Table of Contents

1 AutoUpgrader - Overview

Overview

The AutoUpgrader component is able to automatically upgrade from the Web any Delphi/BCB application. The AutoUpgrader will check your website out for the newest releases of your software and, if the newest version is available— it will download and upgrade your application "*on the fly*", without restarting of computer and even without manual program restart.

With AutoUpgrader your customers will always have and use only *latest* versions of your software!

How to use

Just create the **Information file** (by hands or using InfoFile Designer), upload it to your website and point InfoFileURL property to this file. Every time on detecting the internet connection or on calling the CheckUpdate method, AutoUpgrader will read the upgrade information and, if newest release of your software is available, will try to upgrade your software itself.

How does upgrade mechanism works?

The AutoUpgrader downloads the newest version of your application to the "TEMP" directory (to "TEMP\Upgrade.tmp" file) and replaces your program with a newest version, when download done. After replacing, AutoUpgrader terminate and restart your application.
See detailed description of the **Upgrading Mechanism**...

Frequently Asked Questions

Q. *Does AutoUpgrader support proxy ? Can AutoUpgrader work behind a firewall ?*

A. Yes, AutoUpgrader support proxy and CAN work behind firewall through proxy. Since AutoUpgrader uses standard Internet routines of WinInet.dll library. All these functions users Internet settings which can be specified via Control Panel (see "Control Panel / Internet Options" or "Tools / Internet Options" menu item in the Internet Explorer, and check out "Connections" tab). AutoUpgrader just use system settings and routines (which also used by such programs like Internet Explorer or Outlook Express) as guarantee that all Internet requests will completed.

Q. *Does it require WinInet.dll ?*

A. Yes. However, you may do not worry about this. This library supplies with EVERY Windows version, since Windows 95 beta. AutoUpgrader perfectly works on Windows 95/98/ME and Windows NT4/2000.


C++ Builder users: Please add the "INET.LIB" file (in "..\Lib" directory) to your project before compiling your program with AutoUpgrader. "INET.LIB" contains the references to "wininet.dll" routines.

Q. *Can I upgrade more than one file ?*

A. No, unfortunately, this version of AutoUpgrader can upgrade just executable. Please check out AutoUpgrader Professional edition (v3.0+) at http://www.appcontrols.com/components.html page, or read online manuals on AutoUpgrader Pro at http://www.appcontrols.com/manuals/autoupgraderpro.html

Q. *Can I calculate amount of upgrades ?*

A. Yes, but this is server-side issue. Just specify in the InfoFileURL property location to your CGI program, instead of regular information file See an example CGI script in Perl or download it from http://www.appcontrols.com/misc/autoupgrade-cgi.zip This script stores number of upgrades in the flat database on your web server, but you can also get this value in OnUpgrade event (see *UsersServed* parameter).

2 Installation Instructions

Package without source code

to Delphi 2

1. Unzip files from "Delphi2" directory to your "Delphi 2\Lib" directory.
2. Start Delphi2 IDE.
3. Select "Component\ Install..." menu item.
4. Press "Add" button and select "_AUReg.pas" file.
5. Rebuild library.

to Delphi 3

1. Unzip files from "Delphi3" directory and copy them to "Delphi 3\Lib".
2. Start Delphi3 IDE.
3. Open "AutoUpgraderD3.dpk" file.
4. Install package to the components palette ("Install" button).

to Delphi 4

1. Unzip files from "Delphi4" directory and copy them to "Delphi 4\Lib".
2. Start Delphi4 IDE.
3. Open "AutoUpgraderD4.dpk" file.
4. Install package to the components palette ("Install" button).

to Delphi 5

1. Unzip files from "Delphi5" directory and copy them to "Delphi 5\Lib".
2. Start Delphi5 IDE.
3. Open "AutoUpgraderD5.dpk" file.
4. Install package to the components palette ("Install" button).

to Delphi 6

1. Unzip files from "Delphi6" directory and copy them to "Delphi 6\Lib".
2. Start Delphi6 IDE.
3. Open "AutoUpgraderD6.dpk" file.
4. Install package to the components palette ("Install" button).

to Delphi 7

1. Unzip files from "Delphi7" directory and copy them to "Delphi 7\Lib".
2. Start Delphi7 IDE.
3. Open "AutoUpgraderD7.dpk" file.
4. Install package to the components palette ("Install" button).

to C++ Builder 3

1. Unzip files from "BCB3" directory and copy them to "CBuilder3\Lib".
2. Start C++ Builder 3 IDE.
3. Open "AutoUpgraderCB3.bpk" file.
6. Select "Project \ Make AnimationEffectCB3" menu item.
7. Select "Component\ InstallPackages" menu item.
8. Press "Add" button and select "AutoUpgraderCB3.bpl" file.

to C++ Builder 4

1. Unzip files from "BCB4" directory and copy them to "CBuilder4\Lib".
2. Start C++ Builder 4 IDE.
3. Open "AutoUpgraderCB4.bpk" file.

4. Install package to the components palette ("Install" button).

to C++ Builder 5
1. Unzip files from "BCB5" directory and copy them to "CBuilder5\Lib".
2. Start C++ Builder 5 IDE.
3. Open "AutoUpgraderCB5.bpk"file.
4. Install package to the components palette ("Install" button).

to C++ Builder 6
1. Unzip files from "BCB6" directory and copy them to "CBuilder6\Lib".
2. Start C++ Builder 6 IDE.
3. Open "AutoUpgraderCB6.bpk"file.
4. Install package to the components palette ("Install" button).

Source code
1. Uninstall / delete all previous(trial) instances of AutoUpgrader.
2. Unzip files from "Sources" directory and copy them to "..\Lib" directory.
3. Run Delphi or ++ Builder IDE.
4. Select "Component\ Install..." menu item.
5. Press "Add" button and select "_AUReg.pas" file.
6. Rebuild library.

Note for C++ Builder users
Please add the "INET.LIB" file (in "..\Lib" directory) to your project before compiling the program with AutoUpgrader.

3 Registration Information

AutoUpgradercomponent is SHAREWARE. This means that you can try it out for free, but if you like it and want to use it you have to register it with the author. Before continue read and accept license agreement please.

The only differencebetween the unregisteredand registeredversions is that the registered one has not message box with remind to register and works without Delphi (C++ Builder) running. You can also purchase the source code, if you would like to have it, and be able to compile or modify the AutoUpgraderon any 32bit version of Delphi or C++ Builder.

If you would like to use the AutoUpgraderand receive full, unrestricted version, priority support or even source code — you haveto purchase proper license.

All prices in US dollars. Registering entitles you to unlimited support via E-Mail, minor version updates indefinitely and major version updates for 6 month from date of purchase.

Registration types:

Full, unrestrictedversion without source code:
Single user license:
- **https://secure.element5.com/register.html?productid=140737- $19,95**

Site license:
- https://secure.element5.com/register.html?productid=140738- **$99,95**

Full version including 100% Source Code:

Single user license:

- **https://secure.element5.com/register.html?productid=140739 - $39,95**

Site license:

- https://secure.element5.com/register.html?productid=140740 - **$159,95**

Comments

1. ***Site license*** covers a single organisation in one location (building complex). If you buy a site license, you may use the software in unlimited number of your company's computers withing this area. Site license is very cost-effective if you have many computers (many software developers).

See license agreement for more details.

4 License Agreement

Copyright

Liability disclaimer

Restrictions

You may not attempt to reverse compile, modify, translate or disassemble the software in whole or in part. You may not remove or modify any copyright notice or the method by which it may be invoked.

Operating license

Unregistered version

You may distribute the unregistered version of software freely, provided that all files are included and remain unmodified and that no extra files have been added to the package. You may not ask any money for the distribution. You may use the unregistered version of software free of charge for testing purposes, but if you want to use it for other purposes than testing - you have to register it with the author.

Registered version (single user license)

Once you have registered, you will receive a personal registered copy via email and login information to access your personal area at AppControls.com. This copy may not be copied or lend. You have the non-exclusive right to use registered version of the software only by a single person, on a single computer at a time. You may physically transfer the software from one computer to another, provided that the software is used only by a single person, on a single computer at a time. In group projects where multiple persons will use the software, you must purchase an individual license for each

member of the group or purchase site license. Use over a "local area network" (within the same locale) is permitted provided that the software is used only by a single person, on a single computer at a time. Use over a "wide area network" (outside the same locale) is strictly prohibited under any and all circumstances.

Registered version (site/team license)
Once you have registered, you will receive a personal registered copy via email and login information to access your personal area at AppControls.com. This copy may not be copied or lend. You have the non-exclusive right to use and transfer registered version of software on any number of computers by your company or your team only in one location (building complex). If you purchase a site license, you may use the program in an unlimited number of your company's computers within this area.

Registered version (Educational site license)
Once you have registered, you will receive a personal registered copy via email and login information to access your personal area at AppControls.com. This copy may not be copied or lend. You have the non-exclusive right to use and transfer registered version of software on any number of computers by your educational organisation (school/college/university etc) in one location (building complex). If you buy a educational site license, you may use the program in an unlimited number of your edicational organisation's computers within this area.

Registered version (World-wide license)
Once you have registered, you will receive a personal registered copy via email and login information to access your personal area at AppControls.com. This copy may not be copied or lend. You have the non-exclusive right to use and transfer registered version of software on any number of computers by your company or your team world-wide. If your company has many branches even with thouthands of computers, world wide license covers them all.

Notes (clarification)
"Single-user license" means "single-developer license". "Site license" means that it can be used by any number of software developers within your company.
You can use purchased components in ANY number of your projects and deploy the "end-user" software to ANY number of your users/customers without any additional royalty fees. Howeveryou are not permitted to distribute the component itself (the source code or .dcu files of components).

Back-up and transfer
You may make one copy of the software solely for "back-up" purposes, as prescribed by international copyright laws. You must reproduce and include the copyright notice on the back-up copy.

Terms
This license is effective until terminated. You may terminate it by destroying the program, the documentation and copies thereof. This license will also terminate if you fail to comply with any terms or conditions of this agreement. You agree upon such termination to destroy all copies of the program and of the documentation, or return them to author.

Other rights and restrictions
All other rights and restrictions not specifically granted in this license are reserved by authors.

5 Frequently Asked Questions

Q. *Does AutoUpgradersupport proxy ? Can AutoUpgraderwork behind a firewall ?*
A. Yes, AutoUpgradersupport proxy and CAN work behind firewallthrough proxy. Since AutoUpgrader uses standard Internetroutines of WinInet.dll library. All these functions users Internetsettings which can be specified via Control Panel (see "Control Panel / Internet Options" or "Tools / Internet Options" menu item in the InternetExplorer, and check out "Connections" tab). AutoUpgraderjust use system settings and routines (which also used by such programs like InternetExplorer or Outlook Express) as guaranteethat all Internetrequests will completed.

Q. *Does it requireWinInet.dll ?*
A. Yes. However,you may do not worry about this. This library supplies with EVERY Windows version, since Windows 95 beta. AutoUpgraderperfectly works on Windows 95/98/ME and Windows NT4/2000.
C++ Builder users: Please add the "INET.LIB" file (in "..\Lib" directory) to your project before compiling your programwith AutoUpgrader."INET.LIB" contains the referencesto "wininet.dll" routines.

Q. *Can I upgrade more than one file ?*
A. No, unfortunately,this version of AutoUpgradercan upgradejust executable. Please check out AutoUpgraderProfessional edition (v3.0+)at http://www.appcontrols.com/components.htmlpage, or read online manuals on AutoUpgraderPro at
http://www.appcontrols.com/manuals/autoupgraderpro.html

Q. *Can I calculate amount of upgrades ?*
A. Yes, but this is serverside issue. Just specify in the InfoFileURLproperty location to your CGI program, instead of regular information file See an example CGI script in Perl or download it from http://www.appcontrols.com/download/autoupgrade-cgi.zipThis script stores numberof upgrades in the flat database on your web server,but you can also get this value in OnUpgradeevent (see *UsersServed* parameter).

6 Properties

6.1 Active

Applies to
AutoUpgradercomponent.

Declaration
```
property Active: Boolean;
```

Description
The Active property controls whether the AutoUpgradercomponent can look up the Web for the software updates.

If Active is True and Internetconnection detected, AutoUpgraderwill readthe Info-file(which contains the upgradeinformation)from the Web site specified by InfoFileURLproperty.

Set this propertyto False if you wouldlike to check forupdates manually, using CheckUpdate method.

See also
InfoFileURL property, CheckUpdate method.

6.2 InfoFile

Applies to
AutoUpgrader component.

Declaration
```
type
  TAutoUpgraderInfo = class(TPersistent)
  published
    property DlgCaption: String;
    property DlgMessage: String;
    property SoftwareURL: String;
    property UpgradeMethod: TAutoUpgraderUpdateMethod;
  end;
```

Description
The InfoFile is set of properties that can be stored to the Information-file (which contains the upgrade information) from the InfoFile Designer

6.2.1 DlgCaption

Applies to
AutoUpgrader component as subproperty of InfoFile structure.

Declaration
```
property DlgCaption: String;
```

Description
The DlgCaption property specifies the title of message box that will be displayed when AutoUpgrader founds that new version is available.

Note
This property can be specified in InfoFile Designer

6.2.2 DlgMessage

Applies to
AutoUpgrader component as subproperty of InfoFile structure.

Declaration
```
property DlgMessage: String;
```

Description
The DlgMessage property specifies the message box that will be displayed when AutoUpgrader founds that new version is available.

Note
This property can be specified in InfoFile Designer

6.2.3 SoftwareURL

Applies to
AutoUpgrader component as subproperty of InfoFile structure.

Declaration
```
property SoftwareURL: String;
```

Description
The SoftwareURL property specifies the location to the new version of your software (URL to executable file).

Note
This property can be specified in InfoFile Designer

6.2.4 UpgradeMethod

Applies to
AutoUpgrader component as subproperty of InfoFile structure.

Declaration
```
type
 TacAutoUpgraderUpgradeMethod = (umAutoUpgrade, umRedirectToURL);

property UpgradeMethod: TAutoUpgradeMethod;
```

Description
The UpgradeMethod property controls how you would like to upgrade your software — auto-upgrading the executable "on the fly", or redirecting users to URL specified in SoftwareURL property.

Values

umAutoUpgrade	executable will be upgraded automatically by component.
umRedirectToURL	users will redirected to page specified in SoftwareURL

Note
This property can be specified in InfoFile Designer

6.3 InfoFileURL

Applies to
AutoUpgrader component.

Declaration
```
property InfoFileURL: String;
```

Description
The InfoFileURL property specifies the location in the Web (http:// address) where you've stored the Informationfile (Info-file), which contains the upgrade information for your software.

InfoFileURL may contains the line with following syntax:
http://www.yourdomain.com/download/yoursoftware/softwarename.inf
For example (real information file):
http://www.appcontrols.com/software/download/upgrades/wallpapermanager.inf

If you would like to calculate amount of upgrades — you may specify in the InfoFileURL location to the CGI script. See example of such CGI script here.

See also
Info-File example.

6.4 Password

Applies to
AutoUpgrader component.

Declaration
```
property Password: String;
```

Description
The Password property specifies the password to access the Info-file, if it has been uploaded to password protected web directory.

See also
Username property.

6.5 Username

Applies to
AutoUpgrader component.

Declaration
```
property Username: String;
```

Description
The Username property specifies the password to access the Info-file, if it has been uploaded to password protected web directory.

See also
Password property.

6.6 VersionControl

Applies to
AutoUpgrader component.

Declaration
```
property VersionControl: TVersionControl;
```

Description
The VersionControl property specifies which property will used for version control — VersionDate (`#date` variable in the Info-file) or VersionNumber (`#version` variable in the Info-file).

If VersionControl is "byDate", then after reading the Info-file from the Web, AutoUpgrader will compare `#date` variable of Info-file with VersionDate property. If VersionDate lower than date specified in `#date` variable, AutoUpgrader will think that newest version available in your Web and will try to upgrade the software on user's machine. VersionNumber property will be ignored.

If VersionControl is "byNumber", then after reading the Info-file from the Web, AutoUpgrader will compare `#number` variable with VersionNumber property. If VersionNumber is not equal to value specified in `#number` variable, AutoUpgrader will try to upgrade the software on user's machine, thinking that newest version available. VersionDate property will be ignored.

See also
VersionDate, VersionNumber properties and Info-file example.

6.7 VersionDate

Applies to
AutoUpgrader component.

Declaration
```
property VersionDate: String;
```

Description
The VersionDate property specifies the release date of current program version. This value can be automatically set to current date if VersionDateAutoSet property is True.

VersionDate property must have following format:
MM/DD/YYYY (for example, **02/24/2000**).

If VersionControl is "byDate", then after reading the Info-file from the Web, AutoUpgrader will compare `#date` variable of Info-file with VersionDate property. If VersionDate lower than date specified in `#date` variable, AutoUpgrader will think that newest version available in your Web and will try to upgrade the software on user's machine. VersionNumber property will be ignored.

See also
Info-file example, VersionDateAutoSet, VersionControl and VersionNumber properties.

6.8 VersionDateAutoSet

Applies to
AutoUpgrader component.

Declaration
```
property VersionDateAutoSet: Boolean;
```

Description
The VersionDateAutoSet property controls whether the value of VersionDate property should be specified automatically when application rebuilt.

If VersionDateAutoSet is True, VersionDate property will contains only current date (date of last recompiling).

See also
VersionDate property.

6.9 VersionNumber

Applies to
AutoUpgrader component.

Declaration
```
property VersionNumber: String;
```

Description
The VersionNumber property specifies the current version number of your software.

If VersionControl is "byNumber", then after reading the Info-file from the Web, AutoUpgrader will compare `#number` variable with VersionNumber property. If VersionNumber is not equal to value specified in `#number` variable, AutoUpgrader will try to upgrade the software on user's machine, thinking that newest version available. VersionDate property will be ignored.

See also
Info-file example, VersionControl and VersionDate properties.

7 Methods

7.1 CheckUpdate

Applies to
AutoUpgrader component

Declaration
```
procedure CheckUpdate;
```

Description
The CheckUpdate method reads the Info-file from the Web site, specified by InfoFileURL property and compare the `#date` variable with VersionDate property or #number variable with VersionNumber property.

If newest version of your software is available, it will try to upgrade your program.

See also
Abort method, OnUpgrade, OnNoUpdateAvailable, OnDone, OnError events.

7.2 Abort

Applies to
AutoUpgrader component.

Declaration
```
procedure Abort;
```

Description
The Abort method terminates the upgrade process. Call it to stop the upgrading/ downloading.

See also
CheckUpdate method.

8 Events

8.1 OnDone

Applies to
AutoUpgrader component.

Declaration
```
type
  TOnDoneEvent = procedure(Sender: TObject; FileSize: Integer) of object;

property OnDone: TOnDoneEvent;
```

Description
The OnDone event occurs when the newest version of your program has been successfully downloaded to TEMP directory and AutoUpgrader is about to terminate the program execution to auto-upgrade your software. FileSize variable specifies the size of new version (size of executable).

See also
OnError event.

8.2 OnError

Applies to
AutoUpgrader component.

Declaration
```
property OnError: TNotifyEvent;
```

Description
The OnError event occurs when some errors has occured on upgrading process.

Possilble reasons:
1. No Info-file has been found in location specified by InfoFileURL property
2. Remote connection has been closed (Internet connection lost).
3. Update file not found (path to update file specified in the `#url` variable of Info-file and SoftwareURL property).

See also
OnDone and OnNoUpdateAvailable events.

8.3 OnNoUpdateAvailable

Applies to
AutoUpgrader component.

Declaration
```
property OnNoUpdateAvailable: TNotifyEvent;
```

Description
The OnNoUpdateAvailable event occurs if no update available.

See also
OnUpgradeevent.

8.4 OnProgress

Applies to
AutoUpgradercomponent.

Declaration
```
type
  TOnProgressEvent = procedure(Sender: TObject; TotalSize, ReadSize,
ReadPercents: Integer) of object;

property OnProgress: TOnProgressEvent;
```

Description
The OnProgress eventoccurs when the newest versionof your softwareis availablein the Web and AutoUpgradercurrently downloadingthe new version.

Parameters
TotalSize - size of new versionof your executable
ReadSize - numberof bytes that has been read
ReadPercents - numberof percents that has been read

Example
```
procedure TForm1.AutoUpgradeProgress(Sender: TObject;
  TotalSize, ReadSize, ReadPercents: Integer);
begin
  ProgressBar1.Visible := True;
  ProgressBar1.Max := TotalSize;  // Total file size
  ProgressBar1.Position := ReadSize;  // Size of info that has been read
(in bytes)
end;
```

See also
OnUpgradeevent.

8.5 OnUpgrade

Applies to
AutoUpgradercomponent.

Declaration
```
type
  TOnUpgradeEvent = procedure(Sender: TObject; UsersServed: Integer; var
ShowMessageBox, CanUpgrade: Boolean) of object;

property OnUpgrade: TOnUpgradeEvent;
```

Description
The OnUpgradeeventoccurs once the newest versionavailablefor downloadand AutoUpgraderis ready for upgrading(downloading).

Optional customization

ShowMessageBox- controls whether you want to show the confirmation dialog. This is just optional property, in most cases you don't need to change this parameter, because all information that's displayed in this dialog box may be specified in the Information file

CanUpgrade- controls whether you want to upgrade your software. If you change CanUpgrade to False while processing this event then no confirmation will shown and software will not upgraded anyway.

UsersServed- returns the number of users that has been downloaded the new version. Use this value in case if you using the CGI script instead of Info-file This value returned by CGI as #served variable. If you're using regular Info-file, this value will always **-1**. If you would like to calculate the number of upgrades — please check an example of such CGI script or read FAQ first.

See also

OnNoUpdateAvailable, OnError events.

9 Usage

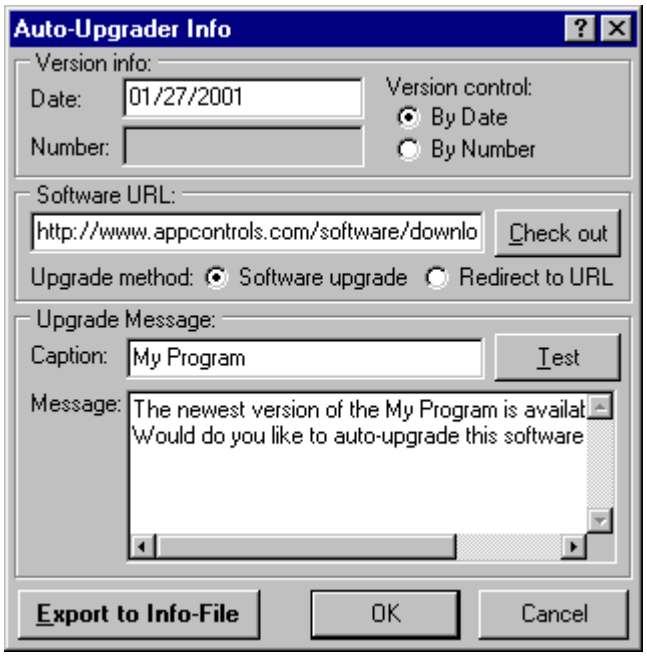
9.1 Usage example

```
procedure TForm1.INetDetector1Changed(Sender: TObject);
begin
  if INetDetector1.Online then
    AutoUpgrader1.CheckUpdate; // Auto-Upgrade
end;

// or

procedure TForm1.UpdateSoftwareMenuClick(Sender: TObject);
begin
  if Application.MessageBox('Check software updates in the Web ?', 'My
Software', mb_YesNo or mb_IconQuestion) = id_Ok then
   begin
    AutoUpgrader1.CheckUpdate; { Auto-Upgrade.
                                 If no update available... -
                                 see OnNoUpdateAvailable event }
   end;
end;
```

9.2 InfoFile Designer

The InfoFile designer of AutoUpgrader component will help you to create the Information files which contains the upgrade information.

Screenshoot



9.3 Info-file example

What is the information file and what for it need ?

Informationfile is the server side part of AutoUpgradercomponent, which contains the upgrade informationof your software. Info-fileit's flat file which should be stored in the web (to location specified by InfoFileURLproperty) and contains the information about the latest versionof your auto-upgradable application.

Once after detecting the internet connection (if Active property is True), or on calling the CheckUpdate method, the AutoUpgradercomponent will load this file from Web and

If VersionControlis "byDate" — compare the date (specified in the `#date` variable(**MM/DD/YYYY**), see below) with date of last compilation of your program (specified in the VersionDate property). If value of `#date` variablehigher than date specified in VersionDate property, then AutoUpgraderwill show the dialog with text specified in the `#message` variable(see example below) and with caption specified in the `#caption` variable.This dialog is the quesions - upgradethe programor not (with buttons "yes" and "no"). If user press "yes" button then AutoUpgraderwill attempt to downloadthe newest versionfrom location specified in the `#url` variable)for furtherauto-upgrading,or redirect user to this `#url` if `#redirect` value has set to "yes".

In case if you are using VersionNumberfor versioncontrol, then beforeupgrading, AutoUpgraderwill compare valuein `#version` variablewith valuespecified in VersionNumberproperty. If valuesis equal then AutoUpgraderwill try to auto-upgradesoftware same way.

Example of information file

```
# --- Begin
#date=02/26/2000
#version=2.0.1b
#message=[The newest version of the COOL program is available for
download.
```

```
List of new features:
1. Feature 1
2. Feature 2
Would you like to upgrade this program imediately ?]
#caption=The cool program
#url=http://www.yourdomain.com/download/cool/CollProgram.exe
#redirect=no
# ---  End
```

Note
First character of this file must be "#".

9.4 Calculation of upgrades

Here is an example of CGI script in Perl, which calculates amount of upgrades. If you would like to use it — just cut and paste this script from here, or download it from http://www.appcontrols.com/misc/autoupgrade-cgi.zip

This script stores number of upgrades in flat database file ($database) variable and return this value in OnUpgrade event (see *UsersServed* parameter).

```
#!/usr/bin/perl
##############################################################################
# Auto-Upgrade  counter  script        Version  1.0                           #
# (c)  2000  UtilMind  Solutions       info@utilmind.com                      #
#                                      http://www.utilmind.com                #
# Created  Jan  18,  1999              Last  Modified  Jan  18,  1999         #
##############################################################################
#   COPYRIGHT  NOTICE                                                        #
# Copyright  (c)  2000 by UtilMind  Solutions.  All  Rights  Reserved.       #
#                                                                            #
# This  program  distributed  as  server  side  part  of  AutoUpgrader  component.  #
##############################################################################
#   INSTALLATION  TIPS                                                       #
# Put  this  script  to  your  CGI-BIN  directory  and  chmod  it  to  755  (executable)  #
# Create  flat  database  file  (users_served.txt)   and  chmod  it  to       #
# 666  (read-write  permission)  and  install  it  to  CGI-BIN  too  (see  $database  #
# variable)                                                                  #
##############################################################################

####################
# Configuration

$database   = "users_served.txt";   # You  may  set  any  other  name  to  this  flat  database

$info_file  = "../autoupgrade.inf";   # !IMPORTANT:  Path  to  your  .Info-file

$use_flock  = 1; # 1 - yes, 0 - no. Should  be  used  but  if  you're  running  Perl  for
Win95/98  it  won't  work


####################
# Executable

print  "Content-Type:  text/html\n\n";
```

```
open(FILE,  $database)  || &error("Can't  locate  database  file");
$count  = <FILE>;
close(FILE);

$count++;  # increasing  the  counter

open(FILE,  ">$database")  || &error("Can't  update  database  file");
if  ($use_flock  eq  1)  {
  flock(FILE,  2);  # locking  write  accessing
}
print  FILE  $count;
if  ($use_flock  eq  1)  {
  flock(FILE,  8);  # unlocking
}
close(FILE);


open(FILE,  "$info_file")  || &error("Can't  locate  info-file");
@DATA  = <FILE>;
close(FILE);

print  @DATA;
print  "#served=$count\n";


sub  error  {
  print  "Error:  $_[0]<br>\n";
  print  "Reason:  $!";
  exit;
}
```

9.5 Upgrading Mechanism

Here is description of the upgrading mechanism to give an idea how does it works…

When the newest version available and user would like to upgrade the application (clicks "Yes" in the dialog box), the **AutoUpgrader** will:

1. Download newest file from the web to "TEMP\Upgrade.tmp" file.
2. When download is done, it creates the "Upgrader.exe" file in the "..\TEMP" directory,
3. Starts the "TEMP\Upgrader.exe",
4. Terminate the execution.

What does the **"Upgrader.exe"** ("Upgrader.exe" is invisible program... is not console but without own window):

1. Wait until the application terminate itself,
2. Replace old executable by downloaded file (Upgrade.tmp).
3. Start newest version (restart the application).
4. Terminate itself.

```
{ Here the source code of upgrader module (included to "AutoUpgrader.inc"
file in registered version).}
program Upgrader;

uses Windows;
```

```
var
  Msg: TMsg;

begin
// Must be started with 3 command line parameters:
//  1: Exe-name of application
//  2: Location of Upgrade.tmp (downloaded file)
//  3: Rename Upgrade.tmp to this name
  if ParamCount = 3 then
  begin
    repeat
      while PeekMessage(Msg, 0, 0, 0, PM_REMOVE) do
      begin
        TranslateMessage(Msg);
        DispatchMessage(Msg);
      end;
    until DeleteFile(PChar(ParamStr(1))); // Delete old version

    // Copy Upgrade.tmp to the newestversionname...
    MoveFile(PChar(ParamStr(2)), PChar(ParamStr(3)));

    // Delete Upgrade.tmp
    DeleteFile(PChar(ParamStr(2)));

    // Execute newest version
    WinExec(PChar(ParamStr(3)), sw_ShowNA);
  end;
end.
```