Сканируем с AutoIt!

Данная статья имеет две цели: во-первых создать средство автоматического сканирования документов и книг, а во-вторых продемонстрировать возможности языка AutoIt.

Что такое AutoIt?

В русскоязычной справке написано следующее:

«AutoIt v3 - это язык для написания сценариев, напоминающий BASIC и предназначенный для автоматизации Windows GUI (графического интерфейса пользователя MS Windows). Его возможности - это методы симуляций нажатий комбинаций клавиш клавиатуры и мыши, функции управления окнами и его визуальными элементами. Решение, явно предлагаемое средствами AutoIt v3, его сценарием, обычно невозможно достичь коротким и понятным кодом какого-либо языка.»

Узнать подробнее об AutoIt и скачать дистрибутив можно с его домашней страницы: http://www.autoitscript.com

Стоит добавить, что скрипты также можно скомпилировать в независимые приложения.

Функциональность:

Для начала определимся с функциональностью скрипта:

1. Сканировать будем через WIA интерфейс, а точнее стандартной программой WinXP - wiaacmgr.exe расположенной в системной папке.
2. Скрипт должен иметь кнопки завершения работы, паузы/продолжения, быстрее/медленнее.
3. Скрипт не должен, по возможности, мешать работать на компьютере во время сканирования другому человеку или, например смотреть фильм на полном экране. ☺

Последовательность действий:

Начало.

Для начала нужно чтобы AutoIt был уже установлен в системе. Скрипты AutoIta представляют собой обычный текстовой файл и должны иметь расширение .au3. В качестве текстового редактора удобно использовать SciTE, идущий в комплекте дистрибутива AutoIt.

Итак, запускаем SciTE, и сохраняем еще пока пустой файл под названием autoscan.au3 – это и будет наш скрипт.

Первое что должен сделать скрипт – запустить «Мастер работы со сканером» - wiaacmgr.exe и поэтому первая строка должна быть такой:

```
Run ("wiaacmgr.exe")
```

Далее уже можно проверить работу скрипта перейдя в меню редактора SciTE : «Tools/Go» или просто нажав клавишу F5. При этом если сканер подключен к компьютеру должен запуститься «Мастер работы со сканером» и на экране появится вот такое окно:

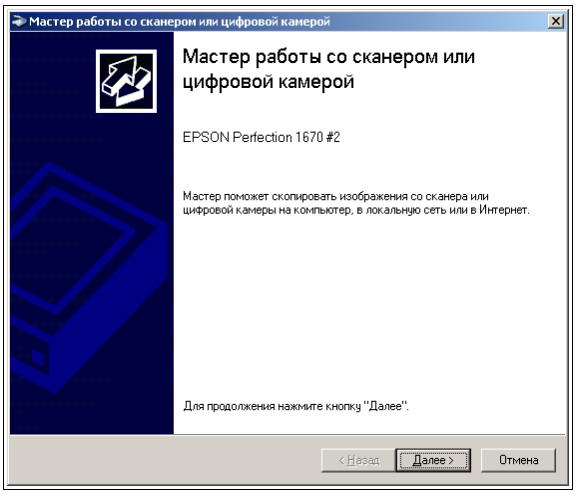

рис. 1.

Но т.к. скрипт будет выполнять и и другие действия, то желательно убедится, что «мастер» запустился нормально. Для этого добавим вторую строчку:

```
WinWait ( "Мастер работы", "Мастер поможет скопировать изображения")
```

Данная команда приостанавливает работу скрипта, до момента появления окна с заголовком «Мастер работы...» и содержащего текст «Мастер поможет скопировать изображения...»

Окна и элементы управления.

Следующий этап – научить скрипт программно нажимать кнопку «Далее». Это можно сделать несколькими способами: симуляцией нажатия кнопок мыши в определенной точке экрана командой MouseClick, эмуляцией нажатия клавиатуры командой Send, а также отправка сообщения клавиатуры (клавиша ENTER) непосредственно управляющему элементу командой ControlSend, в частности кнопке «Далее >» (см. рис. 1). Более надежен третий способ, т.к. позволит выполнять нажатия даже в фоновом режиме. Поэтому добавляем еще одну строчку в редакторе:

```
ControlSend ("Мастер работы", "Мастер поможет скопировать изображения", "Button2",
"{ENTER}" )
```

Первым аргументом функции является заголовок окна, вторым – текст окна, третим – элемент управления, четвертым – нажимаемая клавиша. Определить название элемента управления можно с помощью приложения «AutoIt Window Info» из состава дистрибутива AutoIt. Достаточно просто навести указатель мыши на нужный элемент управления и прочитать всю информацию о нем в окне «AutoIt Window Info», см. рис. 2.

Таким же образом находится информация и о других элементах управления, которые понадобятся в дальнейшем.

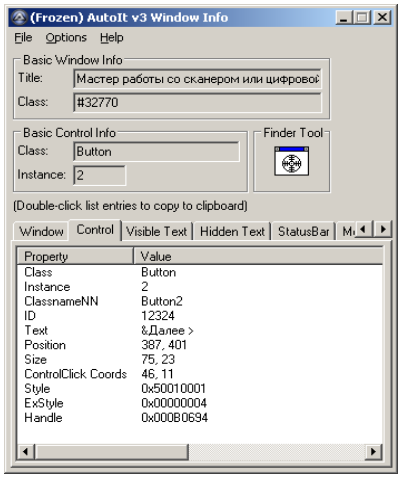

рис. 2

Закрываем «Мастер работы» и вновь запускаем скрипт для проверки клавишей F5, в результате появится такое окно (рис. 3):

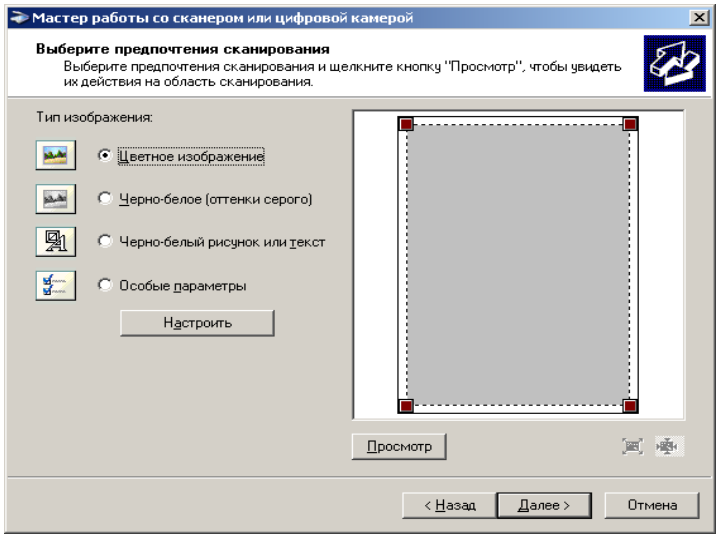

рис. 3

После чего предполагается, что пользователь настроит нужные параметры сканирования (рис. 4), определит область сканирования, сделает просмотр и пробное сканирование, выберет формат файлов, их название и размещение (рис. 5-7), просмотрит на результат сканирования в IrfanView или XnView, все это время скрипт должен ничего не делать, висеть в

системном трее и ждать, когда пользователь подберет окончательно все параметры и нажмет специальную горячую клавишу, для запуска сканирования уже в автоматическом режиме. Достигается это циклом:

```
While 1
    Sleep(1000); задержка в милисекундах
WEnd
```

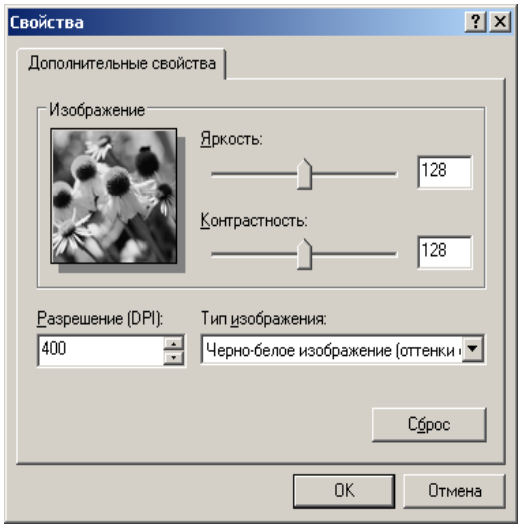

рис. 4.

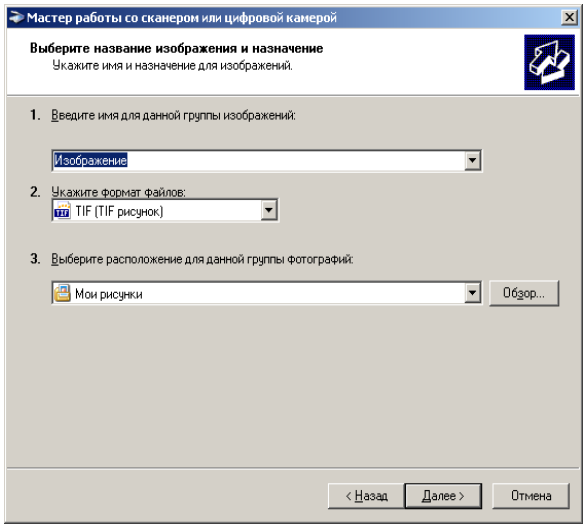

рис. 5.

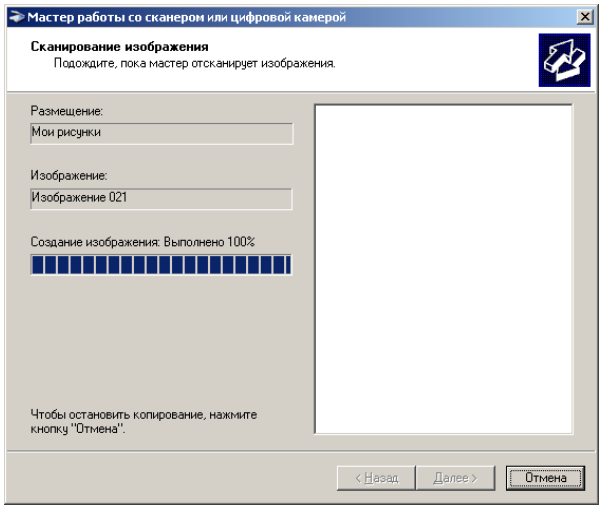

рис. 6.

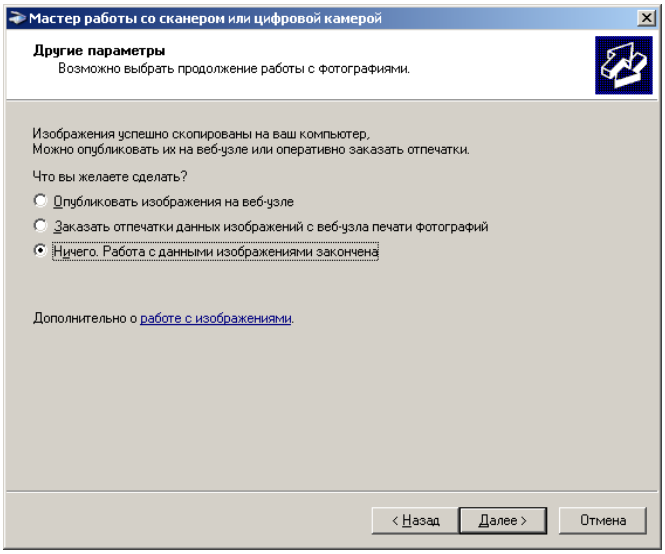

рис. 7.

Горячие клавиши.

Горячие клавиши или HotKey служат для выполнения определенных действий по нажатию заданной комбинации клавиш на клавиатуре. В нашем случае горячие клавиши будут следующими (при желании можно использовать и другие комбинации):

"s" – запуск автосканирования

"Pause" – пауза/продолжение автосканирования

"Esc" – окончание сканирования

"p" – быстрее

"o" – медленнее

```
HotKeySet("s", "RunScan")
HotKeySet("{PAUSE}", "TogglePause")
```

```
HotKeySet("{ESC}", "Terminate")
HotKeySet("p", "HiSpeed")
HotKeySet("o", "LowSpeed")
```

Здесь первый аргумент – горячая клавиша, второй – вызываемая функция.

```
Func Terminate()
  Exit 0
EndFunc
```

```
Func TogglePause()
    $Paused = NOT $Paused
    While $Paused
        sleep(100)
        ToolTip('Script is "Paused"',0,0)
    WEnd
    ToolTip("")
EndFunc
```

Переменная $Paused - глобальная

```
Global $Paused
```

While... Wend – цикл. ToolTip – всплывающая подсказка, sleep(100) – пауза на 100 мсек.

```
Func HiSpeed()
    If $Interval>300 Then
        $Interval = $Interval-300
            sleep(100)
        TrayTip ( "Interval", $Interval&" msec", 10 )
    EndIf
EndFunc

Func LowSpeed()
        $Interval = $Interval+300
        TrayTip ( "Interval", $Interval&" msec", 10 )
EndFunc
```

Переменная $ Interval – глобальная, интервал между двумя сканированиями, по умолчанию – 4000 мсек.

```
Global $Interval=4000
```

```
Func RunScan()
While 1
  WinWait ( "Мастер работы", "Изображения успешно скопированы")
  ControlClick ( "Мастер работы", "Изображения успешно скопированы", "Button11"  )
  WinWait ( "Мастер работы", "Тип изображения")
  ControlClick ( "Мастер работы", "Тип изображения", "Button12"  )
   WinWait ( "Мастер работы", "1.")
  ControlClick ( "Мастер работы", "1.", "Button12"  )
  WinWait ( "Мастер работы", "Изображения успешно скопированы")
  Sleep($Interval); задержка в милисекундах
WEnd
EndFunc
```

Финкция RunScan – основная. Её запуск начинается при нажатии на клавишу "s" и при этом «Мастер» должен быть открыт как на рис. 7. Цикл While-WEnd обеспечивает непрерывность сканирования, WinWait – ожидание соответствующих окон, ControlClick – нажатие на кнопки, Sleep – паузу.

Горячие клавиши и глобальные переменные объявляются в самом начале скрипта, функции – в конце.

Скриптом в принципе уже можно пользоваться, но все же лучше вообще свернуть окно «Мастера» , чтобы не мешало. Это делается добавлением в функцию RunScan строки

```
  WinSetState ( "Мастер работы", "", @SW_MINIMIZE )
```

и в функцию Terminate строки

```
WinSetState ( "Мастер работы", "", @SW_RESTORE )
```

Полный текст скрипта:

```
Global $Interval=4000
Global $Paused

Run ("wiaacmgr.exe")
WinWait ( "Мастер работы", "Мастер поможет скопировать изображения")
ControlSend ( "Мастер работы", "Мастер поможет скопировать изображения", "Button2",
"{ENTER}" )

HotKeySet("{PAUSE}", "TogglePause")
HotKeySet("{ESC}", "Terminate")
HotKeySet("p", "HiSpeed")
HotKeySet("o", "LowSpeed")
HotKeySet("s", "RunScan")

While 1
    Sleep(1000)
WEnd

Func RunScan()
 While 1
   WinWait ( "Мастер работы", "Изображения успешно скопированы")
   WinSetState ( "Мастер работы", "", @SW_MINIMIZE )
   ControlClick ( "Мастер работы", "Изображения успешно скопированы", "Button11"  )
   WinWait ( "Мастер работы", "Тип изображения")
   ControlClick ( "Мастер работы", "Тип изображения", "Button12"  )
   WinWait ( "Мастер работы", "1.")
   ControlClick ( "Мастер работы", "1.", "Button12"  )
   WinWait ( "Мастер работы", "Изображения успешно скопированы")
   Sleep($Interval)
 WEnd
EndFunc

Func Terminate()
   WinSetState ( "Мастер работы", "", @SW_RESTORE )
    Exit 0
EndFunc

Func TogglePause()
     $Paused = NOT $Paused
    While $Paused
        Sleep(100)
        ToolTip('Script is "Paused"',0,0)
    WEnd
    ToolTip("")
EndFunc

Func HiSpeed()
    If $Interval>300 Then
        $Interval = $Interval-300
       sleep(100)
        TrayTip ( "Interval", $Interval&" msec", 10 )
    EndIf
EndFunc

Func LowSpeed()
        $Interval = $Interval+300
        TrayTip ( "Interval", $Interval&" msec", 10 )
EndFunc
```

Компиляция скрипта

Для того, что бы скрипт можно было запускать в системе с неустановленным AutoIt скрипт нужно откомпилировать. Делается это клавишей F7 в редакторе SciTE.

Заключение

Стоит отметить, что кроме AutoIt существуют и другие средства автоматизации сканирования, например, макрос для Auto Macro Recorder (автор - **Astra55**), скрипты VBS.

По подобному принципу возможно создание скриптов «заточенных» под TWAIN интерфейс конкретной модели сканера.Также мною был создан AutoIt скрипт для Epson 1670, в качестве «движка» используется программа EpsonScan поставляемая со сканером.

Автор благодарит:

Astra55, за хорошую идею использовать стандартный WIA драйвер в качестве универсального «движка».

ViSiToR, за топик по AutoIt на Ru-Boarde.

03.01.08

U235

alexrey036<at>gmail.com