6 – 11 также разветвление. Они обеспечивают выполнение инструкции `Beep()`, которая подает сигнал, если минута текущего времени равна установленной минуте подачи сигнала, а также изменение содержания текста надписи (строки 8 – 10).

Итак, сигнал будет звучать каждую секунду, если сигнал включен (флажок выбран), если при этом час текущего времени равен часу подачи сигнала, и если при этом минута текущего времени равна минуте подачи сигнала.

В проект кроме звукового эффекта добавлен и визуальный эффект, проявляющийся при подаче сигнала. Для этого изменяется содержание надписи `lblВремя`. При подаче сигнала в надписи поочередно отображается то текущая дата и время, то строка `"Сигнал!"`. Для этого в программном коде предусмотрены инструкции, содержащиеся в строках 1, 2 и 8 – 10.

В строке 1 объявлена локальная статическая переменная логического типа (об этом говорит слово `Boolean`) с именем `Сигнал`. Логический тип переменной означает, что она может принимать либо значение `True`, либо значение `False`. Пока ей явно не задавалось никакого значения, она после объявления по умолчанию имеет значение `False`. Слово `Static` в объявлении этой переменной означает, что это статическая переменная. Статические переменные сохраняют свое значение после окончания работы подпрограммы и выхода из нее. Вы можете быть уверенными, что при входе в подпрограмму `tmrTick` при повторном ее выполнении, переменная `Сигнал` имеет то же значение, которое она имела при предыдущем выходе из этой подпрограммы.

Строка 2 является инструкцией присвоения. Переменной Сигнал присваивается значение логической операции Not, операндом которой является та же переменная Сигнал. Действие этой инструкции присвоения очень простое. Если переменная Сигнал имеет значение False, то после выполнения такой инструкции присвоения она будет иметь значение True. Если же переменная Сигнал имеет значение True, то после выполнения такой инструкции присвоения она будет иметь значение False.

Строка 8 является разветвлением. Если переменная Сигнал имеет значение True, то отображение текущей даты и времени в надписи lblВремя будет заменено на отображение строки "Сигнал!". Обратите внимание на примененный синтаксис разветвления. Эта инструкция может занимать одну строку. После слова Then можно помещать одну инструкцию, которая будет выполнена только в том случае, если логическое выражение, стоящее перед словом Then имеет значение True. Но при применении такого способа записи разветвления конечная строка End If не требуется.

21. Сохраните проект и проверьте его работу. Устраните обнаруженные недостатки.
22. Попробуйте ответить на вопросы для контроля.
23. Продемонстрируйте работу Вашего проекта преподавателю.
24. Закройте свой проект и систему Visual Studio 2010.
25. Копируйте рабочую папку либо на свой съемный диск, либо на сетевой диск *o* в папку Вашей учебной группы.
26. Удалите на диске *d* свою рабочую папку.

3. Вопросы для контроля

1. Для чего применяется управляющий элемент TextBox?
2. На что влияют значения следующих свойств управляющего элемента TextBox: Name, Text, Font, MaxLength?
3. Для чего применяется управляющий элемент CheckBox?
4. На что влияют значения следующих свойств управляющего элемента CheckBox: Name, Text, Font, Checked?
5. Поясните работу инструкции If ... Then ... Else ... End If?
6. Приведите примеры объектов, применяемых в Вашем проекте.
7. Каково назначение свойств Hour и Minute?
8. Каково назначение функции Format?
9. Какие значения может принимать переменная логического типа?
10. В чем отличие локальной статической переменной от обычной локальной переменной?