

# FinalBuilder 3

Automating the Build Process

---



# Table of Contents

|  |           |
|--|-----------|
| Foreword   | 0         |
| <b>Part I FinalBuilder</b>                           | <b>10</b> |
| 1 Overview .....                                     | 10        |
| 2 What's new in FinalBuilder 3.0 .....               | 11        |
| Important Changes .....                              | 15        |
| 3 License .....                                      | 16        |
| <b>Part II Getting Started</b>                       | <b>17</b> |
| 1 FinalBuilder Concepts .....                        | 17        |
| 2 FinalBuilder IDE .....                             | 17        |
| 3 Working with the Action Types Panel .....          | 18        |
| 4 Action Lists .....                                 | 19        |
| 5 Working with Actions .....                         | 21        |
| 6 Build Log .....                                    | 23        |
| 7 Options Dialog .....                               | 24        |
| 8 Project Summary .....                              | 24        |
| 9 Quick Help .....                                   | 25        |
| 10 Build History .....                               | 26        |
| 11 Run Status .....                                  | 27        |
| 12 Watches .....                                     | 27        |
| 13 Script Editor .....                               | 28        |
| 14 Messages .....                                    | 28        |
| 15 Validation Errors .....                           | 28        |
| 16 Tray Icon .....                                   | 29        |
| 17 Check for Updates .....                           | 31        |
| 18 Project Files & Other Files .....                 | 31        |
| 19 FinalBuilder Plugins .....                        | 31        |
| <b>Part III Scripting</b>                            | <b>32</b> |
| 1 Scripting in FinalBuilder .....                    | 32        |
| 2 Global Script functions .....                      | 33        |
| Format DateTime Formatting Options .....             | 35        |
| MessageBox Constants .....                           | 36        |
| 3 Action Script Events .....                         | 37        |
| Action Properties and Methods .....                  | 38        |
| 4 Accessing the Options settings via scripting ..... | 40        |
| 5 Accessing TStrings based properties .....          | 42        |

## Part IV Variables 43

|                                |    |
|--------------------------------|----|
| 1 Variables Overview .....     | 43 |
| 2 Adding Variables .....       | 44 |
| 3 Using Variables .....        | 44 |
| 4 Project Variables .....      | 45 |
| 5 User Variables .....         | 45 |
| 6 Environment Variables .....  | 46 |
| 7 System Variables .....       | 46 |
| 8 Action List Parameters ..... | 47 |

## Part V Actions Reference 49

|  |    |
|--|----|
| 1 Flow Control .....                           | 54 |
| Delay Action .....                             | 54 |
| Include FinalBuilder Project .....             | 55 |
| Run Action List Action .....                   | 55 |
| Exit Action List Action .....                  | 56 |
| While Loop Action .....                        | 56 |
| If .. Then Action .....                        | 57 |
| If Prev Action Failed Action .....             | 58 |
| Else Action .....                              | 58 |
| Stop Build Action .....                        | 59 |
| Raise Exception .....                          | 59 |
| Switch Action .....                            | 59 |
| Case Action .....                              | 59 |
| Try/Catch/Finally/End Actions .....            | 60 |
| 2 Version Control Actions .....                | 61 |
| ClearCase .....                                | 61 |
| ClearCase Pathnames .....                      | 62 |
| ClearCase Object Selector .....                | 63 |
| Base ClearCase .....                           | 65 |
| ClearCase Update Snapshot View Action .....    | 65 |
| ClearCase Check Out Action .....               | 67 |
| ClearCase Check In Action .....                | 68 |
| ClearCase Undo Checkouts Action .....          | 69 |
| ClearCase Find Checkouts Action .....          | 69 |
| ClearCase Make Element Action .....            | 70 |
| ClearCase Make Label Type Action .....         | 71 |
| ClearCase Apply Label Action .....             | 73 |
| ClearCase Make Attribute Type Action .....     | 74 |
| ClearCase Apply Attribute Action .....         | 76 |
| ClearCase Lock Action .....                    | 77 |
| ClearCase Unlock Action .....                  | 78 |
| ClearCase Run Cleartool (Generic) Action ..... | 78 |
| ClearCase Get Config Spec Action .....         | 79 |
| ClearCase Set Config Spec Action .....         | 80 |
| UCM .....                                      | 81 |
| UCM Make Activity Action .....                 | 81 |
| UCM Set Current Activity Action .....          | 83 |
| UCM Check In Files For Activity Action .....   | 84 |

|   |            |
|---|------------|
| UCM Undo Checkouts for Activity Action..... | 84         |
| UCM Make Baseline Action.....               | 85         |
| UCM Make Baseline from Label Action.....    | 87         |
| <b>Source Safe .....</b>                    | <b>88</b>  |
| Source Safe Check In File(s) Action.....    | 88         |
| Source Safe Check Out File(s) Action.....   | 90         |
| Source Safe Get Latest Version Action.....  | 91         |
| Source Safe Label File(s).....              | 92         |
| Source Safe Project Checkouts.....          | 93         |
| Source Safe Check File Status.....          | 94         |
| Source Safe Undo CheckOut Action.....       | 95         |
| Source Safe Add Files Action.....           | 96         |
| Source Safe Branch.....                     | 97         |
| Source Safe Share.....                      | 98         |
| Source Safe Get Working Directory.....      | 99         |
| Source Safe Override Global Options.....    | 99         |
| <b>Perforce .....</b>                       | <b>100</b> |
| Perforce Options.....                       | 100        |
| Perforce Create Branch.....                 | 100        |
| Perforce Delete Branch.....                 | 100        |
| Perforce Sync.....                          | 100        |
| Perforce Submit.....                        | 101        |
| Perforce open for Edit.....                 | 101        |
| Perforce Opened.....                        | 101        |
| Perforce Create Label.....                  | 101        |
| Perforce Update Label.....                  | 101        |
| Perforce Delete Label.....                  | 101        |
| Perforce Labelsync.....                     | 101        |
| Perforce open for Delete.....               | 101        |
| Perforce Tag.....                           | 101        |
| Perforce Lock.....                          | 101        |
| Perforce Unlock.....                        | 101        |
| Perforce Revert.....                        | 101        |
| Perforce Create Changelist.....             | 101        |
| Perforce Delete Changelist.....             | 101        |
| Perforce Generic.....                       | 101        |
| Perforce Old Actions.....                   | 102        |
| Perforce Synchronise with View Action.....  | 102        |
| Perforce Command Action.....                | 103        |
| <b>CVS Actions .....</b>                    | <b>103</b> |
| CVS Command Action.....                     | 105        |
| <b>Borland StarTeam Actions .....</b>       | <b>105</b> |
| StarTeam Check In Action.....               | 107        |
| StarTeam Check Out Action.....              | 107        |
| StarTeam Lock/Unlock Files Action.....      | 107        |
| StarTeam Create Label Action.....           | 107        |
| StarTeam Apply Label Action.....            | 107        |
| StarTeam Update Status Action.....          | 108        |
| StarTeam Delete Files Action.....           | 108        |
| StarTeam Add Files Action.....              | 108        |
| StarTeam List Files Action.....             | 108        |
| <b>QVCS Actions .....</b>                   | <b>108</b> |
| QVCS Add File Action.....                   | 108        |
| QVCS Check In File(s) Action.....           | 108        |

|   |            |
|---|------------|
| QVCS Check Out File(s) Action.....      | 108        |
| QVCS Get Latest Version Action.....     | 108        |
| QVCS Labels File(s) Action.....         | 108        |
| QVCS Undo Check Out File(s) Action..... | 108        |
| <b>QSC Team Coherence Actions .....</b> | <b>109</b> |
| Team Coherence Connect Action.....      | 110        |
| Team Coherence Set View Action.....     | 111        |
| Team Coherence Get Action.....          | 112        |
| Team Coherence Check In Action.....     | 113        |
| Team Coherence Check Out Action.....    | 114        |
| Team Coherence Attach Label Action..... | 115        |
| Team Coherence Detach Label Action..... | 116        |
| Team Coherence Create Label Action..... | 117        |
| Team Coherence Promote Action.....      | 118        |
| Team Coherence Sync.....                | 119        |
| Team Coherence Create View.....         | 120        |
| Team Coherence Update View.....         | 122        |
| Team Coherence Delete View.....         | 122        |
| <b>SourceGear Vault Actions .....</b>   | <b>122</b> |
| Vault Check Out Action.....             | 124        |
| Vault Get Action.....                   | 125        |
| Vault Get using Wildcards Action.....   | 126        |
| Vault Branch Action.....                | 127        |
| Vault Commit Action.....                | 128        |
| Vault Check In Action.....              | 129        |
| Vault Cloak Action.....                 | 130        |
| Vault UnCloak Action.....               | 131        |
| Vault Add Action.....                   | 132        |
| Vault Create Folder Action.....         | 133        |
| Vault Delete File/Folder Action.....    | 134        |
| Vault Get Version Action.....           | 135        |
| Vault Create Label Action.....          | 136        |
| Vault Move File/Folder Action.....      | 137        |
| Vault Rename File/Folder Action.....    | 138        |
| Vault Share File/Folder Action.....     | 139        |
| Vault Undo Checkout Action.....         | 140        |
| Vault Pin File/Folder Action.....       | 141        |
| Vault UnPin File/Folder Action.....     | 142        |
| Vault Get Label.....                    | 143        |
| Vault Set Working Folder.....           | 144        |
| Vault Diff.....                         | 145        |
| Vault GetLabelDiffs.....                | 146        |
| Vault ListCheckouts.....                | 147        |
| Vault File Status.....                  | 149        |
| <b>Subversion Actions .....</b>         | <b>149</b> |
| Subversion Add.....                     | 149        |
| Subversion Checkout.....                | 149        |
| Subversion Cleanup.....                 | 150        |
| Subversion Copy.....                    | 150        |
| Subversion Commit.....                  | 150        |
| Subversion Export.....                  | 150        |
| Subversion Import.....                  | 150        |
| Subversion Mkdir.....                   | 150        |
| Subversion Update.....                  | 150        |

|  |            |
|--|------------|
| Subversion Status .....                      | 150        |
| <b>Seapine Surround SCM .....</b>            | <b>150</b> |
| Surround SCM Global Options .....            | 150        |
| Surround SCM Override Global Options .....   | 151        |
| Surround SCM Get .....                       | 152        |
| Surround SCM CheckOut .....                  | 154        |
| Surround SCM CheckIn .....                   | 155        |
| Surround SCM Label .....                     | 156        |
| Surround SCM Create Branch .....             | 157        |
| Surround SCM Freeze Branch .....             | 158        |
| Surround SCM Unfreeze Branch .....           | 159        |
| Surround SCM Checkout Report .....           | 159        |
| Surround SCM Generic .....                   | 162        |
| <b>3 Iterators .....</b>                     | <b>163</b> |
| File Iterator .....                          | 164        |
| List Iterator .....                          | 165        |
| Folder Iterator .....                        | 166        |
| File Contents Iterator .....                 | 167        |
| INI File Iterator .....                      | 168        |
| <b>4 Interactive .....</b>                   | <b>169</b> |
| Prompt for Variables Action .....            | 169        |
| Prompt for Variables Action (Enhanced) ..... | 171        |
| Ask Question Action .....                    | 175        |
| Multi Question .....                         | 176        |
| Prompt for File or Directory .....           | 178        |
| <b>5 Misc Actions .....</b>                  | <b>179</b> |
| Action Group .....                           | 179        |
| Set Variable Action .....                    | 180        |
| Beep Action .....                            | 181        |
| Export Log Action .....                      | 182        |
| Get DateTime Action .....                    | 183        |
| Comment Action .....                         | 185        |
| Run Script Action .....                      | 185        |
| CityDesk Action .....                        | 186        |
| SaveVariablesToIni .....                     | 186        |
| LoadVariablesFromIni .....                   | 188        |
| <b>6 Files &amp; Directories .....</b>       | <b>190</b> |
| Check File Exists Actoin .....               | 190        |
| Delete File(s) Action .....                  | 191        |
| Create Directory Action .....                | 192        |
| Delete Directory Action .....                | 193        |
| Move Directory Action .....                  | 194        |
| Copy File(s) Action .....                    | 195        |
| Move File(s) Action .....                    | 197        |
| Set File Attributes Action .....             | 198        |
| Touch File(s) .....                          | 198        |
| Copy/Move File List .....                    | 199        |
| Text Replace Action .....                    | 201        |
| Concatenate Files Action .....               | 201        |
| Rename File or Directory Action .....        | 202        |
| Authenticode .....                           | 203        |
| Create Text File .....                       | 204        |
| Write Text File .....                        | 205        |

|  |            |
|--|------------|
| Read Text File .....                               | 206        |
| XCopy .....  | 207        |
| RoboCopy .....                                     | 207        |
| Extract File Version .....                         | 208        |
| Calculate File CRC32 .....                         | 209        |
| Calculate File MD5 .....                           | 210        |
| <b>7 Windows OS .....</b>                          | <b>211</b> |
| Execute Program Action .....                       | 211        |
| Run DOS Command Action .....                       | 213        |
| Subst Drive Action .....                           | 214        |
| Window Exists Action .....                         | 214        |
| Net Send Message Action .....                      | 215        |
| Get Disk Free Space .....                          | 216        |
| Register DLL/OCX Action .....                      | 217        |
| Control Service Action .....                       | 218        |
| WMI Run Process Action .....                       | 220        |
| WMI Kill Process Action .....                      | 221        |
| WMI Process Info Action .....                      | 222        |
| Shell Execute .....                                | 224        |
| <b>8 Compilers .....</b>                           | <b>226</b> |
| Java Compiler Action .....                         | 226        |
| Jikes Compiler Options.....                        | 228        |
| Borland Compiler Options.....                      | 228        |
| JDK Configurations.....                            | 228        |
| Microsoft .....                                    | 228        |
| Compile Visual Basic Project.....                  | 228        |
| Version Compatibility.....                         | 231        |
| Visual C++ 6 Action.....                           | 233        |
| Visual Studio .NET Action.....                     | 236        |
| Microsoft C# Compiler Action.....                  | 238        |
| Microsoft C# Project Compiler Action.....          | 239        |
| Microsoft VB.NET Compiler Action.....              | 240        |
| Microsoft VB.NET Project Compiler Action.....      | 241        |
| Microsoft J# Compiler Action.....                  | 242        |
| Microsoft J# Project Compiler Action.....          | 243        |
| Borland .....                                      | 244        |
| Compile Delphi Project.....                        | 244        |
| Compile Borland Resource Script.....               | 247        |
| Borland C++ Builder Action.....                    | 249        |
| Borland C++Builder Scripting Reference .....       | 250        |
| Borland C++Builder Compiler Options.....           | 252        |
| Borland C++Builder Advanced Compiler Options ..... | 252        |
| Borland C++Builder Linker Options .....            | 253        |
| Borland C++Builder Advanced Linker Options.....    | 253        |
| Borland C++Builder CPP Options.....                | 254        |
| Borland C++Builder TASM Options .....              | 254        |
| Borland C++Builder TLib Options.....               | 255        |
| Borland C++Builder Pascal Options.....             | 255        |
| Borland C++Builder Corba Options .....             | 255        |
| Borland C++Builder CodeGuard Options.....          | 256        |
| Borland C# Builder Project Compiler Action.....    | 256        |
| Borland Delphi 8 for .NET Action.....              | 257        |
| MadExcept Compiler Action .....                    | 257        |



|   |            |
|---|------------|
| AssemblyInfo Updater Action .....                   | 257        |
| Chrome .....  | 259        |
| Incredibuild .....                                  | 260        |
| <b>9 Installers .....</b>                           | <b>260</b> |
| Wise InstallBuilder/InstallMaster .....             | 260        |
| Wise For Windows Installer .....                    | 262        |
| InstallShield Pro - Std Edition .....               | 263        |
| InstallShield Pro - Windows Installer Edition ..... | 265        |
| InstallShield Developer .....                       | 267        |
| InstallShield Universal Installer .....             | 268        |
| Inno Setup .....                                    | 269        |
| GPInstall Action .....                              | 270        |
| InstallAnywhere Enterprise .....                    | 271        |
| InstallAnywhere .Net .....                          | 272        |
| InstallAware .....                                  | 273        |
| Nullsoft NSIS .....                                 | 274        |
| <b>10 Internet .....</b>                            | <b>275</b> |
| Send Email (SMTP) .....                             | 275        |
| FTP Client .....                                    | 281        |
| Telnet Client Action .....                          | 284        |
| HTTP Get Action .....                               | 286        |
| ICQ Action .....                                    | 286        |
| NNTP News Post Action .....                         | 287        |
| <b>11 Help Compilers .....</b>                      | <b>288</b> |
| Build Doc-O-Matic Project .....                     | 288        |
| WinHelp Compiler .....                              | 290        |
| HTML Help Compiler .....                            | 291        |
| Help & Manual 3 Action .....                        | 292        |
| NDoc Action .....                                   | 292        |
| <b>12 Ini Files &amp; Registry .....</b>            | <b>293</b> |
| Read Ini File .....                                 | 293        |
| Write Ini File .....                                | 294        |
| Read/Set/Delete Registry Value .....                | 296        |
| <b>13 Archiving .....</b>                           | <b>299</b> |
| Create Zip File .....                               | 299        |
| Extract Zip File Action .....                       | 302        |
| WinRAR Action .....                                 | 302        |
| 7Zip .....  | 303        |
| Create Archive.....                                 | 304        |
| Test Archive .....                                  | 307        |
| Extract Archive .....                               | 307        |
| Update Archive.....                                 | 308        |
| List Archive.....                                   | 309        |
| Delete from Archive.....                            | 309        |
| <b>14 Testing Tools .....</b>                       | <b>309</b> |
| AQTest .....  | 309        |
| AutomatedQA Test Complete Actions .....             | 311        |
| NUnit Action .....                                  | 311        |
| MSTest .....  | 312        |
| <b>15 Licensing Tools .....</b>                     | <b>312</b> |
| ASProtect Action .....                              | 312        |
| ProActivate Action .....                            | 313        |

|   |            |
|---|------------|
| Armadillo Action .....                  | 314        |
| <b>16 CD/DVD Burner Actions .....</b>   | <b>314</b> |
| Burn CD/DVD Action .....                | 315        |
| Create ISO Action .....                 | 317        |
| Burn ISO Action .....                   | 318        |
| Check Ready Action .....                | 319        |
| Erase CD/DVD RW .....                   | 320        |
| <b>17 .NET Actions .....</b>            | <b>321</b> |
| <b>.Net Framework Tools .....</b>       | <b>321</b> |
| Run AL.EXE .....                        | 321        |
| Register Assembly in COM [REGASM] ..... | 322        |
| Run ASPNET_REGIIS.EXE .....             | 323        |
| <b>.Net SDK Tools .....</b>             | <b>324</b> |
| Generate Key Pair [SN] .....            | 324        |
| Verify Strong Name [SN] .....           | 325        |
| Install Key in Container [SN] .....     | 326        |
| Extract Public Key [SN] .....           | 327        |
| Re-sign Assembly [SN] .....             | 328        |
| Run SN.EXE .....                        | 329        |
| GAC Install [GACUTIL] .....             | 330        |
| GAC Uninstall [GACUTIL] .....           | 331        |
| GAC Download Cache [GACUTIL] .....      | 332        |
| Type Library Import [TLBIMP] .....      | 333        |
| Type Library Export [TLBEXP] .....      | 334        |
| Run ResGen.exe .....                    | 335        |
| <b>3rd Party Tools .....</b>            | <b>336</b> |
| FxCop .....                             | 337        |
| Dotfuscator .....                       | 337        |
| Demeanor .....                          | 337        |
| XenoCode .....                          | 338        |
| <b>Other .....</b>                      | <b>339</b> |
| Fix TLBImp Project Reference .....      | 339        |
| <b>18 XML Actions .....</b>             | <b>340</b> |
| Transform XML .....                     | 340        |
| Merge XML Action .....                  | 340        |
| Extract XML Fragment Action .....       | 341        |
| Edit XML File Action .....              | 342        |
| Validate XML File .....                 | 343        |
| Delete XML Nodes .....                  | 344        |
| <b>19 Build Tools .....</b>             | <b>345</b> |
| Ant Project Action .....                | 345        |
| Nant Project Action .....               | 346        |
| MSBuild Project Action .....            | 347        |
| <b>20 Database .....</b>                | <b>348</b> |
| <b>SQL Server Actions .....</b>         | <b>348</b> |
| Execute SQL Action .....                | 348        |
| DTSRun Action .....                     | 351        |
| SQL Server Backup Database .....        | 352        |
| SQL Server Remove Unused Space .....    | 353        |
| SQL Server Check Database .....         | 354        |
| SQL Server Check Catalogue .....        | 354        |
| SQL Server Update DB Statistics .....   | 355        |

|   |            |
|---|------------|
| SQL Server Rebuild Indexes .....        | 355        |
| ADO Execute SQL .....                   | 355        |
| ADO Execute Stored Procedure .....      | 356        |
| 21 Source Code Tools .....              | 358        |
| Pascal Analyzer .....                   | 358        |
| <b>Part VI Wizards</b>                  | <b>359</b> |
| 1 Import Borland Project Wizard .....   | 359        |
| Select Project Group Page .....         | 360        |
| Select Compiler Version Page .....      | 361        |
| Common Settings Page .....              | 362        |
| Version Info Page .....                 | 363        |
| Import Options Page .....               | 364        |
| Finish Page .....                       | 365        |
| 2 Import VB6 Project Group Wizard ..... | 366        |
| Select Project Group .....              | 366        |
| Master Project Selection .....          | 367        |
| Common Output Path Setting .....        | 368        |
| Compiler Settings .....                 | 369        |
| Make Settings .....                     | 370        |
| Grouping .....                          | 371        |
| Finish .....                            | 373        |
| <b>Part VII Automating FinalBuilder</b> | <b>373</b> |
| 1 Scheduling Builds .....               | 373        |
| 2 Command Line Interface .....          | 376        |
| 3 Exit Codes .....                      | 377        |
| <b>Part VIII Reference</b>              | <b>377</b> |
| 1 Regular Expression Reference .....    | 377        |
| <b>Part IX Support</b>                  | <b>384</b> |
| 1 Known Problems .....                  | 384        |
| 2 FAQ .....                             | 385        |
| 3 FinalBuilder Support .....            | 386        |
| <b>Index</b>                            | <b>387</b> |

## 1 FinalBuilder



**Point. Click. Build. Deliver!**

**Automated Builds with FinalBuilder**

### 1.1 Overview



**Point. Click. Build. Deliver!**

**Automated Builds with FinalBuilder**

#### **What is FinalBuilder™?**

FinalBuilder is a powerful Automated Build Tool that allows developers to automate their software build process in a reliable and repeatable manner. FinalBuilder enables you to define your build process easily, allowing you to focus on more interesting and important tasks! Anyone can run the build, with one click (or keystroke!) in FinalBuilder. Builds can be scheduled via the windows scheduler, so automated nightly builds are easy to setup. FinalBuilder saves time, often running tasks in seconds that would take minutes or hours if done manually.

#### **Throw out those complicated Batch Files!**

Many developers use dos batch files to automate their builds. These batch files are typically difficult to maintain, have poor error handling, little or no error logging. FinalBuilder is easy to use, so the build process doesn't become the domain of one key person!

## How does FinalBuilder Work?

Most development tools such as compilers, install builders, version controls systems etc, support some sort of automation interface, for example a command line compiler, or a COM interface. FinalBuilder leverages these interfaces into a consistent and easy to use GUI application. FinalBuilder defines Actions, where each action provides an interface to some third party tool or to an internal function. Actions are chained together to create the build process. FinalBuilder also supports Active Scripting, each action exposes events which can be coded in VBScript or JavaScript.

## 1.2 What's new in FinalBuilder 3.0

### FinalBuilder 3 IDE Enhancements

- Check for FinalBuilder updates – check for updates checks for any version updates as well as any new or updated plugins
  - Automatic check for updates
  - Manual check for updates
- FinalBuilder Tray Icon
  - Option for tray icon to always show
  - Option to minimise FB to the system tray instead of task bar
  - Start and Stop Builds from Tray Icon
  - Open recent projects from Tray Icon
- Alert window for Completed Builds or when Build error if FinalBuilder is minimised
- Action Validation
  - Before a build is run FinalBuilder can validate all actions in the build and report any critical problems
  - Manual build validation
  - After an action is added or edited, it is automatically validated
  - Validation errors appear in the bottom section of the IDE, clicking on the error selects the action with the error
- Quick Help
  - Quick Help introduces the basic concepts for new users of FinalBuilder
  - Context sensitive Quick help enables the user to show quick help on any action in the Quick Help screen.
- All FinalBuilder settings are now stored in INI files instead of the Registry
- Edit Variables dialog has been rewritten to be more flexible and more user friendly
  - Variables can now be cut/copied/pasted
  - Variables can be transferred from Project to User or User to Project
- Variables architecture - It is now possible to define variables of different types with the same name, for example you can have a Project variable and an Environment variable with the same name. Only the variable with the highest precedence will actually be used, the precedence (highest to lowest) is Project, User, System, Environment. Overridden variables will show in red in the variables dialog, if a variable is overridden the highest precedence variable will show in green.

- Minor enhancements to action list tree (parent node is selected if node is hidden due to a node being collapsed)
- Modifying variables during a build is now possible using "Modify Variables" function in the Watch Variables panel
- Added frequently used actions group. Displays the top x number of actions used.
- Action List Parameters.
- Protected Action Packages. This Enterprise only feature enables action developers to deploy action packages in an encrypted form so that the source is not available. This also allows action developers to sell their FinalBuilder plugins with or without the source.
- Added extra script functions:
  - IncludeTrailingPathDelimiter
  - ExcludeTrailingPathDelimiter
  - ExpandFileName
  - FileExists
  - GetCurrentDir
  - SetCurrentDir
  - ExpandRelativePath
  - Alert
  - ExtractMajorVer
  - ExtractMinorVer
  - ExtractReleaseVer
  - ExtractBuildVer
- Suppress log output for an action
- Save log output for an action to an FB variable
- Editing fields in action properties dialog has been enhanced
  - Hint now displays "F2 to edit expression", "F3 to add variable" if the statement fails expand expression (eg. if a variable used has not been defined)
  - You can invoke the expression evaluator by pressing Ctrl-Space
  - The expression evaluator is automatically invoked when you type %
  - F3 allows you to add a variable
  - Global option "Show Expanded variables Hint when typing" to enable/disable automatic expression evaluator
- Expression evaluation now also possible in Edit Variable Dialog. Includes all the features of action property dialog hints, except for F3 (since you're already in the Edit Variables dialog).
- Added quick way to add exception handling around selected actions (available from the action context menu and also the Action menu):
  - Wrap Actions with Try Finally
  - Wrap Actions with Try Catch
  - Wrap Actions with Try Catch Finally
- MRU Item Count can be set
- Help file has been greatly improved

### FinalBuilder 3 Plugins

- FinalBuilder 3 introduces native .Net plugins, active script based plugins, and COM based plugins
- FinalBuilder ActionStudio... more details

### FinalBuilder 3 New Actions

- SubVersion – Subversion is an open source version control system developed by the authors of CVS
  - Subversion Add
  - Subversion Checkout
  - Subversion Cleanup
  - Subversion Commit
  - Subversion Copy
  - Subversion Export
  - Subversion Import
  - Subversion Mkdir
  - Subversion Update
  - Subversion Status
- FxCop
- Wise Owl Demeanor
- PreEmptive Dotfuscator
- Create Text File
- XenoCode
- MultiQuestion
- New Archive Actions (based on 7zip)
  - Create Archive (supports Zip, 7z, GZip, BZip2, TAR)
  - Test Archive (supports Zip, 7z, GZip, BZip2, TAR)
  - List files in Archive (supports Zip, 7z, GZip, BZip2, TAR, RAR, ARJ, CAB, CPIO, RPM, DEB, SPLIT)
  - Extract Archive (supports Zip, 7z, GZip, BZip2, TAR, RAR, ARJ, CAB, CPIO, RPM, DEB, SPLIT)
  - Update Archive (supports Zip, 7z, GZip, BZip2, TAR)
  - Delete file from Archive (supports Zip, 7z, GZip, BZip2, TAR)
- InstallAware
- XCopy – XCopy is a flexible way to copy files and directories and is faster than using the standard file copy action
- Robocopy – Robocopy is a very powerful tool for copying, moving, and synchronising directories and files with error recovery. It is included in the Windows NT Resource Kit
  - Robocopy (used for copying files and folders)
  - Robocopy run Job
  - Robocopy Move
  - Robocopy Mirror
- IncrediBuild from Xoreax – enables distributed builds of VisualStudio projects, see <http://www.xoreax.com>

- IncrediBuild build action
  - Enable IncrediBuild Agent
  - Disable IncrediBuild Agent
  - Stop current IncrediBuild Compile
  - Reset IncrediBuild Swapfile
- NullSoft Installer
- MStest
- ZeroG InstallAnywhere 6 Enterprise
- ZeroG InstallAnywhere .Net
- Folder Iterator action
- Prompt for File or Directory
- Chrome
- Get File Version Information Action
- Read Text File Action
- Write to Text file (insert at beginning, or append to end)
- File Contents Iterator Action – this will iterate for each line in the input file
- INI File Iterator Action – this will iterate over the sections in an INI file or over the name=value pairs of a section of an INI file
- List Iterator Action – this will iterate for each item in a list. It includes a script method "OnFirstRun" so that the list items can be dynamically set. Eg:
  - Action.ListOfItems.Clear
  - Action.ListOfItems.Add "item1"
  - Action.ListOfItems.Add "item2"
- Move directory action
- SQL Server maintenance actions
  - Backup SQL Server Database
  - Check DB
  - Check Catalogues
  - Remove Unused Space
  - Rebuild Index
  - Update DB Statistics
- Execute ADO stored procedure
- Vault
  - Diff Action
  - GetLabelDiffs Action
  - List Checkouts Action
  - SetWorkingFolder Action
  - FileStatus Action
- Calculate File CRC32
- Calculate File MD5
- ShellExecute. Allows you to execute a file with the Windows Shell (eg. print, edit, etc)
- Pascal Analyzer
- Save/Load Variables to/from INI file



- Microsoft Visual SourceSafe
  - Branch Action
  - Share Action
  - Get Working Folder Action

### Enhanced Actions

- HTTP Get now supports using a Proxy Server
- VSS – RespectCloakedProjects option exposed in property pages for Add, Check In, Get Latest, and Check Out.
- WinRAR – now can add multiple files, extra command line options, comments.
- WMI Execute Process, Kill Process, and Process Info – can now specify the credentials to use (ie. Username & Password)
- Microsoft VisualSourceSafe – the VSS action have been completely rewritten so that they operate against the vss command line utility rather than the COM API which has proved to be unreliable.
- ADO – allow SQL input from file
- TeamCoherence, added:
  - CreateView
  - DeleteView
  - UpdateView
- SourceGear Vault – added the ability for each action to provide no authentication or host information so that Vault can pick up the information from the vault\_cmdline\_client\_session.txt file
- SetVariable now allows you to specify the type
- Enhanced prompt for variables has been further enhanced. Now includes timeout, changing order of items, and file and directory types.
- Compile Delphi action – can now use \$(DELPHI), \$(BCB), and \$(BDS) in the following fields: Project File, Icon File, Starting Directory, Output Directory, Unit Output Directory, BPL Output Directory, and DCP Output Directory.

## 1.2.1 Important Changes

### Active Scripting

FinalBuilder 3 no longer supports the PerlScript & PythonScript scripting languages. The reason for this is that the available active script implementations of these languages do not fully support all of the active script interfaces required to allow FinalBuilder to inter-operate with them correctly.

## 1.3 License

Please Read the License Agreement for FinalBuilder before using this software.

### **FinalBuilder License Agreement**

This is the license agreement for FinalBuilder ("Software"). BY INSTALLING, COPYING, OR OTHERWISE USING THE SOFTWARE, YOU AGREE TO BE BOUND BY ALL OF THE TERMS AND CONDITIONS OF THE LICENSE AGREEMENT.

This Software is owned by VSoft Technologies Pty Ltd and is protected by copyright law and international copyright treaty. Therefore, you must treat this Software like any other copyrighted material (eg., a book), except that you may either make one copy of the Software solely for backup or archival purposes or transfer the Software to a single hard disk provided you keep the original solely for backup or archival purposes.

You may not alter any of the programs or accompanying files without written permission from VSoft Technologies Pty Ltd. Any resale or commercial distribution of the Software is strictly prohibited, unless VSoft Technologies Pty Ltd has given explicit written permission.

You have the right to use the Software as set forth in this licensing agreement. You are not obtaining title to the Software or any copyrights. You may not sublicense, rent, lease, convey, modify, translate, convert to another programming language, decompile, or disassemble the Product for any purpose.

VSoft Technologies Pty Ltd grants to you as an individual, a personal, nonexclusive license to install and use the Software for the sole purpose of developing and generating software systems. You may install a copy of the Software on a computer and freely move the Software from one computer to another, provided that you are the only individual using the Software. If you are an entity, VSoft Technologies Pty Ltd grants you the right to designate one individual within your organization to have the right to use the Software in the manner provided above.

### **Trial Version License Amendment**

The restrictions of this section ("Trial Version License Amendment") do not apply if you have purchased a commercial license for the Software and you are using the commercial version of this Software.

This version of the Software is a Trial version. This means that you may use the Software for Evaluation purposes only. You may use the Software to test whether it meets your demands. The Software is equipped with a mechanism that prevents the usage of the Product after a certain period of time has elapsed. You agree that you will delete the Software from all computer systems to which you have installed it when this date has been reached, or purchase a license to allow you to continue using the Product.

### **Disclaimer**

THIS SOFTWARE IS PROVIDED TO YOU "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED INCLUDING BUT NOT LIMITED TO THE APPLIED WARRANTIES OF MERCHANTABILITY AND/OR FITNESS FOR A PARTICULAR PURPOSE. YOU ASSUME THE ENTIRE RISK AS TO THE ACCURACY AND THE USE OF THE SOFTWARE AND ALL OTHER RISK ARISING OUT OF THE USE OR PERFORMANCE OF THIS SOFTWARE AND DOCUMENTATION. VSoft Technologies Pty Ltd SHALL NOT BE LIABLE FOR ANY DAMAGES WHATSOEVER ARISING OUT OF THE USE OF OR INABILITY TO USE THIS SOFTWARE, EVEN IF VSoft Technologies Pty Ltd HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE. TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, IN NO EVENT SHALL VSoft Technologies Pty Ltd BE LIABLE FOR ANY CONSEQUENTIAL, INCIDENTAL, DIRECT, INDIRECT, SPECIAL, PUNITIVE, OR OTHER DAMAGES WHATSOEVER,

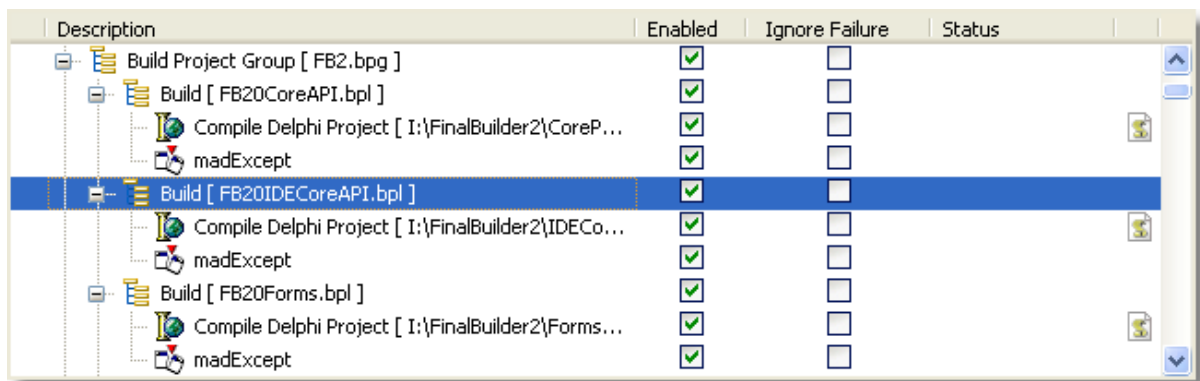
INCLUDING BUT NOT LIMITED TO DAMAGES OR LOSS OF BUSINESS PROFITS, BUSINESS INTERRUPTION, LOSS OF BUSINESS INFORMATION, OR OTHER PECUNIARY LOSS, EVEN IF VSoft Technologies Pty Ltd HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. BECAUSE SOME STATES/JURISDICTIONS DO NOT ALLOW THE EXCLUSION OR LIMITATION OF LIABILITY FOR CONSEQUENTIAL OR INCIDENTAL DAMAGES, THE ABOVE LIMITATION MAY NOT APPLY.

## 2 Getting Started

### 2.1 FinalBuilder Concepts

#### Actions and Action Lists

FinalBuilder uses Action Lists to perform the build process. An Action List is a sequence of discreet steps called Actions which are executed when the build process runs. An Action can perform tasks such as calling a compiler, copying files, creating directories etc.



| Description  | Enabled                             | Ignore Failure           | Status |
|--|-------------------------------------|--------------------------|--------|
| Build Project Group [ FB2.bpg ]                    | <input checked="" type="checkbox"/> | <input type="checkbox"/> |        |
| Build [ FB20CoreAPI.bpl ]                          | <input checked="" type="checkbox"/> | <input type="checkbox"/> |        |
| Compile Delphi Project [ I:\FinalBuilder2\CoreP... | <input checked="" type="checkbox"/> | <input type="checkbox"/> |        |
| madExcept  | <input checked="" type="checkbox"/> | <input type="checkbox"/> |        |
| Build [ FB20IDECoreAPI.bpl ]                       | <input checked="" type="checkbox"/> | <input type="checkbox"/> |        |
| Compile Delphi Project [ I:\FinalBuilder2\IDEC...  | <input checked="" type="checkbox"/> | <input type="checkbox"/> |        |
| madExcept  | <input checked="" type="checkbox"/> | <input type="checkbox"/> |        |
| Build [ FB20Forms.bpl ]                            | <input checked="" type="checkbox"/> | <input type="checkbox"/> |        |
| Compile Delphi Project [ I:\FinalBuilder2\Forms... | <input checked="" type="checkbox"/> | <input type="checkbox"/> |        |
| madExcept  | <input checked="" type="checkbox"/> | <input type="checkbox"/> |        |

By default each FinalBuilder project has a Main Action List and an OnFailure Action List. The build starts at the first enabled action in the Main action list, if an error occurs then the build will switch to the OnFailure Action List (assuming it is not empty) and continue from there. You can think of the OnFailure Action List as a global error handler, it is from here you can perform cleanup tasks when a build fails, such as deleting temporary or intermediate files etc. Action Lists are covered later in this help file.

#### User Action Lists

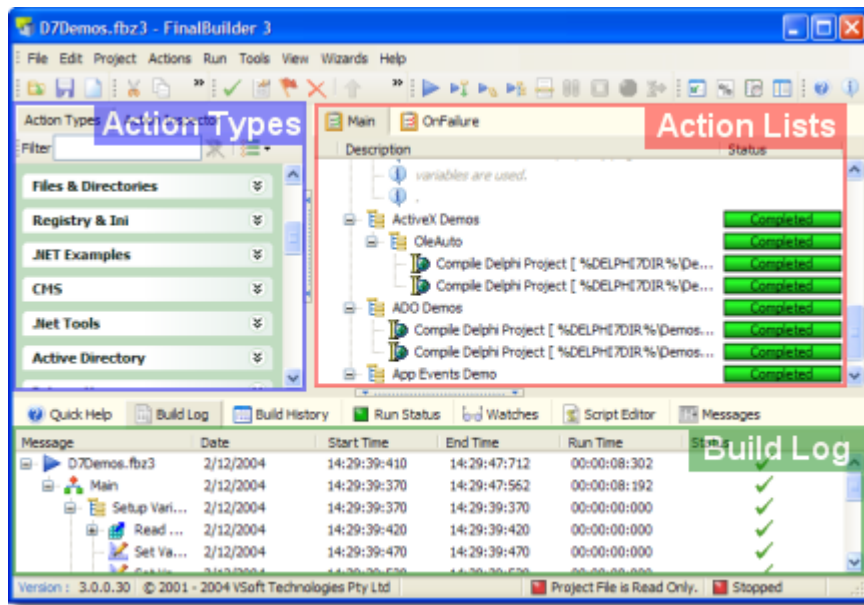
In addition to the default Action Lists, FinalBuilder also allows you to define custom Action Lists for each project, which can then be run using the Run Action List action.

#### See Also

Working with the Action Types Panel | Action Lists

### 2.2 FinalBuilder IDE

The FinalBuilder IDE Workspace is comprised of 3 main panels :



The Action Types panel lists the available action types that you can use in your build process. To add an action to an Action list, just click on the Action Type that you want to add, it will be inserted after the currently selected action in the current action list, or you can drag and drop the action on the list where you would like it. This is covered in more detail later in Working with the Action Types Panel. The Action Inspector (which is located as another tab in the Action Types panel) provides a quick way to change some of the properties of the actions, more complex properties are set from the properties dialog available for each action.

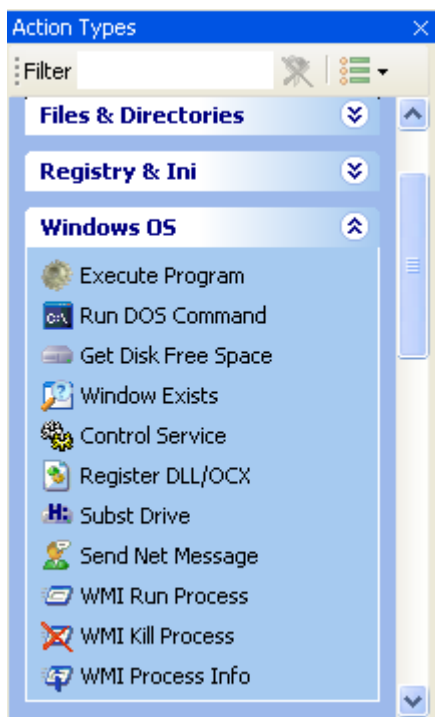
The bottom section contains the following pages:

- Build Log
- Quick Help
- Build History
- Run Status
- Watches
- Script Editor
- Messages
- Validation Errors

Once you have created your actions, click on the Run button to run the build. The Build Log tab at the bottom of the IDE will show the output from the actions.

## 2.3 Working with the Action Types Panel

The Action Types Panel is your starting point in the FinalBuilder IDE.




The available Action Types are grouped together in categories. The categories are based either on functionality, or in some cases on the third party product they support, for example "Source Safe" or "CVS". You can search for Action Types by name using the Filter edit box at the top of the Action Types Panel. You can use the keyboard shortcut Ctrl+I to set the focus to the filter edit control :



To Cancel the Filter click on the "Clear Filter" button or press Escape :



To Select an Action Type Category quickly, use the Action Categories  drop down menu button and select the category from the menu.

## See Also

Action Lists

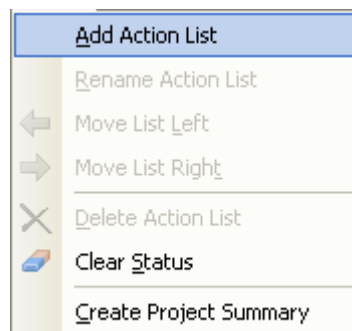
## 2.4 Action Lists

By Default, a new FinalBuilder project has 2 Actions Lists, Main and OnFailure. These default Action Lists cannot be deleted or moved. When a build starts, the first enabled action in the Main Action List is executed, after that if any action fails, the execution switches to the OnFailure Action List (provided it actually has any actions in it).

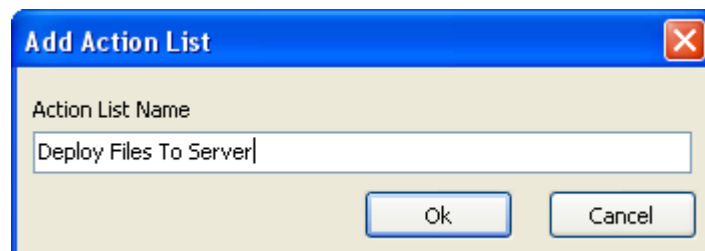
| Description  | Enabled                             | Ignore Failure           | Status |
|--|-------------------------------------|--------------------------|--------|
| Build Project Group [ FB2.bpg ]                    | <input checked="" type="checkbox"/> | <input type="checkbox"/> |        |
| Build [ FB20CoreAPI.bpl ]                          | <input checked="" type="checkbox"/> | <input type="checkbox"/> |        |
| Compile Delphi Project [ I:\FinalBuilder2\CoreP... | <input checked="" type="checkbox"/> | <input type="checkbox"/> |        |
| madExcept  | <input checked="" type="checkbox"/> | <input type="checkbox"/> |        |
| Build [ FB20IDECoreAPI.bpl ]                       | <input checked="" type="checkbox"/> | <input type="checkbox"/> |        |
| Compile Delphi Project [ I:\FinalBuilder2\IDEC...  | <input checked="" type="checkbox"/> | <input type="checkbox"/> |        |
| madExcept  | <input checked="" type="checkbox"/> | <input type="checkbox"/> |        |
| Build [ FB20Forms.bpl ]                            | <input checked="" type="checkbox"/> | <input type="checkbox"/> |        |
| Compile Delphi Project [ I:\FinalBuilder2\Forms... | <input checked="" type="checkbox"/> | <input type="checkbox"/> |        |
| madExcept  | <input checked="" type="checkbox"/> | <input type="checkbox"/> |        |

### Adding Action Lists

You can also Add/Delete/Rename or re-order extra Action Lists from the Project menu :



After clicking on Add Action List, provide a unique name for the new Action List.



To run these extra action lists, add a "Run Action List" action to the calling action list, then set the ActionList property of that action.

| Description     | Enabled                             | Ignore Failure | Status |
|-----------------|-------------------------------------|----------------|--------|
| Run Action List | <input checked="" type="checkbox"/> |                |        |

These custom Action Lists can be treated like subroutines, you can call them as often as you require in your Build Process. Note that recursion is not supported. For example, the above "Deploy Files To Server" Action List could be used to deploy to many different servers through the use of FinalBuilder Variables.

## See Also

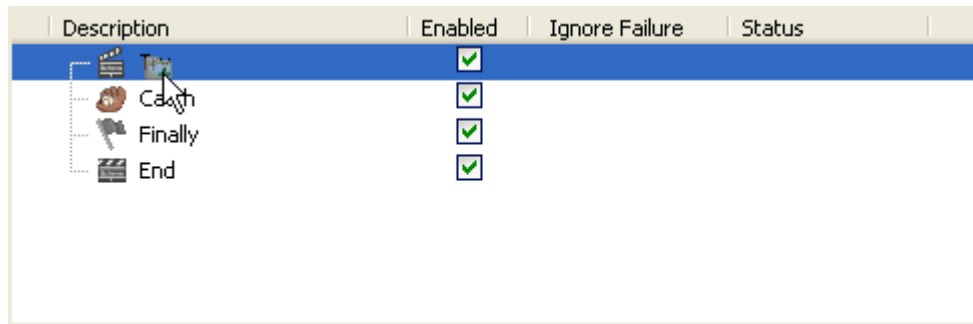
FinalBuilder Variables |

## 2.5 Working with Actions

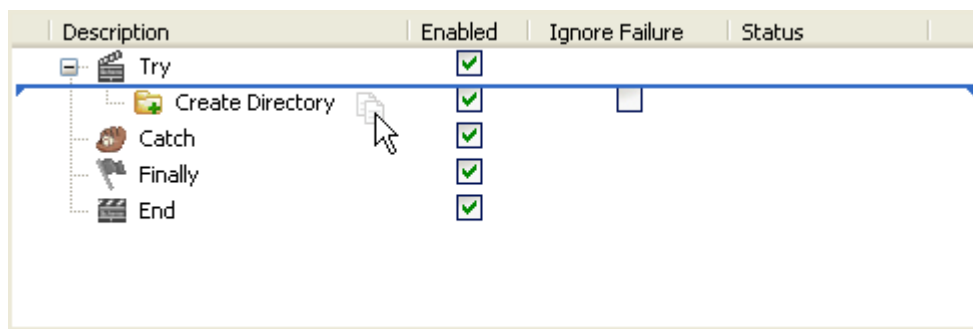
### Adding Actions to an Action List

You can add actions to an Action List using two methods. Simply clicking on the name of an Action Type in the Action Types Panel will add the Action to the Action List After the currently selected Action in the Action List. You can also use Drag n Drop to add actions to the Action list. Using Drag n Drop enables you to place the new action with more precision.

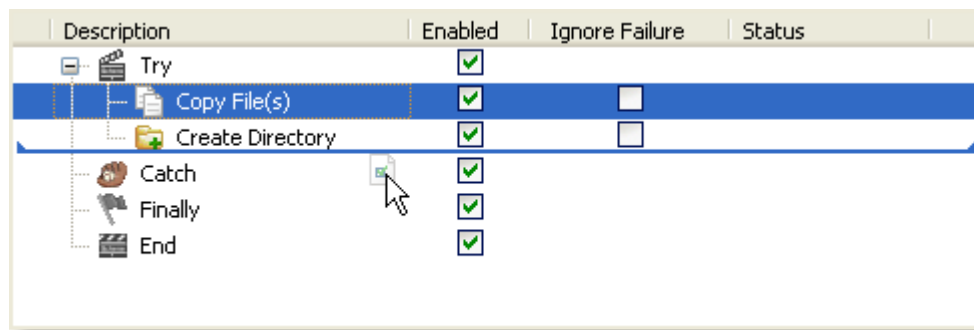
The Action List Tree view provides guide lines to indicate where the action will be dropped. In this example the action will be dropped as a child to the highlighted action, because the mouse is over the icon or name of the action,



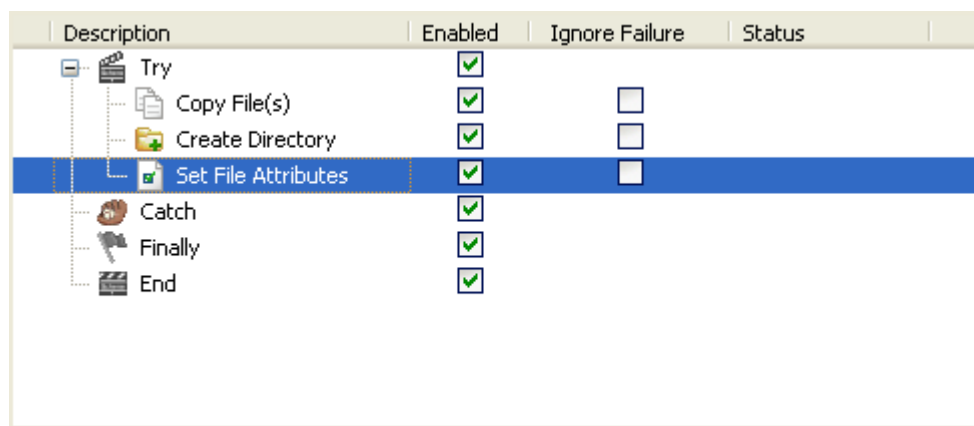
In this example the Action will be dropped Before the action under the guide line, because the guide line indicators point down.



In this example the Action will be dropped After the action above the line, because the guide line indicators point up.



After the above drag n drop operations our action list looks like this :



## Moving Actions

Actions can be moved using Drag n Drop, or using the Arrow buttons on the Actions Toolbar.



You can also use the Ctrl+Arrow keys to move actions up/down or indent/outdent.

Any Action can be a parent to other actions. Comment Actions are the only exception to this, they are pseudo actions that never actually execute, and thus cannot have child actions. When an Action has Child Actions, it executes first, and then if it succeeded then the child actions execute.

## Selecting Actions

The usual windows selection rules apply, using the control and shift keys to multi-select, however you can only multi-select actions at the same level in the tree.

## Copying Actions

Actions can be copied and pasted using the clipboard, in which case the actions are pasted after the currently selected Action. You can also use Drag n Drop with the Control



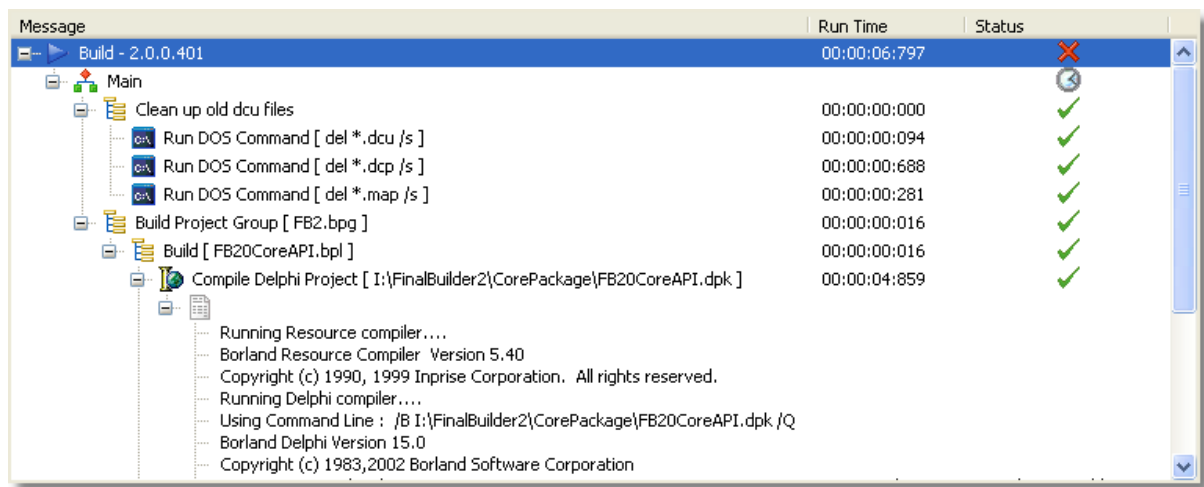
Key down to copy and drag the selected Actions.

### Deleting Actions






To Delete an Action, select it and press delete, or use the Delete button on the Actions Toolbar.

## 2.6 Build Log

The Build Log Tab contains a tree which contains nodes representing the actions that have been executed, with any output from these actions.



The status Column displays an icon representing the current status of an action :

-  Running - The action is currently running
-  Skipped - The Action was skipped, either because the Condition was not met, or because the SkipAction parameter in the BeforeAction event handler was set to True
-  Completed - The Action completed successfully
-  Error - The Run usually ends when an action has this status.
-  Error Ignored - An Error occurred, but the Ignore Failure property was set to True.

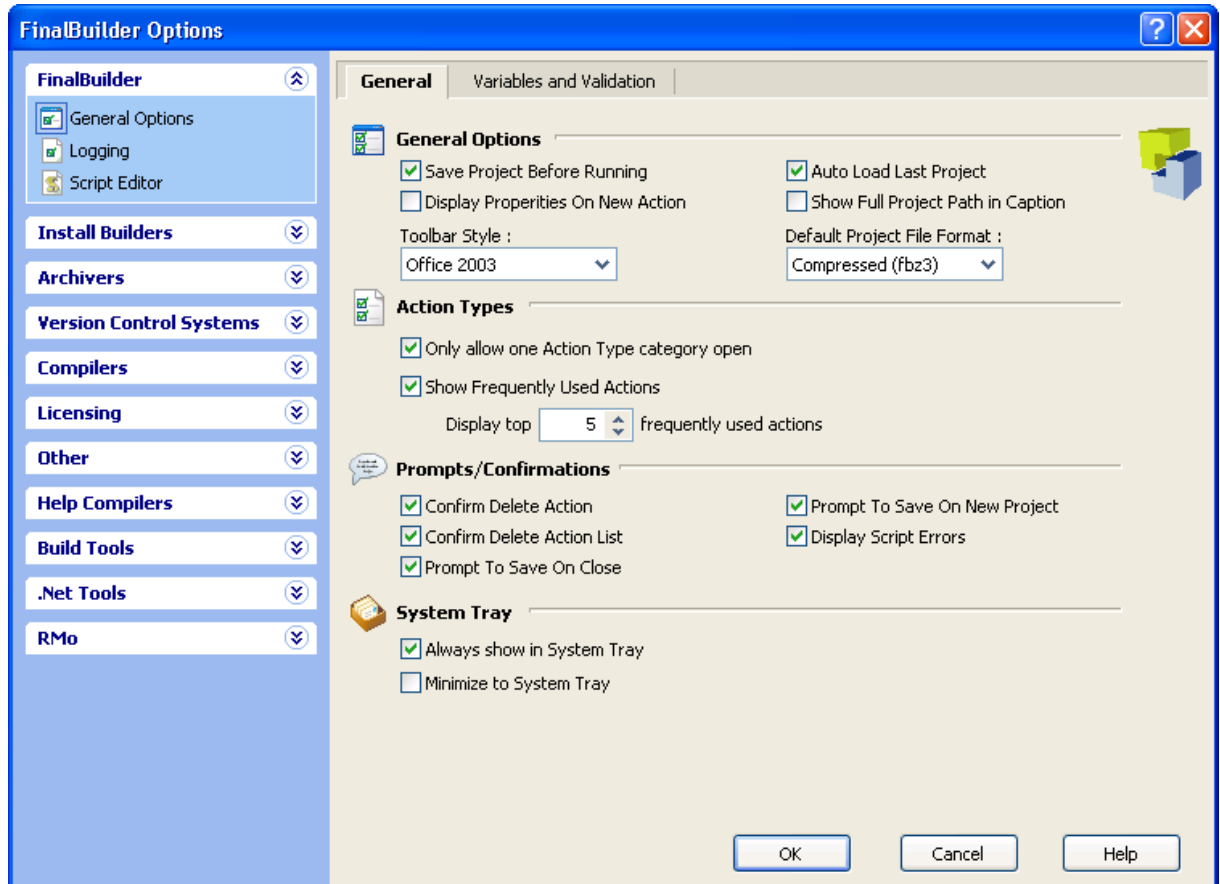
You can copy the text from the log text nodes by selecting the node and pressing Ctrl C.

### Live Log View

When a build is executing, updating the log can be very CPU intensive. You can use the "Live Log View" option in the Build Log tab to disable live logging if the CPU overhead of the live logging is effecting your build process. All the log information is still written to the log file, and when the build completes you can open the most recent build to examine the log.

## 2.7 Options Dialog

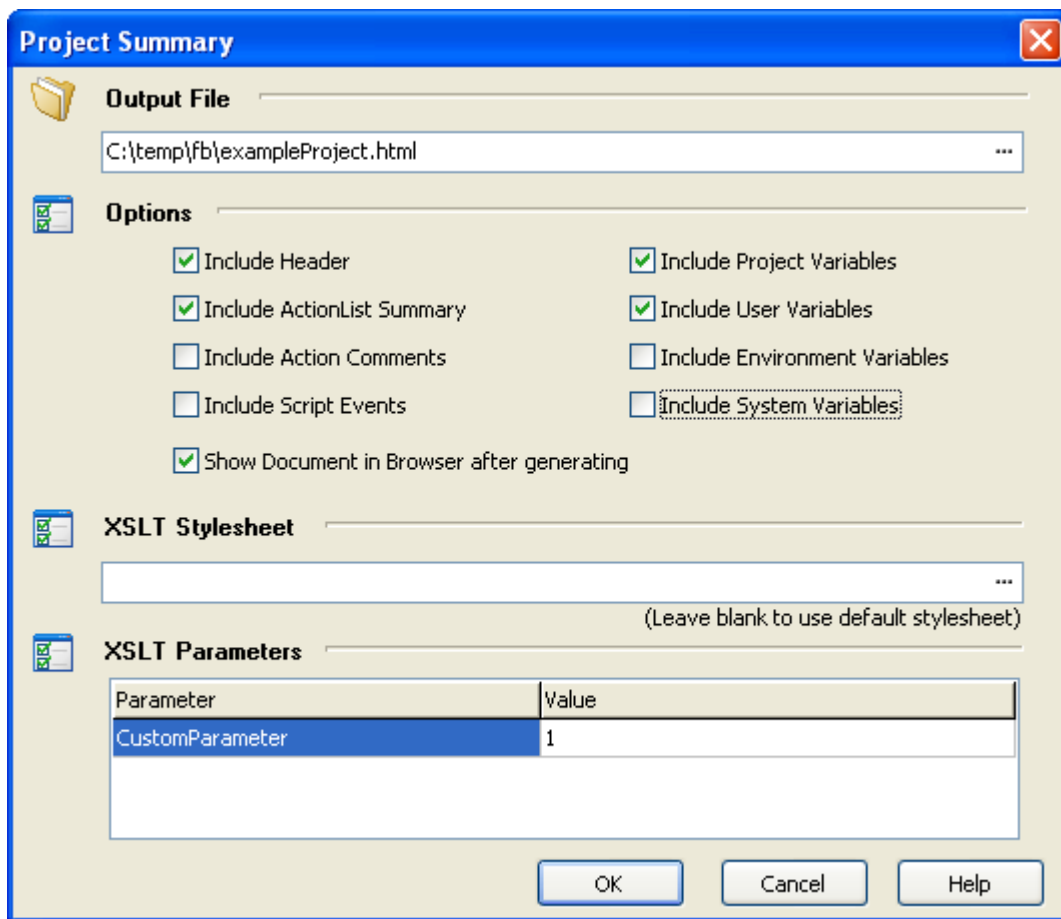
The options dialog allows you to set general FinalBuilder options and preferences, as well as options for some action types.



## 2.8 Project Summary

The project summary produces an HTML Document with an overview of the project.

You can choose which information is output to the summary document.



The XSLT Stylesheet option allows you to use your own custom stylesheet to control the format of the output. If this field is left blank the default stylesheet (fbp2\_2\_html.xml) will be used.

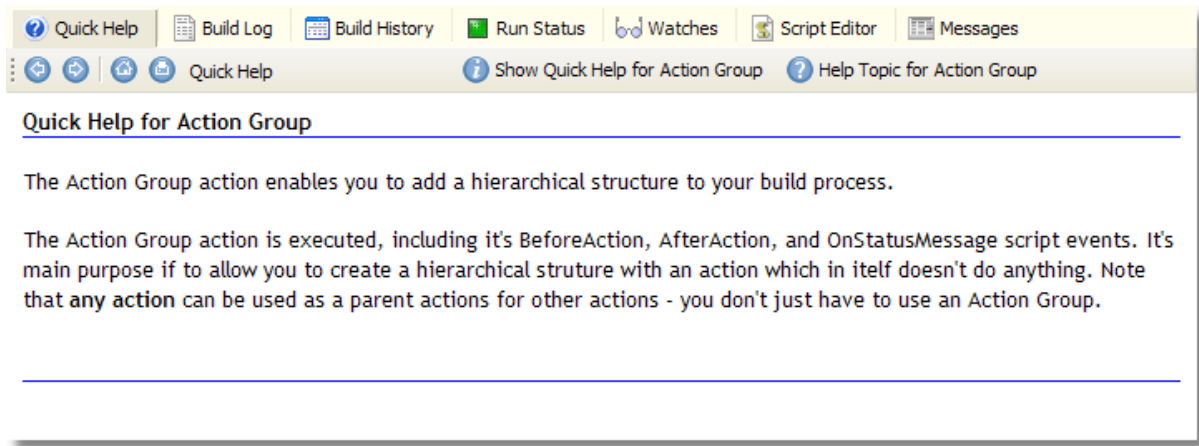
The XSLT Parameters allow you to pass custom parameters to your stylesheet.

## 2.9 Quick Help

The Quick Help page provides a range of functions.

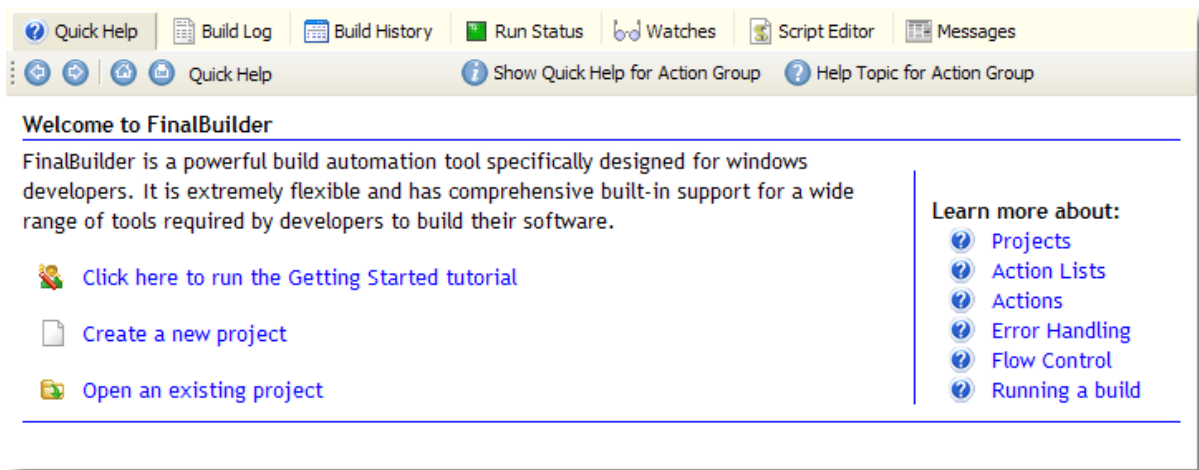
Primarily it can provide some quick help for an action. To view the quick help for an action you can either:

- Select the action from your Action List and:
  - press Alt F1, or
  - right-click and select Quick Help, or
  - click the "Show Quick Help for ..." button on the quick help tab
- Right-Click on an action in Action Types and select "Show Quick Help"



## Getting Started

It's second function is to provide a "Getting Started" tutorial for people to learn the basics of FinalBuilder.



To show the Welcome to FinalBuilder, click the Home button (3rd from the left) in the quick help. This information is also shown by default when FinalBuilder starts.

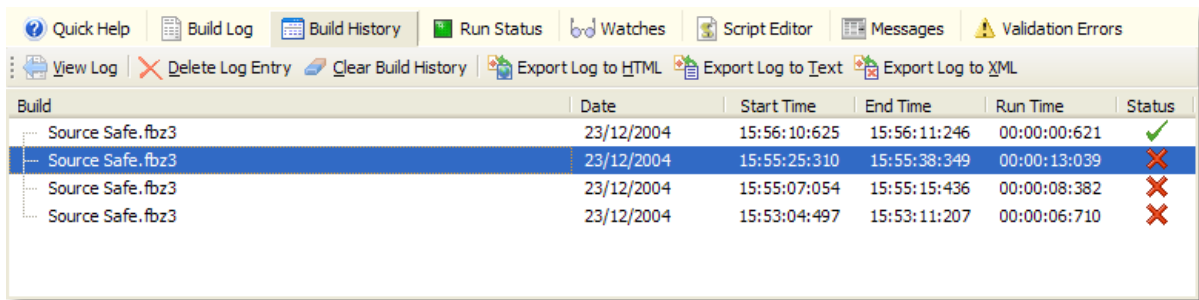
## Check For Updates

The third function of Quick Help is to display information about a new version of FinalBuilder. When FinalBuilder loads, it automatically checks for any updates and displays the information in the Quick Help window. You can manually force a check for update by choosing the "Check for Updates" option in the Help menu. More information in topic Check for Updates

## 2.10 Build History

The Build History displays a summary of the previous builds of the current project.

You can load up a previous build into the Build Log to see the details of that build.



| Build            | Date       | Start Time   | End Time     | Run Time     | Status |
|------------------|------------|--------------|--------------|--------------|--------|
| Source Safe.fbz3 | 23/12/2004 | 15:56:10:625 | 15:56:11:246 | 00:00:00:621 | ✓      |
| Source Safe.fbz3 | 23/12/2004 | 15:55:25:310 | 15:55:38:349 | 00:00:13:039 | ✗      |
| Source Safe.fbz3 | 23/12/2004 | 15:55:07:054 | 15:55:15:436 | 00:00:08:382 | ✗      |
| Source Safe.fbz3 | 23/12/2004 | 15:53:04:497 | 15:53:11:207 | 00:00:06:710 | ✗      |

This shows the last four builds of the Source Safe project.

The available options are:

**View Log** - load build details in the Build Log.

**Delete Log Entry** - removes this build log from the Log History

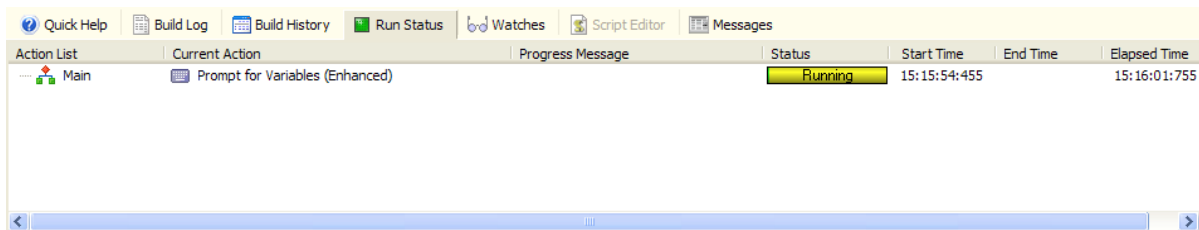
**Clear Build History** - removes all the build logs from the Build History

**Export Log** - Exports the selected build log to HTML, Text, or XML

The Build History only maintains a certain amount of build logs, you can change the setting in Tools->Options.

## 2.11 Run Status

The Run Status provides a summarised view of the current build process.

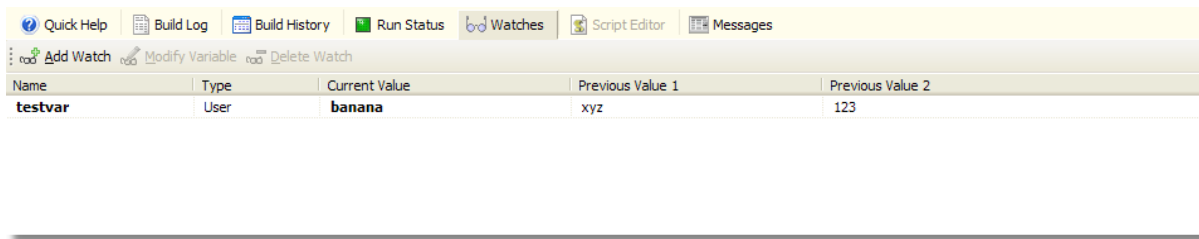


| Action List | Current Action                  | Progress Message | Status  | Start Time   | End Time | Elapsed Time |
|-------------|---------------------------------|------------------|---------|--------------|----------|--------------|
| Main        | Prompt for Variables (Enhanced) |                  | Running | 15:15:54:455 |          | 15:16:01:755 |

## 2.12 Watches

Watches enables you to watch and modify variable values during a build process.

It is designed as a debugging aid so that you can step over actions and see the current and previous values for the specified variables.



## 2.13 Script Editor

The script editor is where you can write VBScript or JScript in response to any events fired at a particular action.

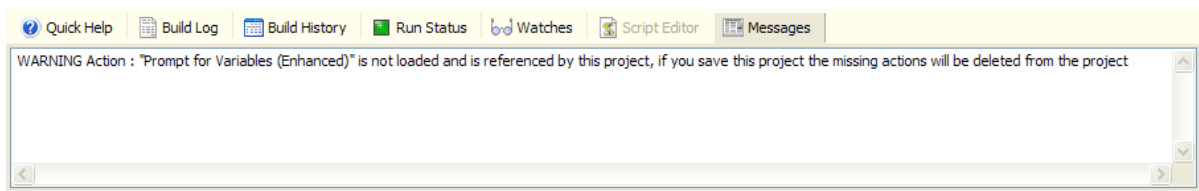
Changing the highlighted action in the action list will change the script editor to show the available events and the script code for the selected event of the current action.

For more information on scripting see Scripting in FinalBuilder

## 2.14 Messages

The Messages tab displays important information requiring your attention.

For example, if a project is loaded and it contains an action which is has not been loaded in the IDE (because the package has been disabled or removed), then it shows an error message such as the one below:

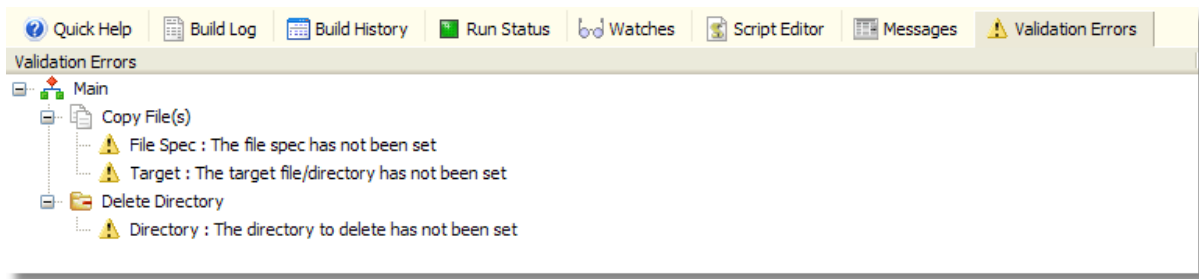


## 2.15 Validation Errors

The validation tab displays any validation errors in the current project or the action just edited.

Validation can be triggered in the following ways:

- When a build starts (see Tools->Options->General Options->Variables and Validation)
- After an action has been edited with the property dialog
- Manually (Project->Validate Project)



Double-Clicking on the validation errors will highlight the action in your project with the validation error.

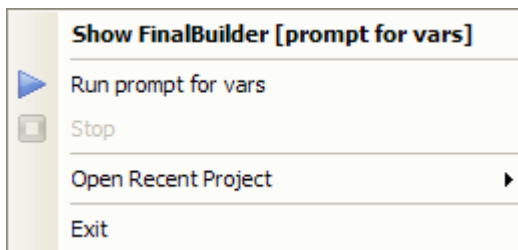
The validation does not prevent the project from being saved, but will prevent the project from being Run. You can turn off validation before a project being run in Tools->Options->General Options->Variables and Validation

## 2.16 Tray Icon

FinalBuilder can optionally show in the System Tray:



The FB Tray Icon provides the following menu on Right-Click:



**Show FinalBuilder [ <project> ]** - This will restore FinalBuilder if it is minimised. The current open project name is shown in brackets.

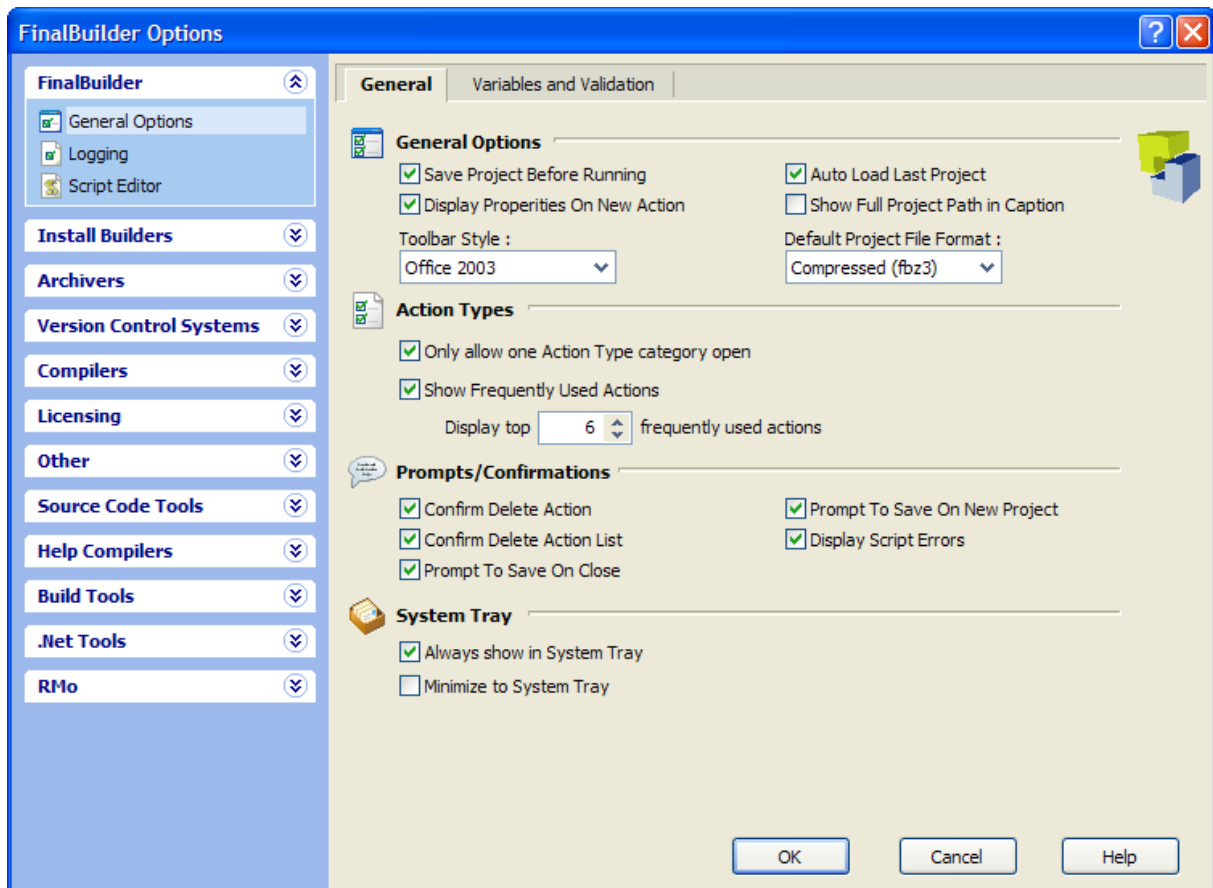
**Run <project>** - This will start the build process.

**Stop** - Stops the build process

**Open Recent Project** - This allows you to open a project from the MRU (Most Recently Used) list of FinalBuilder projects.

**Exit** - Closes FinalBuilder

To change the FB System Tray settings:









### System Tray:

**Always show in System Tray** - The FB Tray icon will show at all times, regardless of the windows state (Minimised, Maximised, etc)

**Minimize to System Tray** - When FB is minimised FB will not show on the task bar or the task manager, but will show on the system tray.

The Tray Icon will also display the state of the build process:

-  - FinalBuilder is currently idle
-  - Build is currently running
-  - Build completed successfully
-  - Build completed with an error
-  - Build failed validation
-  - Build is paused



## 2.17 Check for Updates

Once a month FinalBuilder will prompt the user if they would like to check for any updates by querying the FinalBuilder website. No user information is sent in this process.

If an update is available, the Quick Help tab will display the details of the update and provide a link to any updated files and further instructions.

You can manually check for updates by choosing the "Check for Updates" menu item in the Help menu.

## 2.18 Project Files & Other Files

FinalBuilder can use two different file formats for the project file.

### **<project>.fbz3**

Compressed FinalBuilder project file. This is the default format for a project file. The FBZ3 file is a zipped FBP3 file, so you can use any standard zip tool to uncompress it.

### **<project>.fbp3**

Uncompressed FinalBuilder project file. Project files are standard XML format and so are quite verbose. It is recommended to use the compressed project file format for large projects.

FinalBuilder also maintains other files alongside your project file in the same directory.

### **<project>.fblz**

The FBLZ file is the log archive file. It records the logs of any previous builds, up to the "Log History Count" option (default is 4). If you put the log archive file under version control make sure it is not read only when you run your build's, otherwise the log file won't be written to.

### **<project>.fbw**

The FBW file is the watches file (ie. the variable watches you have defined), you don't need to version this file as it's really only needed when debugging projects.

### **<project>.fbd**

The FBD file is where FinalBuilder stores auto incremented version information (eg. Delphi Compiler, VS.Net Solution compiler, etc.).

## 2.19 FinalBuilder Plugins

FinalBuilder Plugins extend the functionality of FinalBuilder.

For more information on plugins see the integrated help inside FinalBuilder ActionStudio and the ActionStudioManual.pdf in the FinalBuilder directory.

## 3 Scripting

### 3.1 Scripting in FinalBuilder

#### Active Scripting

FinalBuilder Supports Active Scripting using the VBScript and JScript languages that is installed on your machine. Other languages such as PerlScript and Python Script are available from third party vendors.

For VBScript/JavaScript languages support you need to install Microsoft Scripting v3.0 or higher (free product from Microsoft). If you have installed Internet Explorer 3 or higher, you already have v3.0 of Microsoft Scripting but we suggest to download and install latest scripting engine version (5.0 or higher). Windows 2000 and XP already have active scripting installed by default.

All related documentation and VBScript/JavaScript languages descriptions can also be found on Microsoft site

<http://msdn.microsoft.com/scripting/>

#### Debugging

Active scripting based languages can be debugged using the Active Script Debugger, provided the language vendor supports debugging. To enable debugging you will need the Active Script Debugger installed, or Visual Studio.NET (which overrides the script debugger).

#### What Can I do in the Scripts

Basically, anything you can do in any other Active Scripting Host application. Script Events are Triggered Before and After and Action Executes. See Action Script Events for more information on the Events.

#### Using FinalBuilder Variables in Scripts

FinalBuilder variables are available in the script events as global variables. You can reference them just as you would any other identifier.

#### Including External Scripts

You can include external script files in your Action scripts, by inserting a comment with USEUNIT scriptfilename

eg. - VBScript

```
'USEUNIT c:\finalbuilder\scripts\iis_stuff.vbs
```

eg. - JavaScript

```
//USEUNIT c:\finalbuilder\scripts\iis_stuff.js
```

Note that the external script file must be written in the same script language as that of your event

handler script. FinalBuilder provides sample scripts that do things such as restart IIS, shut down COM+ components etc (so that dll's can be updated).

You can also use Finalbuilder variables in the path, for example : 'USEUNIT "%SCRIPTPATH%\Test.vbs"

Note that this is the only time you would use the %variable% syntax for FinalBuilder variables in the script editor, as the useunit line is preprocessed before the script is run.

#### See Also

Global Script Functions | Action Script Events | Accessing TStrings based Properties

## 3.2 Global Script functions

### Global Scripting Functions

Apart from the standard VBScript & JavaScript engine functions, FinalBuilder exposes the following functions :

**procedure** SaveProject;  
Saves the current FinalBuilder project.

**function** ExtractFilePath(value : string) : string;  
Extracts the path (minus the filename) from a fully qualified filename.

**function** ExtractFileName(value : string) : string;  
Extracts the FileName (minus the path) from a fully qualified filename.

**function** ExtractFileDrive(value : string) : string;  
Extracts the filename drive letter

**function** ExtractFileExt(value : string) : string;  
Extracts the file extension including the period

**procedure** FBSetCaption(value : string);  
Sets the titlebar caption for FinalBuilder  
**FBSetCaption is a no-op when Finalbuilder is run from the scheduler.**

**function** GetClipboardText : string;  
Gets the text currently on the clipboard  
**GetClipboardText is a no-op when Finalbuilder is run from the scheduler.**

**procedure** CopyToClipboard(const value : string)  
Copies the string to the clipboard  
**CopyToClipboard is a no-op when Finalbuilder is run from the scheduler.**

**function** FBFormatDateTime(format : string; value : DateTime) : string;  
Formats the specified DateTime as a string  
See Format DateTime Formatting Options

**function** StrToDate(value : string) : DateTime;  
Converts a string into a Date

**function** StrToDateTime(value : string) : DateTime;  
Converts a string into a DateTime

**function** ChangeFileExt(filename : string; newext : string) : string;  
Changes the file extension of filename to the specified new extension.  
eg. ChangeFileExt("c:\temp\test.txt", ".doc") = "c:\temp\test.doc"

**function** IncludeTrailingPathDelimiter(value : string) : string;  
Appends a trailing path delimiter to the specified directory if required.

**function** ExcludeTrailingPathDelimiter(value : string) : string;  
Removes a trailing path delimiter from the specified directory if it exists.

**function** ExpandFileName(value : string) : string;  
Expands the short filename and path to the full filename/path

**function** FileExists(value : string) : boolean;  
Returns true if the specified file exists

**function** GetCurrentDir : string;  
Returns the current working directory

**function** SetCurrentDir(value : string) : boolean;  
Set the current working directory

**function** ExpandRelativePath(filepath : string; relativeto : string) : string;  
Returns the full path and filename of the file specified with the relative path.  
eg. ExpandRelativePath("../..\Source", "myfile.txt") = "c:\Dev\Source\myfile.txt"

**function** NewGUIDString : string;  
Creates a new GUID (Globally Unique Identifier)

**function** ExtractMajorVer(value : string) : string;  
Extracts the Major Version value from a version string  
eg. ExtractMajorVer("3.0.23.1") = "3"

**function** ExtractMinorVer(value : string) : string;  
Extracts the Minor Version value from a version string  
eg. ExtractMajorVer("3.0.23.1") = "0"

**function** ExtractReleaseVer(value : string) : string;  
Extracts the Release Version value from a version string  
eg. ExtractMajorVer("3.0.23.1") = "23"

**function** ExtractBuildVer(value : string) : string;  
Extracts the Build Version value from a version string  
eg. ExtractMajorVer("3.0.23.1") = "1"

**function** MessageBox(text : string; title : string; style : integer) : integer;  
Displays a message box to the user.  
See MessageBox Constants

**procedure** alert(text : string);

Displays an alert dialog to the user.

### 3.2.1 Format DateTime Formatting Options

The valid format specifiers for the **FBFormatDateTime** script function are

| <b>Specifier</b> | <b>Displays</b> |
|------------------|-----------------|
|------------------|-----------------|

|        |  |
|--------|--|
| c      | Displays the date using the format given by the ShortDateFormat global variable, followed by the time using the format given by the LongTimeFormat global variable. The time is not displayed if the date-time value indicates midnight precisely. |
| d      | Displays the day as a number without a leading zero (1-31).  |
| dd     | Displays the day as a number with a leading zero (01-31).  |
| ddd    | Displays the day as an abbreviation (Sun-Sat) using the strings given by the ShortDayNames global variable.  |
| dddd   | Displays the day as a full name (Sunday-Saturday) using the strings given by the LongDayNames global variable.   |
| dddddd | Displays the date using the format given by the ShortDateFormat global variable.   |
| dddddd | Displays the date using the format given by the LongDateFormat global variable.  |
| e      | Displays the year in the current period/era as a number without a leading zero (Japanese, Korean and Taiwanese locales only).  |
| ee     | Displays the year in the current period/era as a number with a leading zero (Japanese, Korean and Taiwanese locales only).   |
| g      | Displays the period/era as an abbreviation (Japanese and Taiwanese locales only).  |
| gg     | Displays the period/era as a full name. (Japanese and Taiwanese locales only).   |
| m      | Displays the month as a number without a leading zero (1-12). If the m specifier immediately follows an h or hh specifier, the minute rather than the month is displayed.  |
| mm     | Displays the month as a number with a leading zero (01-12). If the mm specifier immediately follows an h or hh specifier, the minute rather than the month is displayed.   |
| mmm    | Displays the month as an abbreviation (Jan-Dec) using the strings given by the ShortMonthNames global variable.  |
| mmmm   | Displays the month as a full name (January-December) using the strings given by the LongMonthNames global variable.  |
| yy     | Displays the year as a two-digit number (00-99).   |
| yyyy   | Displays the year as a four-digit number (0000-9999).  |
| h      | Displays the hour without a leading zero (0-23).   |
| hh     | Displays the hour with a leading zero (00-23).   |
| n      | Displays the minute without a leading zero (0-59).   |
| nn     | Displays the minute with a leading zero (00-59).   |
| s      | Displays the second without a leading zero (0-59).   |
| ss     | Displays the second with a leading zero (00-59).   |
| z      | Displays the millisecond without a leading zero (0-999).   |
| zzz    | Displays the millisecond with a leading zero (000-999).  |
| t      | Displays the time using the format given by the ShortTimeFormat global variable.   |
| tt     | Displays the time using the format given by the LongTimeFormat global variable.  |
| am/pm  | Uses the 12-hour clock for the preceding h or hh specifier, and displays 'am' for any hour before noon, and 'pm' for any hour after noon. The am/pm specifier can use lower, upper, or mixed case, and the result is displayed accordingly.        |
| a/p    | Uses the 12-hour clock for the preceding h or hh specifier, and displays 'a' for any hour before noon, and 'p' for any hour after noon. The a/p specifier can use lower, upper,  |

or mixed case, and the result is displayed accordingly.

**ampm** Uses the 12-hour clock for the preceding h or hh specifier, and displays the contents of the TimeAMString global variable for any hour before noon, and the contents of the TimePMString global variable for any hour after noon.

**/** Displays the date separator character given by the DateSeparator global variable.

**:** Displays the time separator character given by the TimeSeparator global variable.

**'xx'/"xx"** Characters enclosed in single or double quotes are displayed as-is, and do not affect formatting.

Example usages of the FBFormatDateTime function (in VBScript)

```
dim s
```

```
s = FBFormatDateTime("ddmmyyyy",Now) ' returns current date in this format
01012003
```

```
s = FBFormatDateTime("c",Now) ' returns the current date in the system short
date format, eg. 01/12/2003
```

### 3.2.2 MessageBox Constants

**function** MessageBox(text : string; title : string; style : integer) : integer;

Displays a message box to the user.

The valid values for the style are:

```
mbOK
mbOKCANCEL
mbABORTRETRYIGNORE
mbYESNOCANCEL
mbRETRYCANCEL
mbICONHAND
mbICONQUESTION
mbICONEXCLAMATION
mbICONASTERISK
mbDEFBUTTON1
mbDEFBUTTON2
mbDEFBUTTON3
mbDEFBUTTON4
```

The MessageBox function return value corresponds to the button which the user pressed:

```
mrOK
mrCANCEL
mrABORT
mrRETRY
mrIGNORE
mrYES
mrNO
mrCLOSE
mrHELP
```

eg.

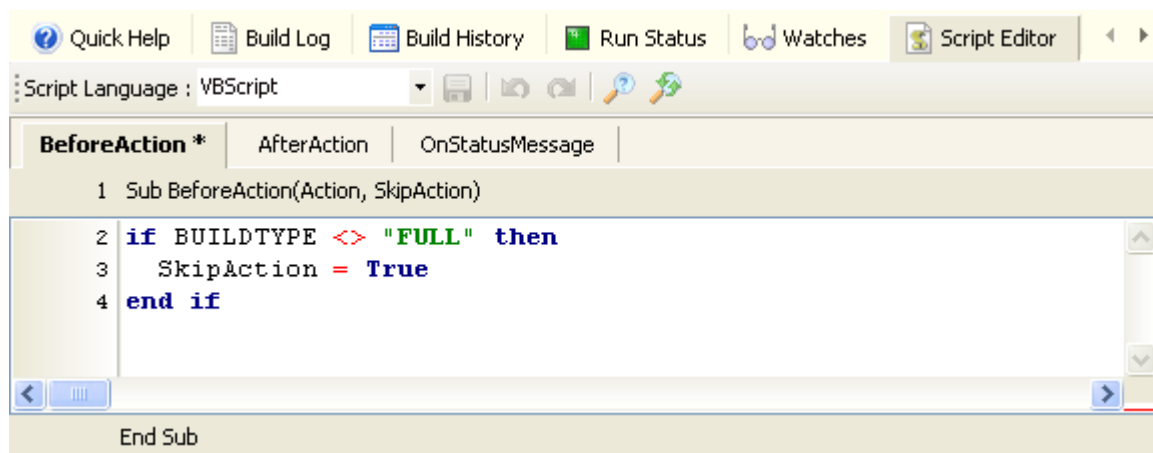
```
if MessageBox("Do you want to cancel the build", "Cancel Build", mbOKCANCEL) = mrOK
then
```

```
// build needs to stop  
end if
```

### 3.3 Action Script Events

Each Action in FinalBuilder has a **BeforeAction**, **AfterAction**, and **OnStatusMessage** Event Script; some Actions define more scripting events. In addition, each action has a Condition property which is a Boolean expression which should return true for the action to execute. See Action Reference for more information. The Condition property is displayed in the Property dialog for the action, in the Properties tab (Execute Condition section).

The action Script Events are shown at the bottom of the FinalBuilder IDE in the Script Editor tab, see screenshot.



The following script events are common to all actions:

#### **BeforeAction event**

Description: called before the action is executed.

Parameters:

- **Action** parameter allows access to the action properties and methods (see Action Properties and Methods).
- **SkipAction** parameter allows the script to return true for the action to be skipped during a build process.

#### **AfterAction event**

Description: called after the action has executed.

Parameters:

- **Action** parameter allows access to the action properties and methods (see Action Properties and Methods).
- **ActionResult** parameter - indicates if the action succeeded or failed. This may also be set to override the action status.
- **Continue** parameter - return false to stop the build, return true to continue the build ignoring the ActionResult

#### **OnStatusMessage Event**

Description: called whenever the action generates a log message

Parameters:

- **Action** parameter allows access to the action properties and methods (see Action Properties and Methods).

- **StatusMessage** - the status message object contains the information of the status message (Lines, MessageText, MessageTitle, and Progress)

### 3.3.1 Action Properties and Methods

#### Action Properties and Methods

The Action object is passed into Script Events. The methods and properties available on the Action object are as follows:

#### Procedure/Methods

- procedure **SetLogTitle**(value As String)
- procedure **SendLogMessage**(MessageText As String)
- procedure **Echo**(MessageText As String)
- procedure **SendProgress**(Value As String, progress As Integer)

#### Functions

- function **ExpandExpression**(expr As String) As String
- function **ChildActions**(index As Integer) As OleVariant
- function **Parent** As OleVariant

#### Properties

- **ActionComment** As String
- **ActionLogTitle** As String
- **ActionName** As String
- **ChildActionCount** As Integer
- **Description** As String
- **IgnoreFailure** As Boolean
- **LogToVariable** As String
- **PauseInterval** As Cardinal
- **SuppressStatusMessages** As Boolean

Detailed information on the procedures, functions and properties:

procedure **SetLogTitle**(value As String)

Allows you to set the title of the FinalBuilder log.

procedure **SendLogMessage**(MessageText As String)

Sends a message to the output window. Note that this will also trigger the OnStatusMessage event (except when called from within the OnStatusMessage event!)

procedure **Echo**(MessageText As String)

Same as SendLogMessage (see above).

procedure **SendProgress**(Value As String, progress As Integer)

Sends a progress message to the Run Status window. This enables the action to report it's progress as it's executing. Progress is a percentage and should therefore range between 0 and 100.

function **ExpandExpression**(expr As String) As String

This expands the string passed in by substituting FinalBuilder variables when %<variable>% is encountered.

function **ChildActions**(index As Integer) As OleVariant

Allows access to child actions, index is zero based. Use ChildActionCount property to get a count of



the child actions.

Example : Turn off debug info on all child Delphi compiler actions... (in this example IncludeDebugInfo is a FinalBuilder variable) add this code to the BeforeAction event handler of the parent action (an action group for example).

```
dim child
dim count
dim i
```

```
count = Action.ChildActionCount - 1
```

```
for i = 0 to count
  set child = Action.ChildActions(i)
  if not (child is nothing) then
    if child.ActionName = "Compile Delphi Win32 Project" then
      child.CompilerOpt.DebugInfo = IncludeDebugInfo
    end if
  end if
next
```

**function Parent As OleVariant**

Returns the Parent action of the current action. This will return null for root level actions.

**property: ActionComment As String**

Provides access to the action comment.

**property: ActionLogTitle As String**

Allows you to change the title of the action's entry in the output window. Note that this has no effect if set from the AfterAction event (since the log entry has already been made.)

eg.

```
Action.LogTitle = "Full Build - VB6"
Action.SendLogMessage("File deleted")
```

**property: ActionName As String**

Provides access to the Action's name

**property: ChildActionCount As Integer**

Returns the number of Child Actions the action has.

**property: Description As String**

Provides access to the Action's description

**property: IgnoreFailure As Boolean**

Provides access to the Action's IgnoreFailure flag

**property: LogToVariable As String**

Provides access to the Action's LogToVariable flag

**property: PauseInterval As Cardinal**

Provides access to the Action's PauseInterval property

**property: SuppressStatusMessages As Boolean**

Provides access to the Action's SuppressStatusMessages property

## 3.4 Accessing the Options settings via scripting

FinalBuilder allows you to modify FinalBuilder options at runtime using scripting. This makes it possible to specify the path to a third party tool at runtime so that a FinalBuilder project can for example use a different version of a third party tool. Note that these modified settings are not saved and will be lost when you close FinalBuilder.

The options objects are accessible using the `GetOptionsObject` script function. This function takes the name of the options object as a parameter. The name is the same as appears in the Options dialog. For example to access the Delphi options you would use this code (VBScript example) :

```
dim delphiOptions
set delphiOptions = GetOptionsObject("Borland Delphi")
delphiOptions.D6LibraryPath = delphiOptions.D6LibraryPath &
";$(DELPHI)\Components\Lib"
```

Listed below are the available properties on the options objects.

### FinalBuilder General Options

Name : General Options

```
property PromptOnClose : boolean
property PromptOnNew : boolean
property DisplayPropDialog : boolean
property SaveBeforeRun : Boolean
property ShowTree : Boolean
property AutoLoadLastProject : Boolean
property PromptOnDeleteAction: boolean
property AutoSizeListColumns : boolean
property DisplayScriptErrors : Boolean
```

### FinalBuilder Script Editor

Name : Script Editor

```
property LineNumbers: boolean
property DefaultScriptLanguage : String
property AutoSaveScripts : boolean
```

### FinalBuilder Logging Options

Name : Logging

```
property LogHistoryCount : integer
property IncludeActionOutput : boolean
property OnlyIncludeErrorAction : boolean
property ConfirmDeleteRunLog : boolean
property ConfirmDeleteLogFile : boolean
```

### Delphi Options

Name : Borland Delphi

property D3LibraryPath : string  
property D4LibraryPath : string  
property D5LibraryPath : string  
property D6LibraryPath : string  
property BCB3LibraryPath : string  
property BCB4LibraryPath : string  
property BCB5LibraryPath : string  
property BCB6LibraryPath : string  
property UseRegForLibraryPath : Boolean

### **Inno Setup Options**

Name : Inno Setup

property InnoDLL : string

### **Help & Manual 3 Options**

Name : Help & Manual

property HM3Location: string;

### **Wise Installer Options**

Name : Wise Installer

property WiseInstaller : string  
property WiseWindowsInstaller : string

### **Doc-O-Matic Options**

Name : Doc-O-Matic

property DocomaticExe: string

### **Help Compiler Options**

Name : Microsoft Help

property HCRTFLocation : string  
property HHCLocation : string

### **InstallShield Options**

Category : Install Builders  
Name : Installshield Pro

property CompilerLocation : string  
property IFXLocation : string  
property ISRTLLocation : string  
property IncludePath : string

```
property ISBuildLocation : string
property ISCmdBldLocation : string
```

### **GPInstall Options**

```
Name          : GP-Install

property GPBuilderLocation: string
```

### **ASProtect Options**

```
Name          : ASProtect

property ASProtectLocation: string
```

### **Armadillo Options**

```
Name          : Armadillo

property ArmadilloLocation:string
```

### **InternetOptions**

```
Name          : Internet

property FTPServer : string;
property FTPUser   : string;
property FTPPass   : string;
property FTPPort   : integer ;

property SMTPHost  : string;
property SMTPPort  : integer;
property SMTPAuth  : boolean;
property SMTPEHLO  : boolean;
property SMTPUser  : string;
property SMTPPass  : string;
```

## **3.5 Accessing TStrings based properties**

You will note that some Action properties are of type TStrings. TStrings is a standard Delphi type that we have exposed to the scripting engine. It is basically a string collection, below are the properties of TStrings that you will find useful:

```
procedure Clear;
function Add(const S: string): Integer;
procedure Delete(Index: Integer);
procedure Insert(Index: Integer; const S: string);
function IndexOf(const S: string): Integer;
property Strings[Index: Integer]: string; //index is zero based
property Count: Integer;
property Text: string; //returns the entries in the collection, each entry on a new line
property Names[Index: Integer]: string; // used to access the name of a string
```

which is of the format <name>=<value>

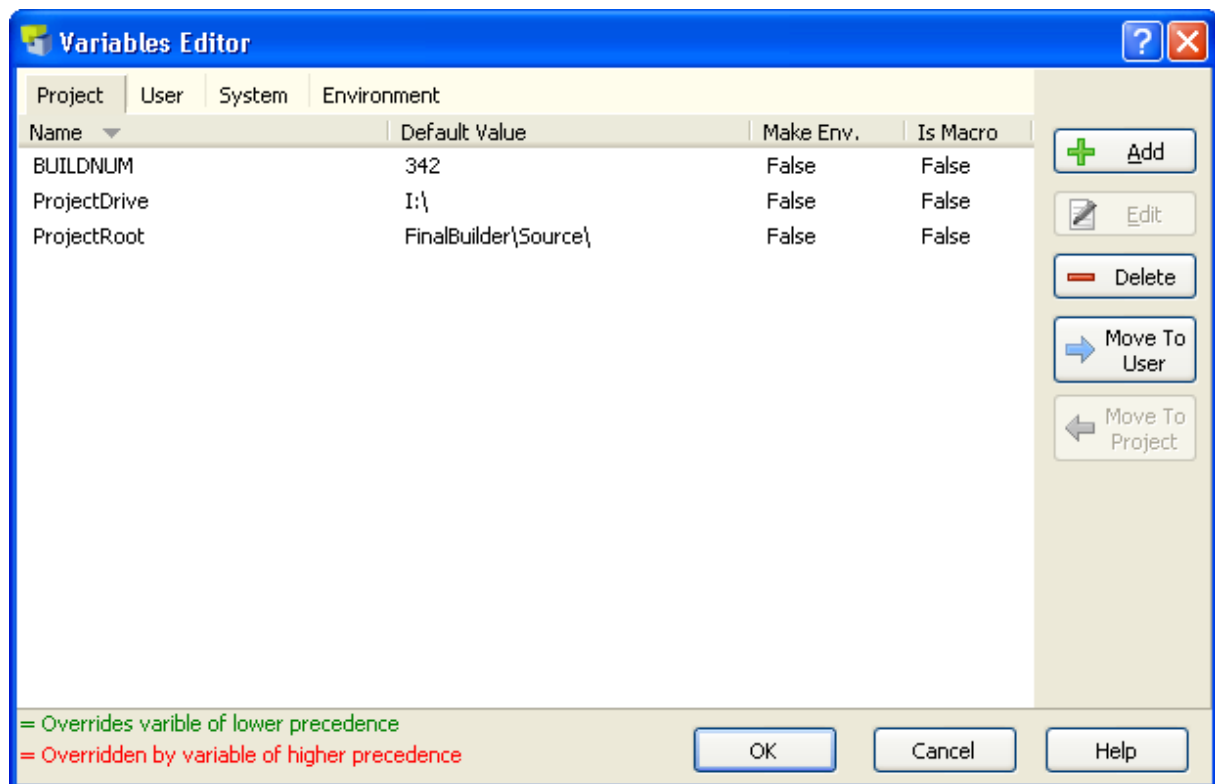
**property** Values[Index: String]: string; // used to access the value of a string which is of the format <name>=<value>

**property** ValueFromIndex[Index: Integer] : string; // used to access the value of a string which is of the format <name>=<value>

## 4 Variables

### 4.1 Variables Overview

Variables in FinalBuilder are the key to making your builds dynamic. Variables can be used in nearly every text property of the actions, for example in fields that specify file paths, directories etc. Variables can be modified at run time by either using the Set Variable Action in the Active Scripting events. Variables defined in FinalBuilder can be referenced in the active scripts just a normal (eg. VBScript or JScript) variables. Only Project and User variables can be modified.



#### Make Env.

Project and User Variables can be made available as environment variables to applications that are executed by FinalBuilder.

#### Is Macro

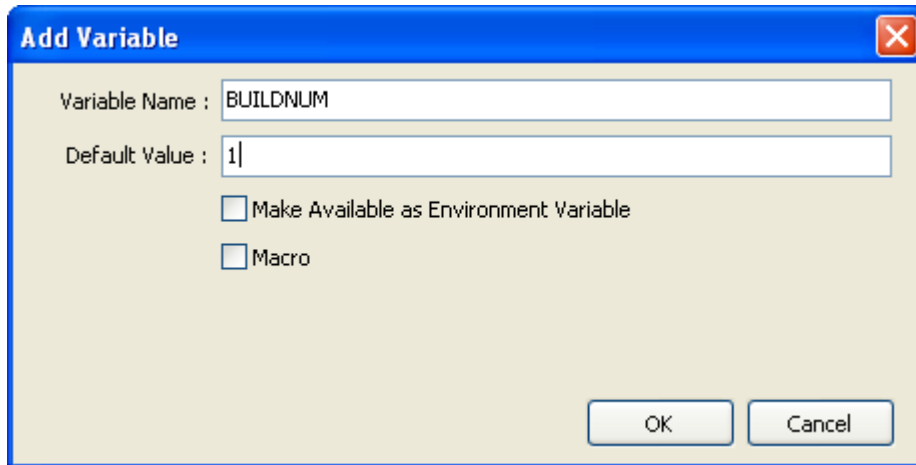
The Is Macro flag forces FinalBuilder to re-evaluate the variable whenever it's referenced during the build, otherwise the value of the variable is evaluated when the build starts and that value is used throughout the build. An Is Macro variable is like a function - it's value cannot be set during the build process using the Set Variable action or any other means.

#### See Also

Using Variables  
Project Variables  
User Variables  
Environment Variables  
System Variables  
Action List Parameters

## 4.2 Adding Variables

When adding or editing a variable, the following dialog is displayed:



### Variable Name

The name by which you will refer to the variable in your build process.  
In an action you reference the variable as follows, eg. %BUILDNUM%  
In script you reference the variable by name, without the % signs.

### Default Value

The variable will be set to this default value when a build is started

### Make Available as Environment Variable

Project and User Variables can be made available as environment variables to applications that are executed by FinalBuilder.

### Macro

This will treat the variable as a "macro" or "function". Its value will be re-evaluated whenever it is referenced and it cannot be set

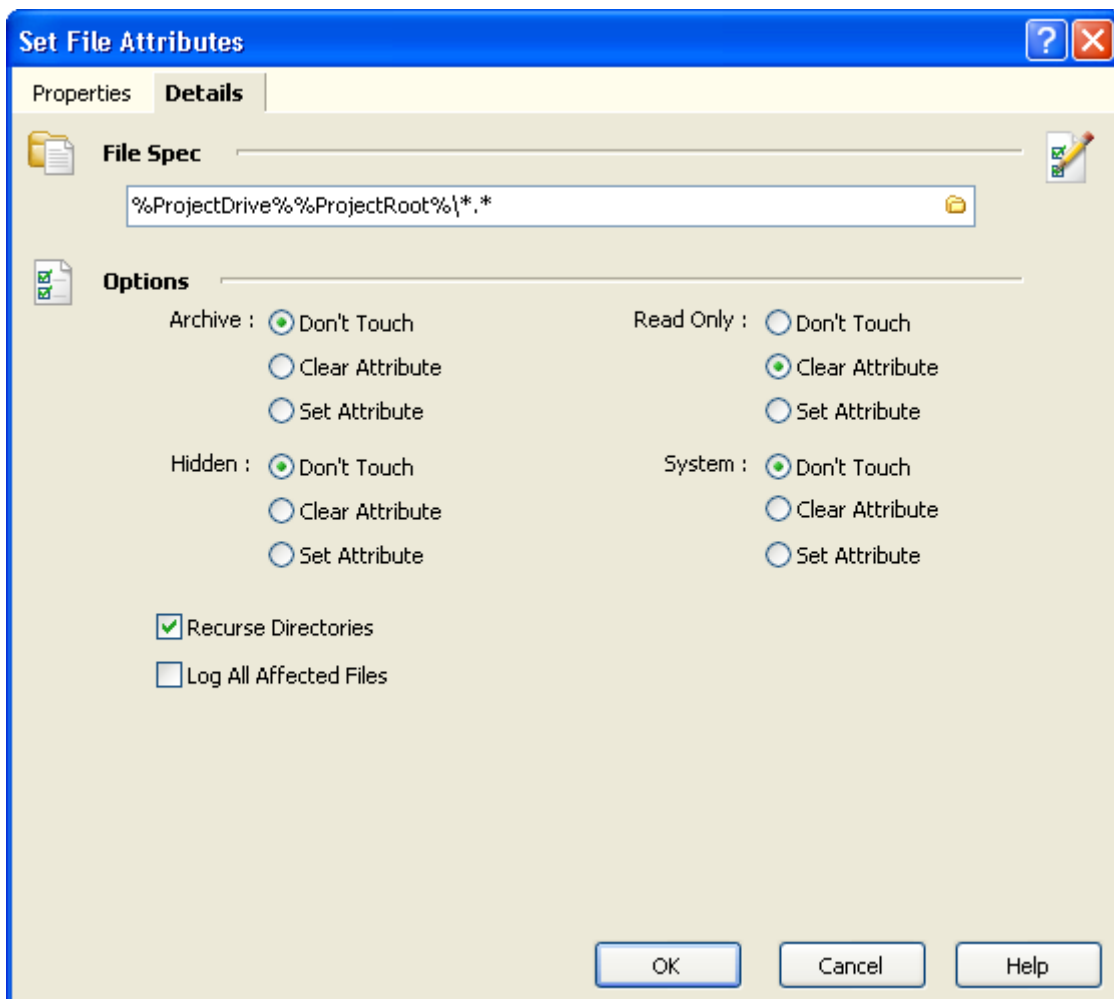
## 4.3 Using Variables

Variables can be used in most text properties for Actions. To use a Variable in a property, enclose the variable in percent symbols,

eg. : %BUILDOUTPUTDIR%

If you need to use a % symbol in a property, make sure you escape it with another one, eg. %%

A common use for variables is to set file paths, for example to set the File Spec for Set File Attributes:



## 4.4 Project Variables

Project Variables are stored in the project file. This makes them available on any machine where the project is run.

## 4.5 User Variables

User Variables are stored in an INI file in the following location:

`\Documents and Settings\\Application Data\FinalBuilder3\FBUserVariables.ini`

They are useful for defining variables that will be used in multiple projects but are specific to a user.

## 4.6 Environment Variables

FinalBuilder reads in the systems Environment variables when it starts. These can be used in the same manner as Project and User Variables. Setting an Environment variable can only be done during a run, either with the Set Variable Action or in the Action Script events . The values assigned to Environment variables by FinalBuilder during a run are not saved after a run. Finalbuilder sets them temporarily and then restores them to their original values at the end of the run.

Environment variables in FinalBuilder are also made available to applications that it executes.

**Note:** If you modify an Environment variable outside of FinalBuilder, FinalBuilder will not see this change until you restart FinalBuilder.

## 4.7 System Variables

System Variables are defined by FinalBuilder at startup. The actual variables available will depend on which software you have installed on your machine. System variables cannot be modified.

These System Variables are available on all machines :

|                 |   |
|-----------------|---|
| SYSDIR          | The Windows System Directory, eg. 'D:\WINNT\System32'                             |
| WINDIR          | The Windows Directory - eg. 'D:\WINNT'  |
| COMPUTERNAME    | The name of the computer you are running FinalBuilder on.                         |
| USERNAME        | The Name of the currently logged on User  |
| DOSCMD          | The path to the Dos command interpreter. This is used internally by FinalBuilder. |
| ISAUTOBUILD     | True if the build is being run automatically from the command line.               |
| FBPROJECT       | The full path to the currently running FinalBuilder project file.                 |
| FBPROJECTDIR    | The dir where the currently running FinalBuilder project lives.                   |
| FBDIR           | The directory in which FinalBuilder is installed.                                 |
| FBIGNOREDERRORS | The number of actions in error during the build that were ignored.                |

These Variables may appear on your system, depending on whether you have the software installed :



|                   |  |
|-------------------|--|
| VB6               | The path to the Visual Basic 6 compiler                            |
| DELPHI3DIR        | The Directory where Delphi 3 is installed                          |
| DELPHI3           | The path to the Delphi 3 command line compiler                     |
| DELPHI3_BRCC32    | The path to the Delphi 3 Resource compiler                         |
| DELPHI4DIR        | The Directory where Delphi 4 is installed                          |
| DELPHI4           | The path to the Delphi 4 command line compiler                     |
| DELPHI4_BRCC32    | The path to the Delphi 4 Resource compiler                         |
| DELPHI5DIR        | The Directory where Delphi 5 is installed                          |
| DELPHI5           | The path to the Delphi 5 command line compiler                     |
| DELPHI5_BRCC32    | The path to the Delphi 5 Resource compiler                         |
| DELPHI6DIR        | The Directory where Delphi 6 is installed                          |
| DELPHI6           | The path to the Delphi 6 command line compiler                     |
| DELPHI6_BRCC32    | The path to the Delphi 6 Resource compiler                         |
| DELPHI7DIR        | The Directory where Delphi 7 is installed                          |
| DELPHI7           | The path to the Delphi 7 command line compiler                     |
| DELPHI7_BRCC32    | The path to the Delphi 7 Resource compiler                         |
| DELPHI2005        | The path to the Delphi 2005 command line compiler                  |
| DELPHI2005DIR     | The Directory where Delphi 2005 is installed                       |
| DELPHI2005_BRCC32 | The path to the Delphi 2005 Resource compiler                      |
| DELPHI2005DOTNET  | The path to the Delphi 2005 .Net command line compiler (dccil.exe) |

## 4.8 Action List Parameters

Action List Parameters are designed to allow you to call an action list with a different set parameters.

The scope of action list parameters is confined to the actions within the Action List which defines the parameters. The syntax to access an action list parameters is #(<actionlistparameter>) eg. #(BUILDVER)

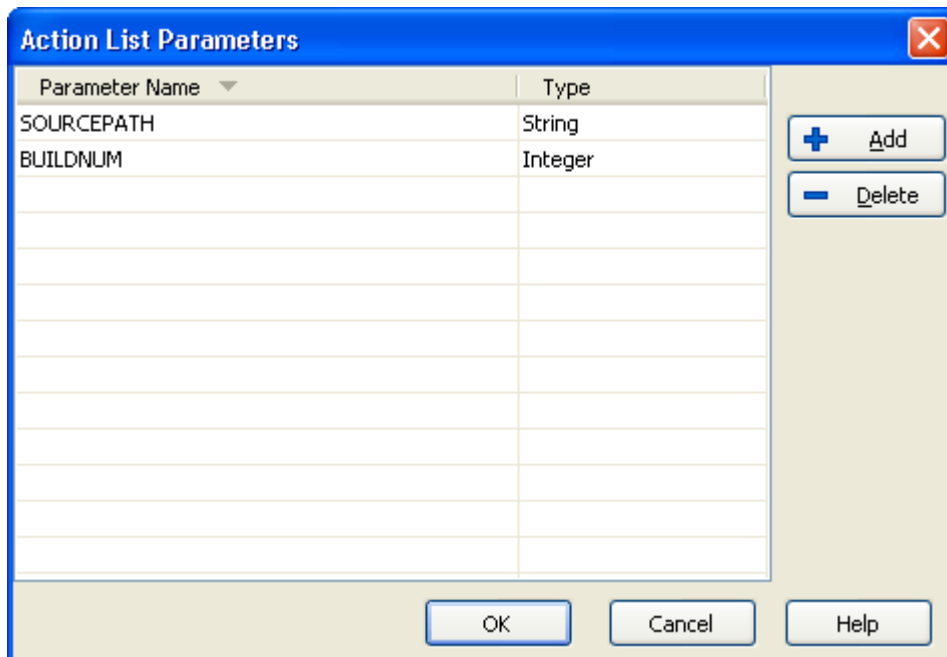
Action List Parameters cannot be modified; they are initialised in the Run Action List action before the action list is executed.

Action List Parameters can be accessed via script and will take precedence over a FinalBuilder variable of the same name. The syntax to access the action list parameter from script is %<parameter>%, eg. %buildver%

Action List Parameters are available on any action list except Main and OnFailure.

To view or modify action list parameters you can either:

1. Right-click on the action list tab and select "Action List Parameters...", or
2. Select the "Action List Parameters..." menu item from the Project menu.

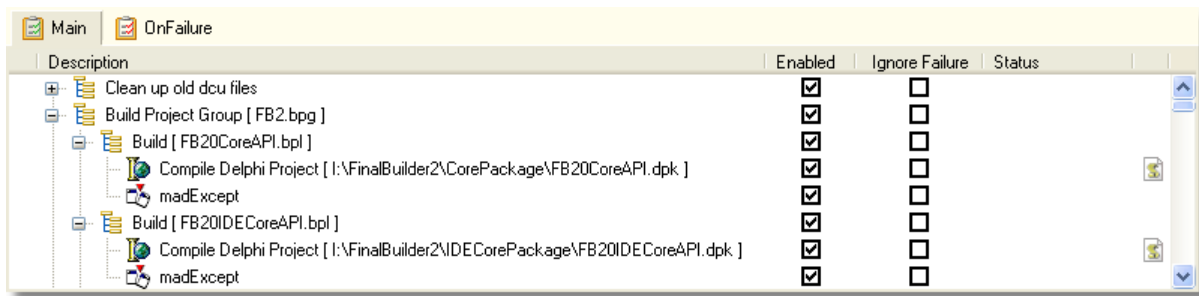


In the above example, the variables #(SOURCEPATH) and #(BUILDNUM) are available to any actions in the action list with these parameters.

To run an action list, use a "Run Action List" action (available in the "Flow Control" action types category).

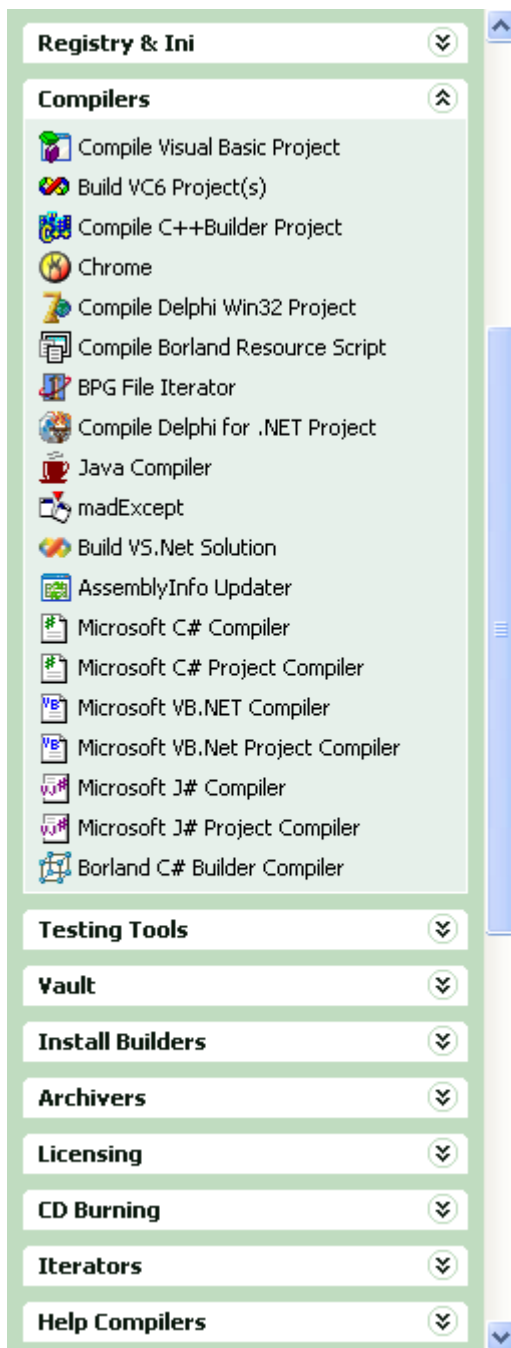
The Run Action List action allows you to specify the value of the action list parameters:





### Using Actions

To add an action to the action list, select an action from the Action Types List (on the left hand side of the main window) by clicking on the action type you wish to add.



After clicking on the action type, by default the properties dialog will be displayed :

**Action Group**

**Properties**

**Action Description**

Compile Source Code

Comment : All the compilation are done here.

**Options**

☒ Action Enabled ☐ Ignore Failure

Pause after Run : 0 ms

**Logging Options**

☐ Suppress Log Messages ☐ Log to Variable

**Execute Condition**

Script Language : VBScript

Condition must return a boolean value (True or False)  
Condition syntax defined by script language

Condition : DOCOMPILE

**Copyright**

FinalBuilder - Copyright © 2000-2004 VSoft Technologies Pty Ltd

OK Cancel Help

**Action Description** - Give the action a meaningful but short description. This is displayed in the build script.

**Comment** - Add a comment for your own documentation purposes

**Action Enabled** - If not checked then the action will not be run under any circumstances during your build. This also affects any child actions.

**Ignore Failure** - If not checked, then the build process continues onto the next action and a failure condition is not raised. You can use "If Prev Action Failed" action immediately after an action with Ignore Failure turned on to detect if the action did fail.

**Pause After Run** - The build process will wait for the specified period of time after this action completes before continuing onto the next action.

**Suppress Log Messages** - No log messages from the action will be displayed or logged.

**Log To Variable** - All log messages will be copied into the specified variable to allow you to process the log messages further.

**Execute Condition** - The condition field is evaluated before running the action. If the condition evaluates to True or is empty then the action will execute, otherwise it will not. To use FinalBuilder

variables in the condition, you do not need to specify the %'s. eg. if you have a FB variable "DOCOMPILE" then you simply specify DOCOMPILE and not %DOCOMPILE%. A previous action should set the DOCOMPILE variable to either True or False depending on certain conditions.

This is another example of how to use the condition property:

```
(VBScript)  
BuildType = "Full"
```

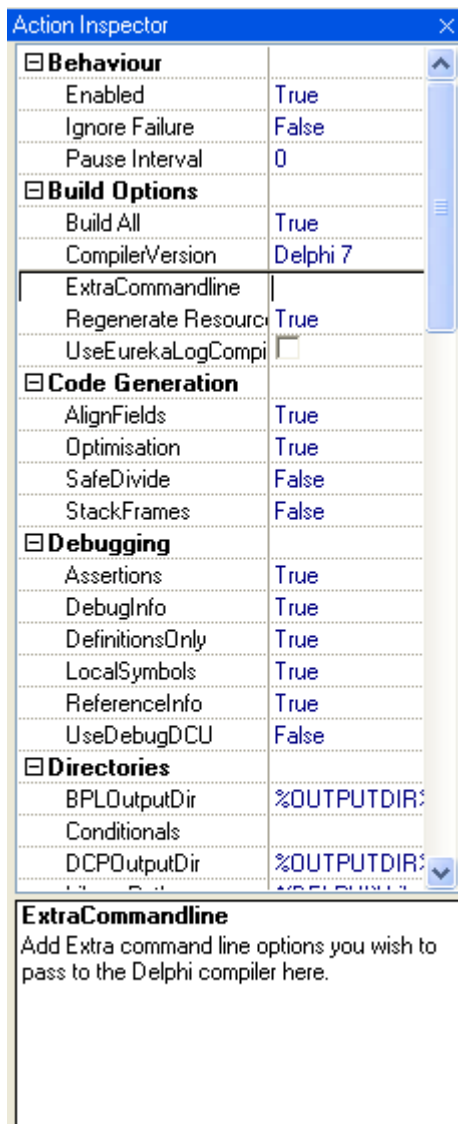
```
(JScript)  
BuildType == "Full"
```

where BuildType is a FinalBuilder variable.

**Copyright** - Displays any copyright information for the particular action.

The number of tabs in this dialog will depend on which action type you selected.

In addition to the properties window, you can also edit some of the properties in the Action Inspector.



## 5.1 Flow Control

### 5.1.1 Delay Action

This Action will pause for the time specified in the Delay property (ms).

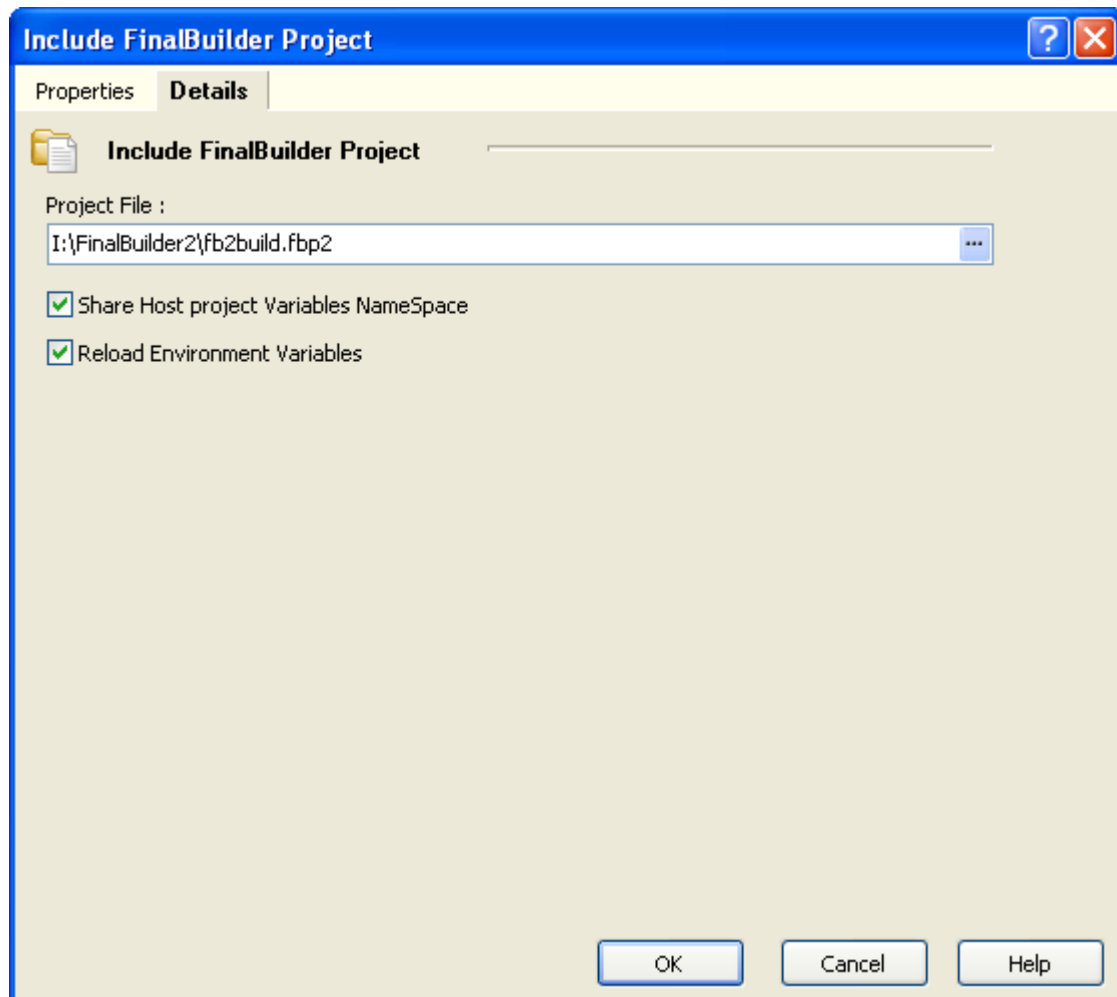


### 5.1.2 Include FinalBuilder Project

This action type allows you to include other FinalBuilder projects in your build process. This allows you to modularise your build process.

Note that this Action should be used with care. If "Share Host project Variables Namespace" is checked, then the host project and the Included project will share one variable namespace. When the included project is executed, any project variables are loaded at that time, if there are variable name clashes then the existing project variable will be kept. This can lead to unexpected results. When this option is not enabled, the host and the included projects each have their own variable namespace.

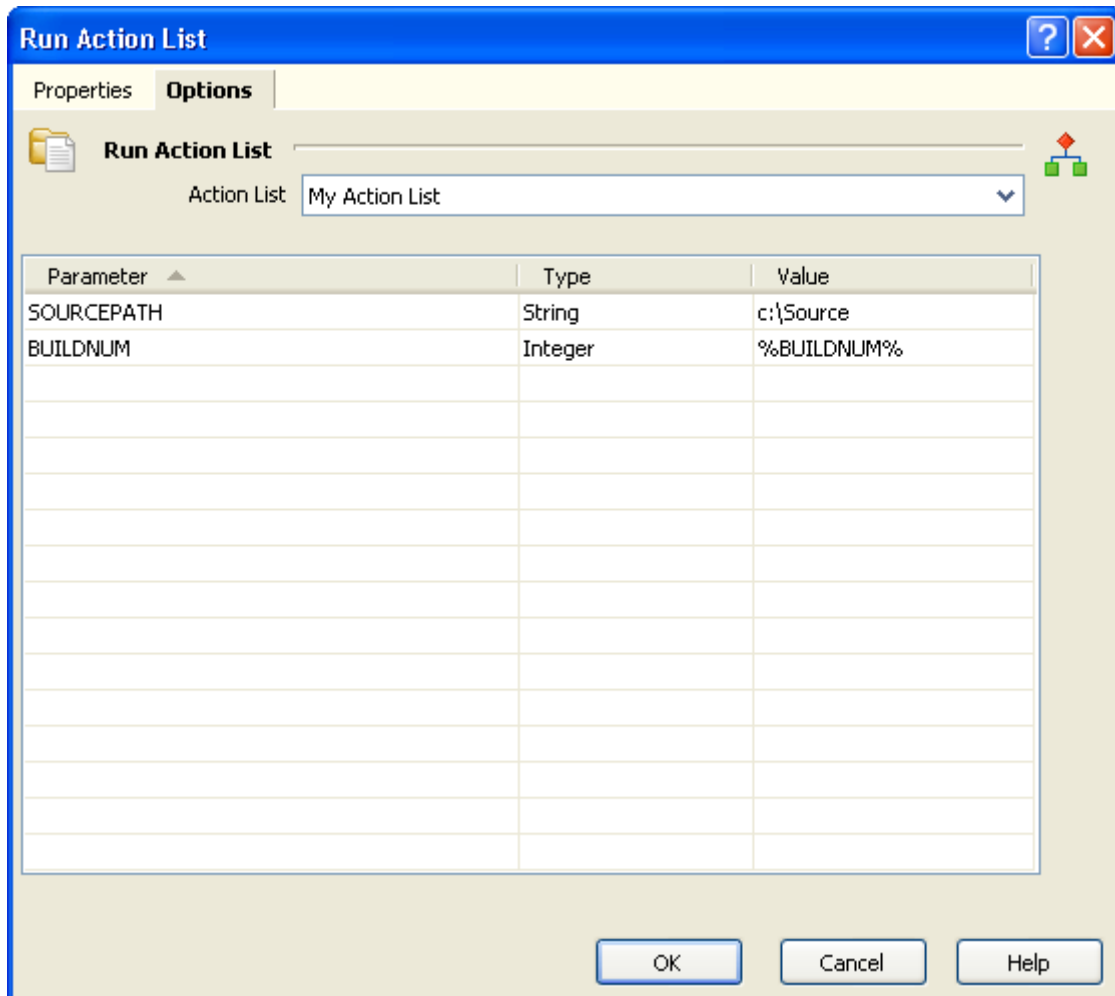
The "Reload Environment Variables" forces FinalBuilder to reload the environment variables before executing the included project, environment variables are usually only loaded at startup.



### 5.1.3 Run Action List Action

This action enables you to run a different Action List, you can add new Actions Lists from the Project Menu. Note you should take care to avoid circular references, Finalbuilder

does not check for circular references at design time but will detect them at runtime and cause the build to fail.



### See Also

Action Lists  
Action List Parameters

#### 5.1.4 Exit Action List Action

The action causes the current action list to terminate and flow returns to the calling action list. Note that all Finally sections will run first before the action list is exited.

### See Also

Stop Build Action

#### 5.1.5 While Loop Action

The While Loop will execute while it's Condition evaluates to True. The While action has no specific functionality other it's looping capability (specifically it tells the stepping engine which action to run next, overriding the default behaviour).

**NOTE:** Note that it will execute its script events (like any other action) first before executing its child actions. What this means is that the While Loop Action will execute its AfterAction Event Before any of the Child Actions are executed. The reason for this is that the stepping engine does not have any knowledge of which type of action it is executing and does not treat the while loop action any differently.

**While Loop**

**Properties**

**Action Description**

While Loop

Comment :

**Options**

☒ Action Enabled

☐ Ignore Failure

Pause after Run : 0 ms

**Logging Options**

☐ Suppress Log Messages

☐ Log to Variable

**Execute Condition**

Script Language : VBScript

Condition must return a boolean value (True or False)  
Condition syntax defined by script language

Condition : Count < 20

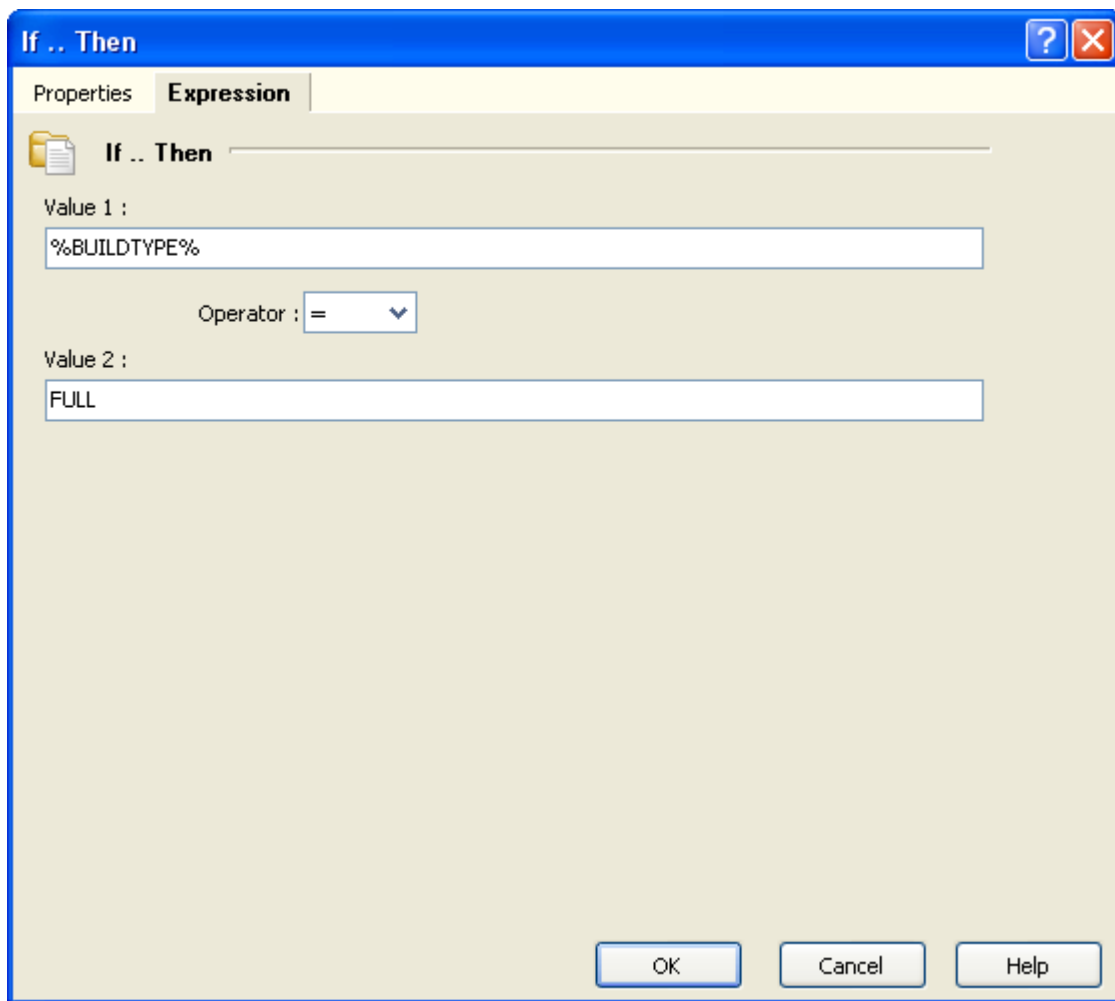
**Copyright**

FinalBuilder - Copyright © 2000-2004 VSoft Technologies Pty Ltd

OK Cancel Help

### 5.1.6 If .. Then Action

This action uses a simple expression to determine if its child actions should run or not.



You can also use execute a set of actions if the if statement evaluates to false.  
See Else Action

### 5.1.7 If Prev Action Failed Action

This action will execute it's child actions only if the previous sibling action failed. For this to happen the previous sibling action needs Ignore Failure enabled.

You can also use execute a set of actions if the previous action succeeds.  
See Else Action

### 5.1.8 Else Action

The else action can be used as an Else part to the "If..Then" action, the "If Prev Action Failed" action or the Switch Action. When used with the Switch Action the Else action should be the last child action of the Switch Action (an error will be reported if not).

#### See Also

If .. Then Action | If Prev Action Failed Action | Switch Action | Case Action

### 5.1.9 Stop Build Action

This action will cause the build to stop . The BuildResult property determines if the build stops with success or failure. If BuildResult is False the OnFailure List will execute. Note that if the Stop Action is inside a Try/Finally block the Finally block will still execute.

#### See Also

Exit Action List Action

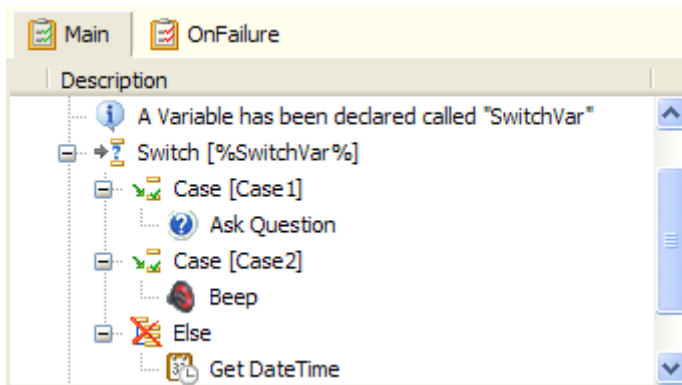
### 5.1.10 Raise Exception

The raise exception action will cause an error condition to be set at the Raise Exception action. The flow of the build will then change as if an error had occurred. The flow will depend on if the action is enclosed in any TRY blocks, and if the error isn't handled by an EXCEPT block then the flow will switch to the OnFailure action list.

### 5.1.11 Switch Action

This action (along with the Case & Else actions) provides a simple selector based on a simple case insensitive string comparison. The Case actions should be child actions of the Switch action. When a case value matches the switch value the child actions of that Case action will be executed.

Below is an example using Switch, Case, and Else.



#### See Also

Case Action | Else Action

### 5.1.12 Case Action

See the Switch Action for details.

#### See Also

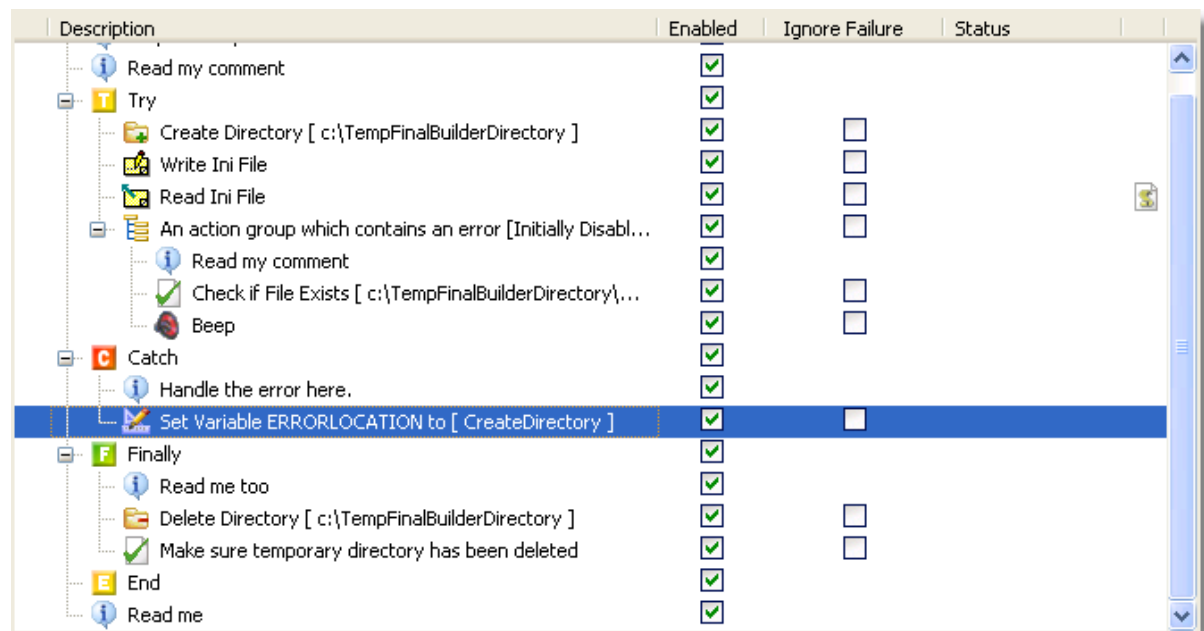
Switch Action | Else Action

### 5.1.13 Try/Catch/Finally/End Actions

The Try action, along with the Catch, Finally and End actions provide structured exception handling. They allow you to create localised error handling and resource protection, just as you do in programming languages such as C++, C#, Delphi etc.

For each Try action, there should be a matching End Action at the same level. The Try must also have either a Catch Action or a Finally Action as it's next sibling. The actions can be used in the following combinations :

1. Try  
....  
Catch  
...  
End
2. Try  
...  
Finally  
...  
End
3. Try  
...  
Catch  
...  
Finally  
...  
End

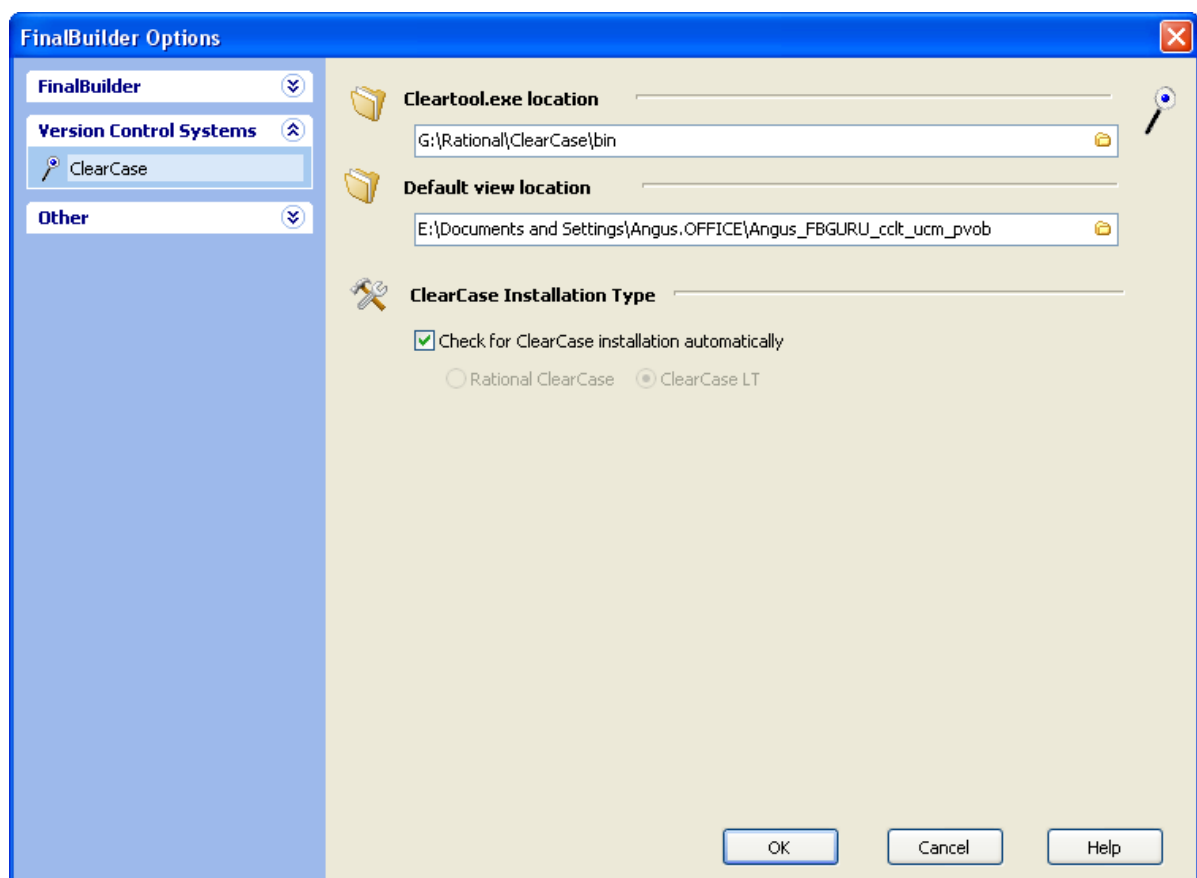


Note that the Finally Action will almost always execute it's child actions, the exception being if there is a structural error with the try or catch (ie. a missing end, or finally before catch etc).

## 5.2 Version Control Actions

### 5.2.1 ClearCase

The Rational ClearCase actions provide an interface via Rational's Cleartool. Each action wraps almost all the functionality of the relevant cleartool command. For unsupported commands, there is a generic Cleartool interface which allows Cleartool to be called directly.



**Default View Location** allows you to specify a "view context" directory to be used as the default for all ClearCase actions which use a view context.

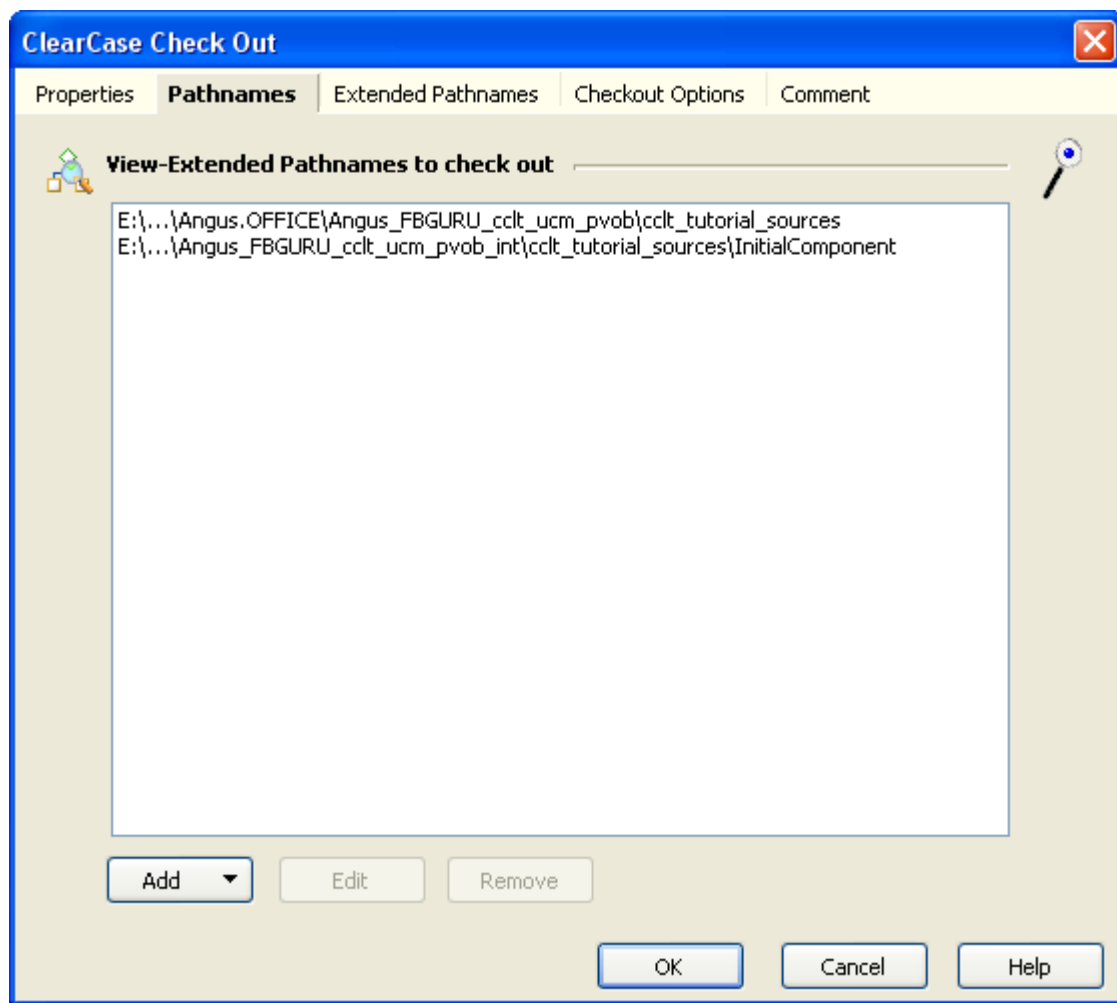
**Cleartool.exe location & ClearCase Installation Type** should be set automatically on a computer with a ClearCase installation, although they can be overridden manually if desired. ClearCase Installation Type enables and disables some options which are only available in ClearCase or ClearCase LT, respectively.

### 5.2.1.1 ClearCase Pathnames

When choosing elements from a ClearCase VOB, selections can be made via View-Extended, Version-Extended or VOB-Extended pathnames.

For a full discussion of Pathnames in ClearCase, see the Pathnames section of the ClearCase manual (run *cleartool man pathnames\_ccase*).

**View-Extended Pathnames** can be found under the '**Pathnames**' tab:

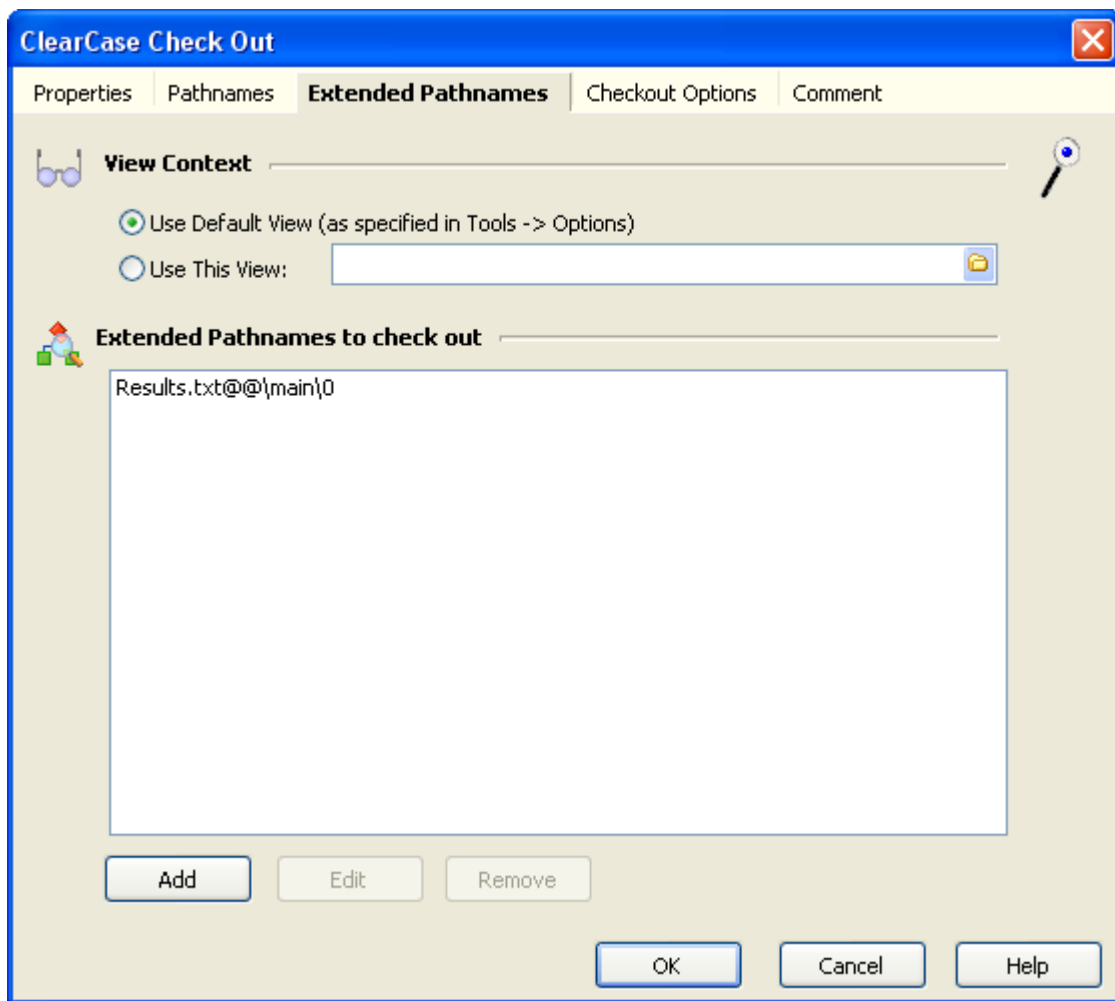


Hover the mouse over a pathname to see the full path.

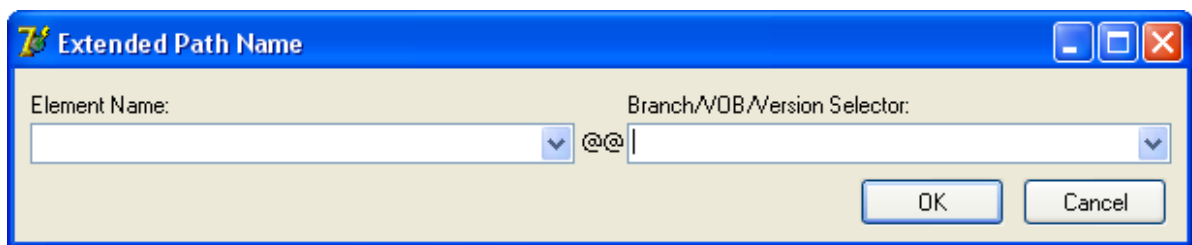
**Version- and VOB- Extended Pathnames** can be found under the '**Extended Pathnames**' tab:

Note: Extended Pathnames are not available in ClearCase LT.





Click on Add to specify extended pathnames to act on:



Click on the drop downs to see a list of elements for that view, or a list of available Branches, VOBs or Versions for each element.

#### 5.2.1.2 ClearCase Object Selector

The Object Selector allows some ClearCase actions to target abstract ClearCase objects.

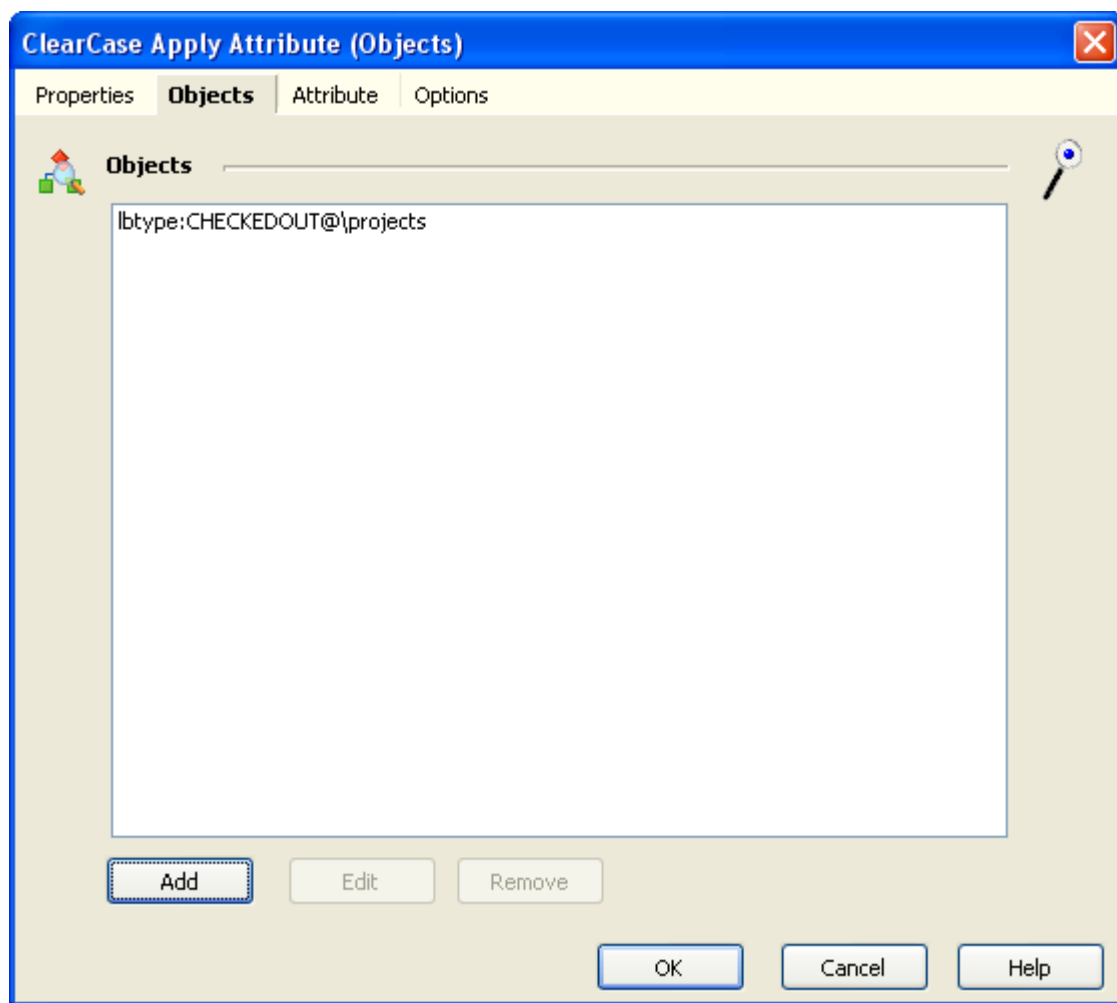
Objects that can be targeted include:

- VOBs
- Attribute Types
- Branches
- Elements
- Hyperlinks
- Labels
- Triggers
- Pools
- Object IDs

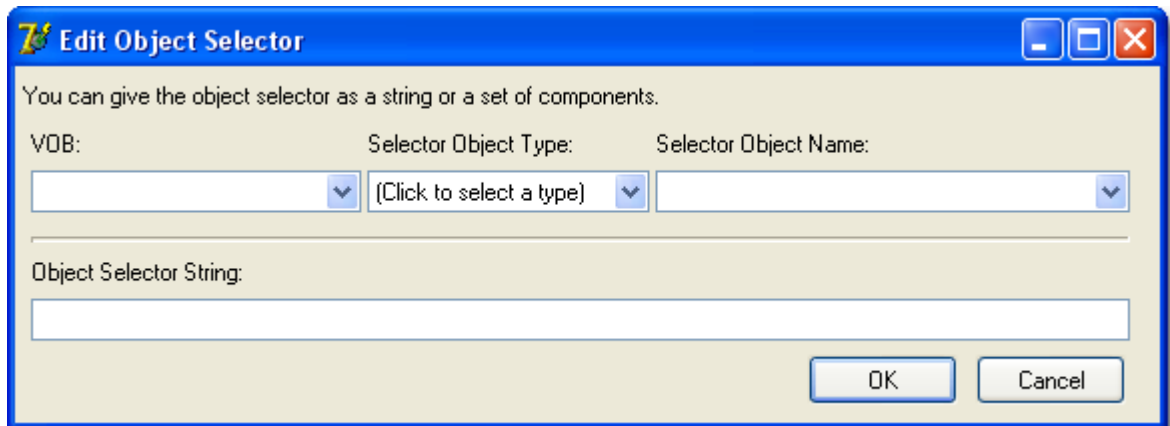
Also,

- UCM Activities
- UCM Baselines
- UCM Components
- UCM Folders
- UCM Projects
- UCM Streams

In the screenshot below, a label type has been selected to apply an attribute:



Clicking the Add button brings up the object selection dialog:



Object selectors can be specified as an object selector string (bottom row) or as a combination of constituent components (top row.) 'Object Name' dropdown menu in the top row is context sensitive (ie its contents will depend on what VOB name and object type has been selected.)

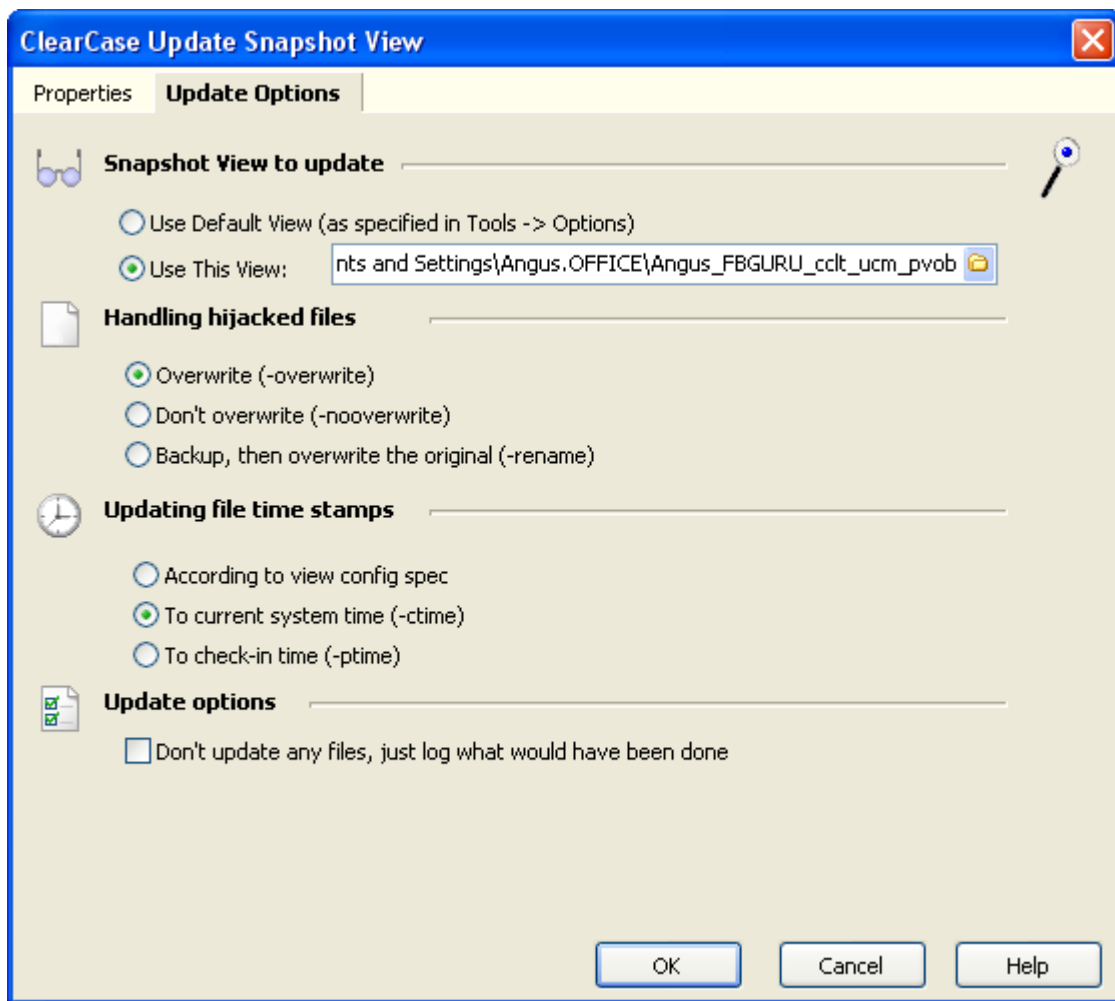
Typing a selector into the bottom edit field changes the contents of the top fields automatically, and vice versa.

### 5.2.1.3 Base ClearCase

#### 5.2.1.3.1 ClearCase Update Snapshot View Action

Use this action to update a ClearCase LT Snapshot View (this action is not necessary when using ClearCase with MVFS.)

For full details of the update command, see the ClearCase manual page (type *cleartool man update*.)



The snapshot view to update can be either the default view (specified on the Options panel) or a view specific to that action.

### Handling Hijacked Files

Hijacked files can either be overwritten with the copy from the VOB, left alone, or backed up (to *filename.keep.*) See the manual page for complete details.

### Updating File Time Stamps

Updated files can be updated to local system time, or kept with the timestamp from the VOB.

### Update Options

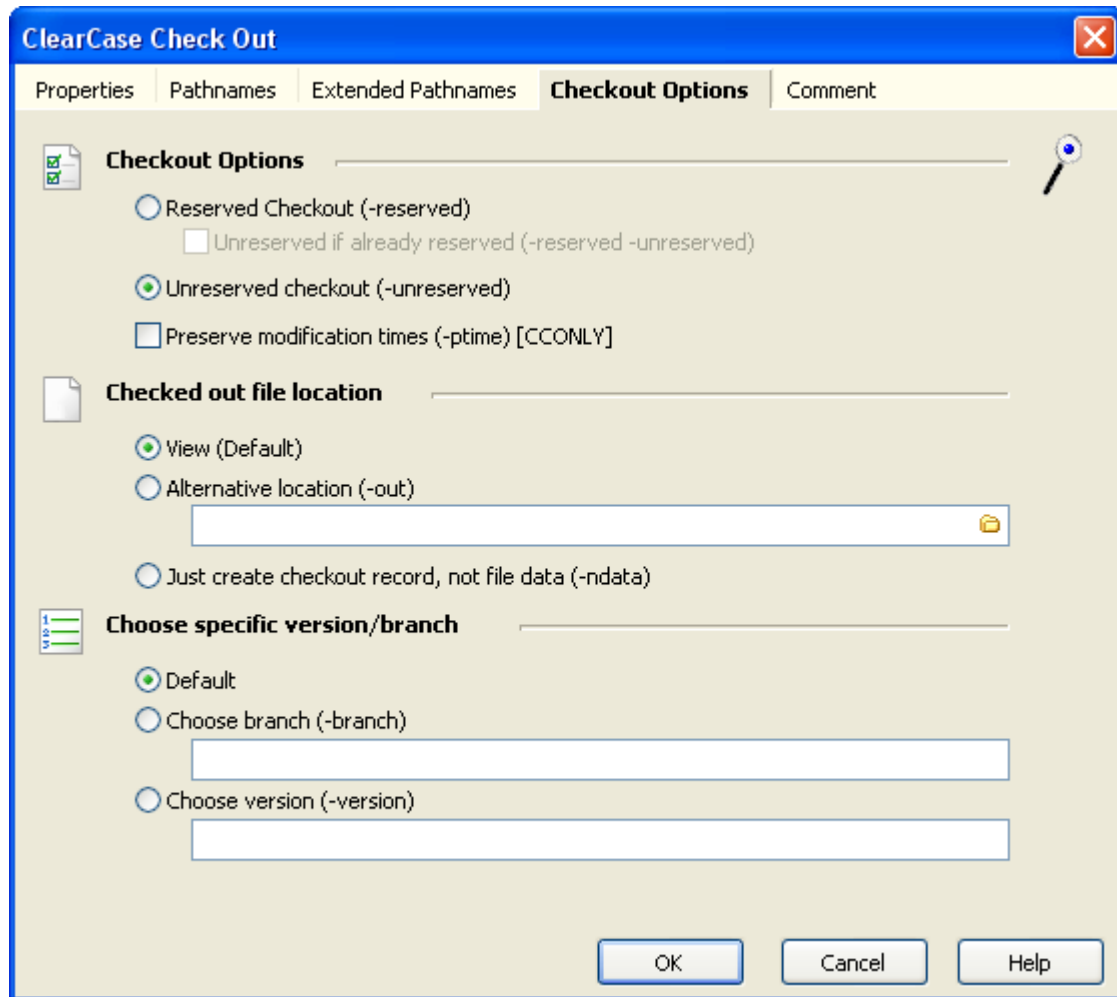
Update can also be used to just log the differences between the VOB and the current view directory.

## 5.2.1.3.2 ClearCase Check Out Action

Use this option to check out elements (files or directories) from VOBs.

For full details of the checkout command, see the ClearCase manual page (type *cleartool man checkout*.)

The ClearCase Check Out Action can use View- or Extended- pathnames.

**Checkout Options**

Checked out files can be reserved or unreserved. Only one user can hold a reserved checkout on a single element. See the manual page for more details.

**Checked Out File Location**

By default, files are checked out to the view they were accessed from.

### Choose specific version/branch

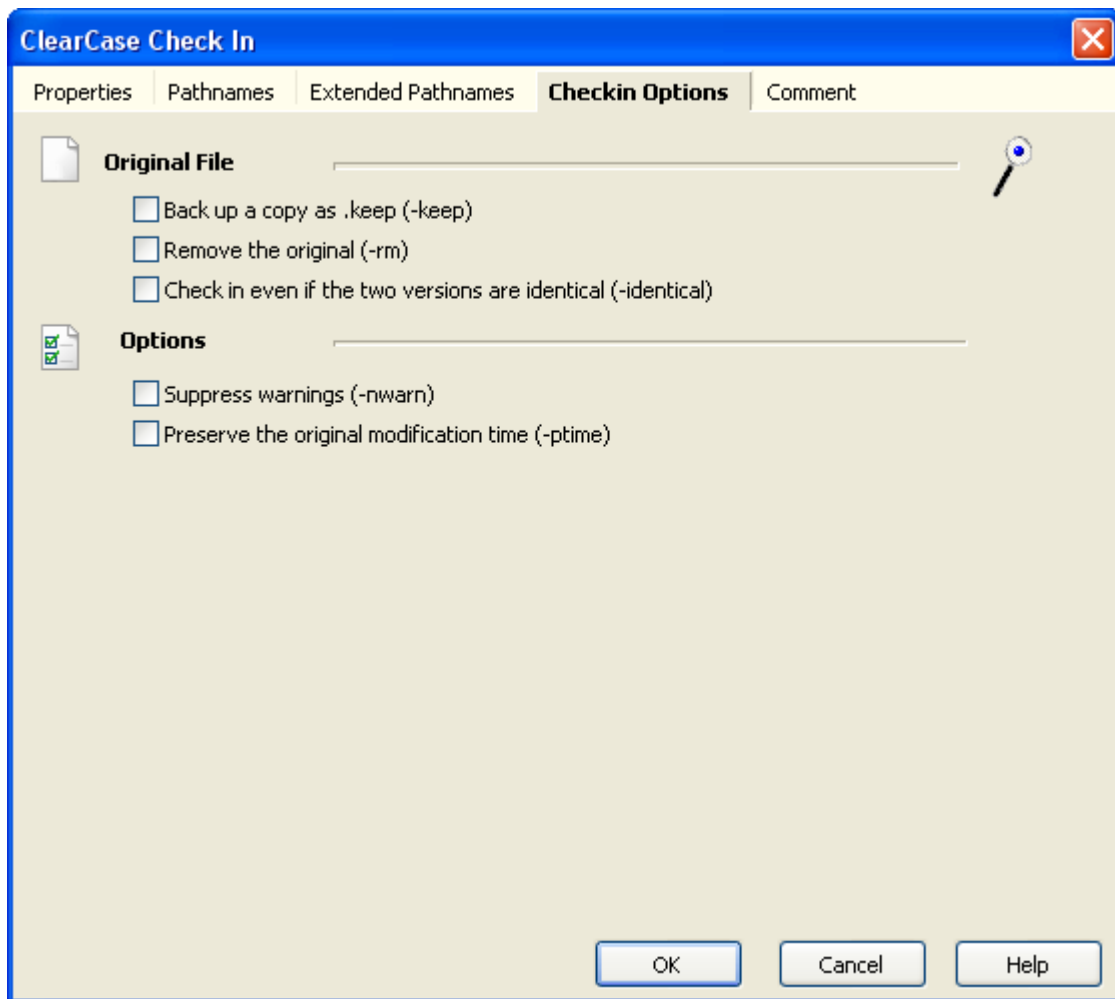
Different versions/branches to the one specified by the config spec can only be checked out from dynamic views. This option is not available on ClearCase LT.

#### 5.2.1.3.3 ClearCase Check In Action

Use this action to check in elements (files or directories) to a VOB. To add new elements to a VOB, use the ClearCase Make Element action.

For full details of the checkin command, see the ClearCase manual page (type *cleartool man checkin.*)

The ClearCase Check In Action can use View- or Extended- pathnames.



### Original File

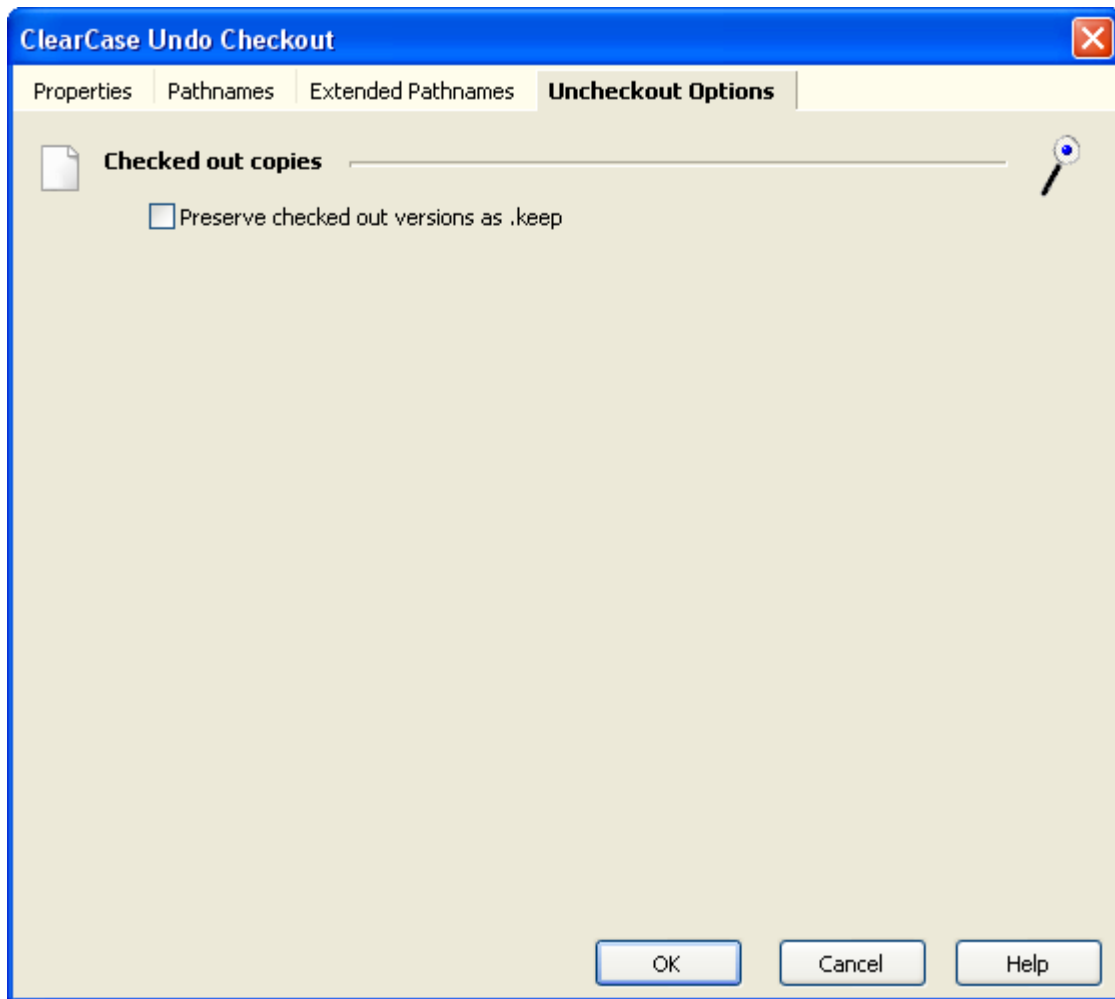
By default, in a dynamic view, cleartool deletes each view-private, checked-out file after using it to create a new version in the VOB. In a snapshot view, cleartool uses the checked-out file to create a new version, then loads the new version into the view.

#### 5.2.1.3.4 ClearCase Undo Checkouts Action

Use this action to undo the check out of some checked out files. For users using UCM, there is also a UCM Undo Checkouts for Activity action..

For full details of the uncheckout command, see the ClearCase manual page (type *cleartool man uncheckout*.)

The ClearCase Undo Checkouts Action can use View- or Extended- pathnames.



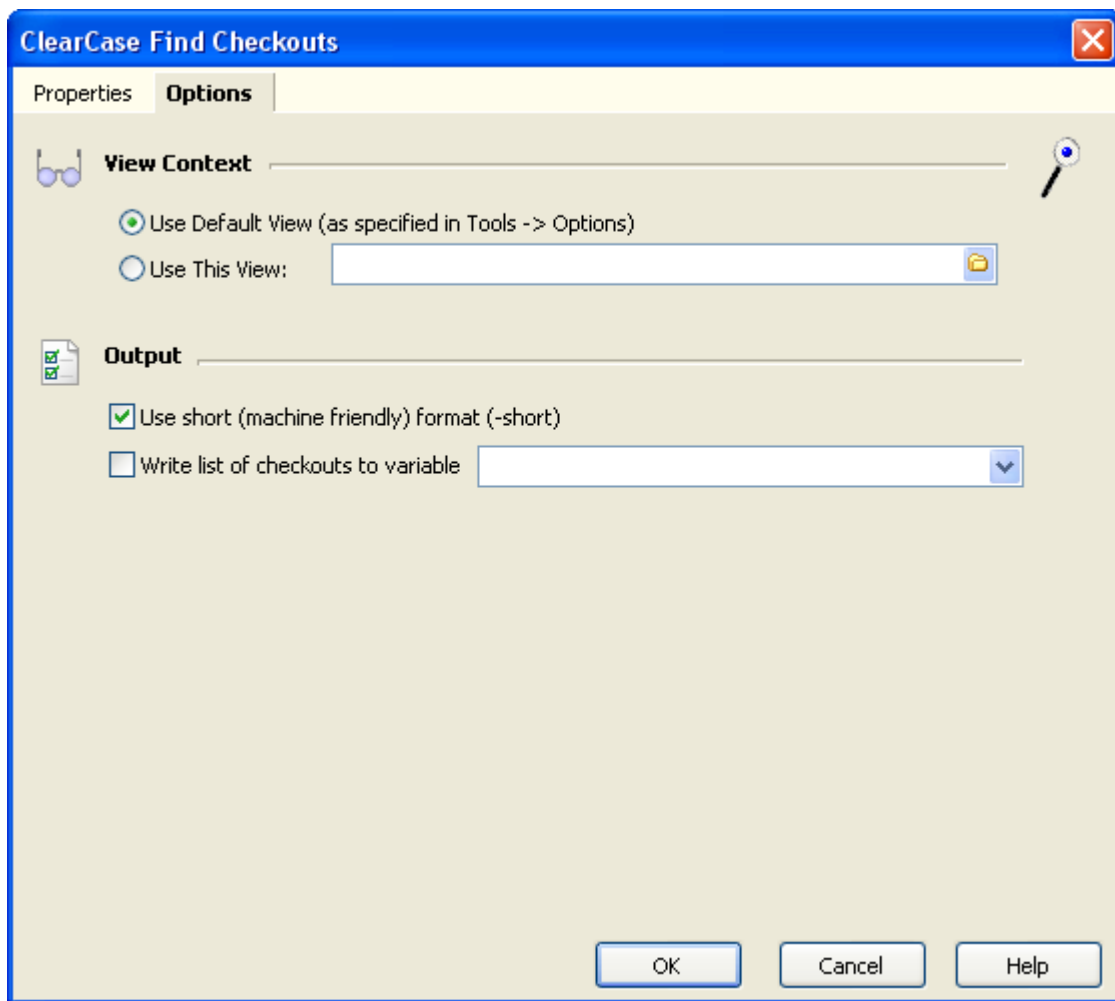
**Checked out copies** can be deleted or preserved as *filename.keep*.

#### 5.2.1.3.5 ClearCase Find Checkouts Action

Use this action to find all the checked out elements from a view.

For full details of the find command, see the ClearCase manual page (type *cleartool man find*.)

The ClearCase Find Checkouts Action can use View- or Extended- pathnames.



The view to search in can be either the default view (specified on the Options panel) or a view specific to that action.

## Output

The list of checkouts can be either long (with details of the checkout) or a short summary of the checked out files.

The list of checkouts can also be written to a variable and used (for example) with a list iterator. If you are writing to a variable, it is recommended that you use the short (machine friendly) format.

### 5.2.1.3.6 ClearCase Make Element Action

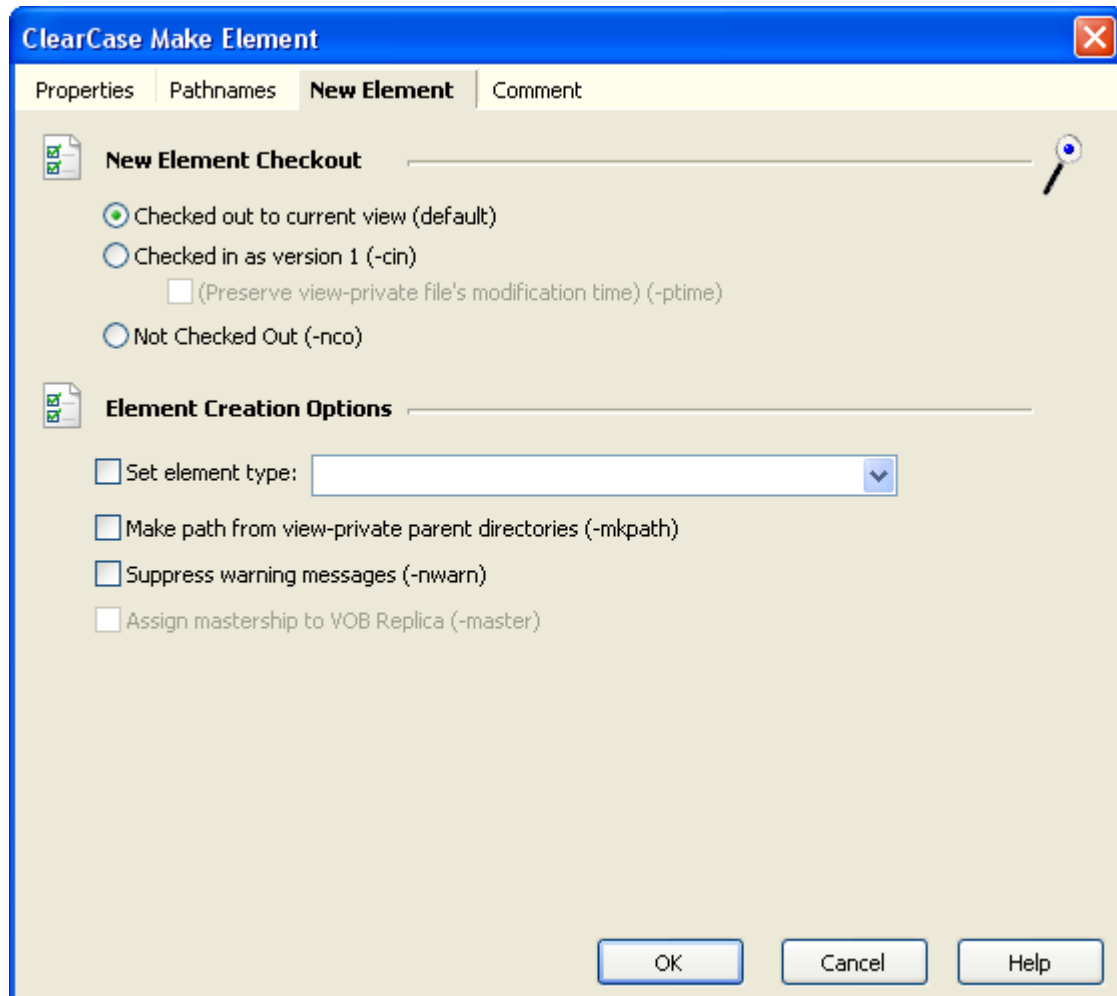
Use this action to make a new element (file, directory, etc.) The new element is created in both the VOB and the view directory.

For full details of the mkelem command, see the ClearCase manual page (type *cleartool man mkelem*.)

The ClearCase Make Element Action can only create elements specified by View-extended



pathnames.



## Element Creation Options

### Set Element Type

The element type is not required, but allows ClearCase to automatically manage the contents of some files. Set the element type to 'Directory' if you are creating a directory. Click on the dropdown menu to get a list of all element types in all VOBs.

### Make path from view-private parent directories

Use this option if the new element has a parent directory which also does not exist in the VOB.

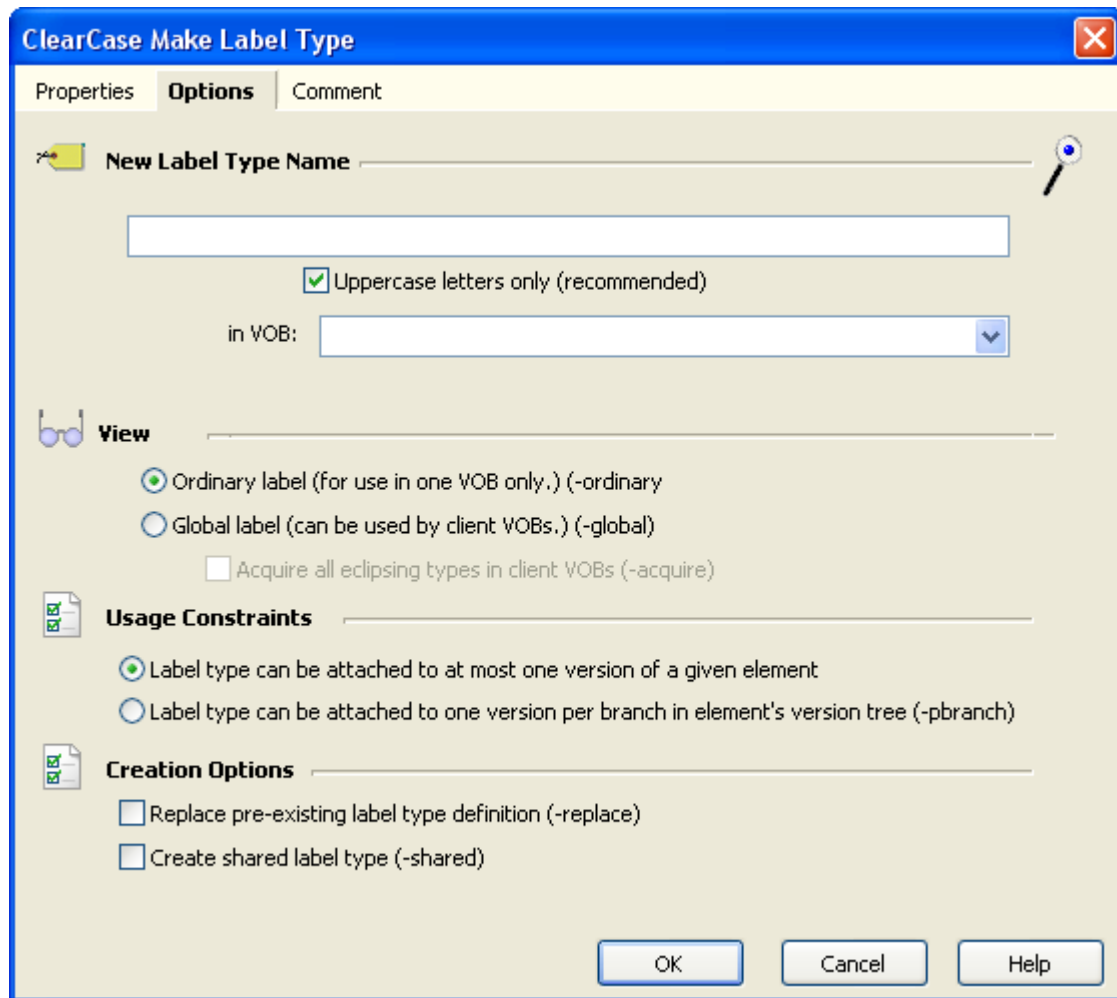
### Assign Mastership to VOB Replica

This option is not available in ClearCase LT.

#### 5.2.1.3.7 ClearCase Make Label Type Action

Use this action to make a new label type in a VOB. The new label type can then be assigned to elements and components within that VOB.

For full details of the mklbtype command, see the ClearCase manual page (type *cleartool man mklbtype*.)



### New Label Type Name

By convention, and to avoid conflict with branch names, ClearCase Label Type Names are normally given in all uppercase letters.

Click on the dropdown menu to see a list of all available VOBs.

### View

The label can be scoped either for a single VOB (Ordinary) or for all client VOBs as well (Global.)

### Usage Constraints

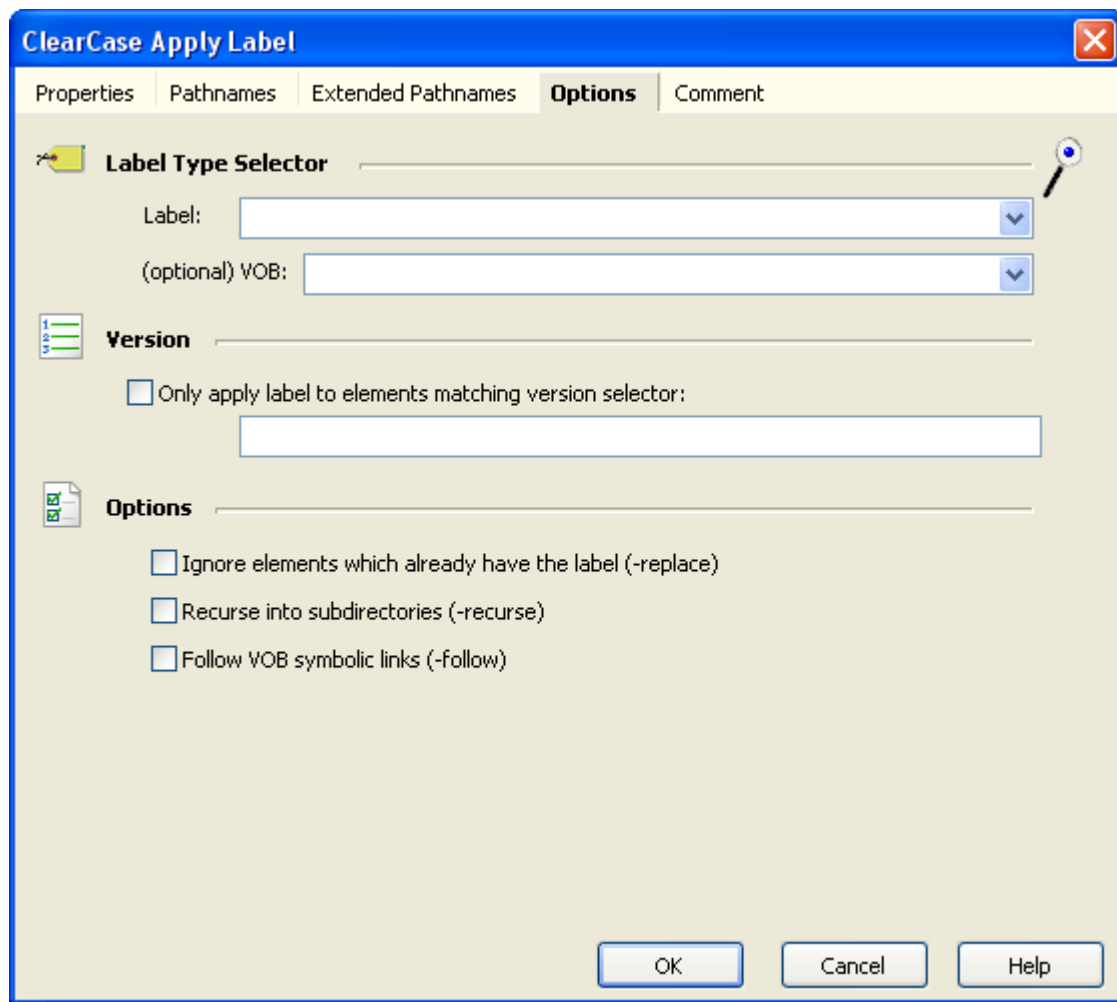
By default, and to avoid ambiguity, label types can be attached to only one version of a given element. This constraint can be relaxed to allow each branch to contain a labelled version.

#### 5.2.1.3.8 ClearCase Apply Label Action

Use this action to apply a label to one or more elements in a VOB.

For full details of the `mklabel` command, see the ClearCase manual page (type *cleartool man mklabel*.)

The ClearCase Apply Label Action can use View- or Extended- pathnames.



### Label Type Selector

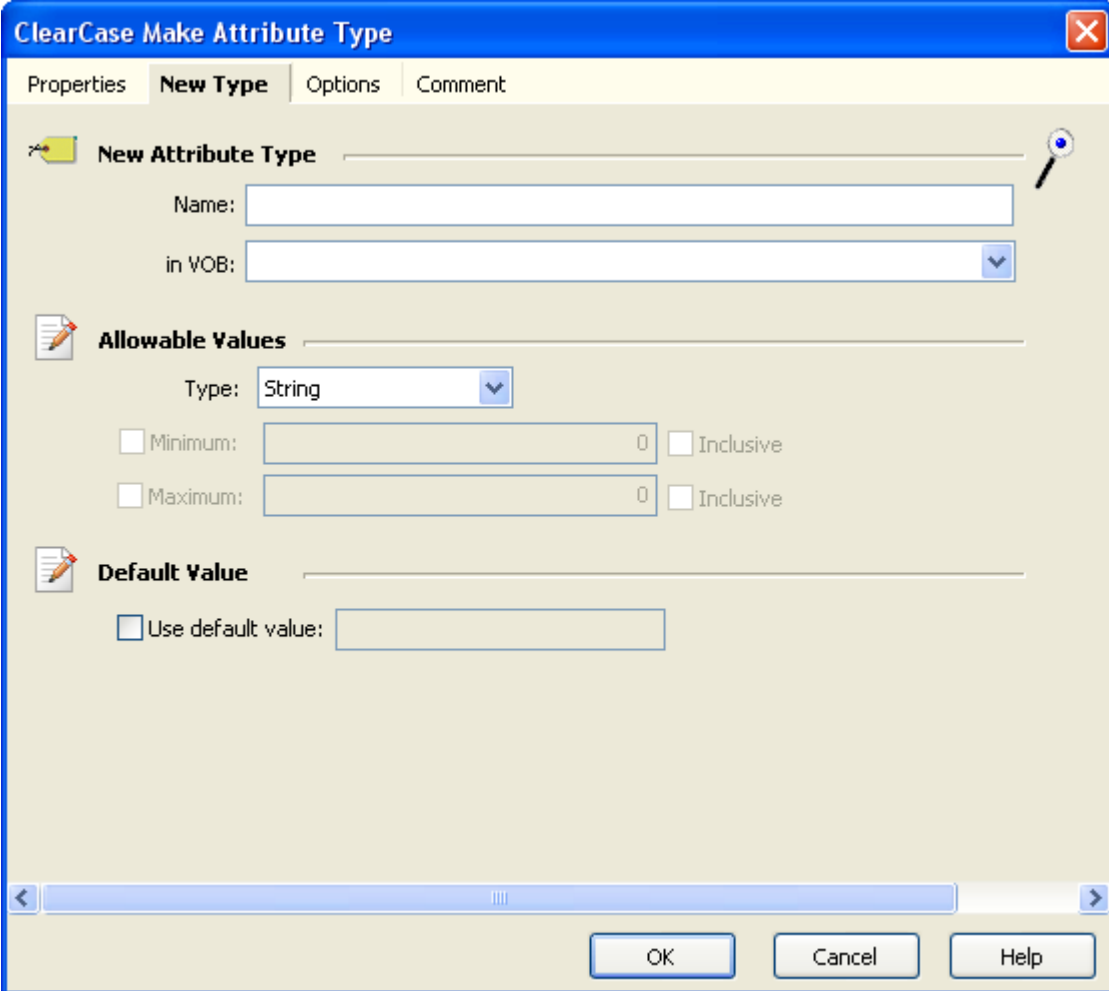
If you have chosen a VOB, Click on the Label drop down menu to see a list of labels available in that VOB.

A VOB selector is not required if the Label exists in the default VOB for the view context.

## 5.2.1.3.9 ClearCase Make Attribute Type Action

Use this action to make a new attribute type in a VOB. The new attribute type can then have values assigned to elements and components within that VOB.

For full details of the mkatttype command, see the ClearCase manual page (type *cleartool man mkatttype*.)

The image shows the 'ClearCase Make Attribute Type' dialog box. It has a title bar with a close button. Below the title bar are four tabs: 'Properties', 'New Type' (which is selected), 'Options', and 'Comment'. The 'New Type' tab contains three sections: 'New Attribute Type' with a text field for 'Name' and a dropdown for 'in VOB'; 'Allowable Values' with a 'Type' dropdown set to 'String', and two rows for 'Minimum' and 'Maximum' values, each with an 'Inclusive' checkbox; and 'Default Value' with a checkbox 'Use default value:' and an empty text field. At the bottom are 'OK', 'Cancel', and 'Help' buttons.**New Attribute Type**

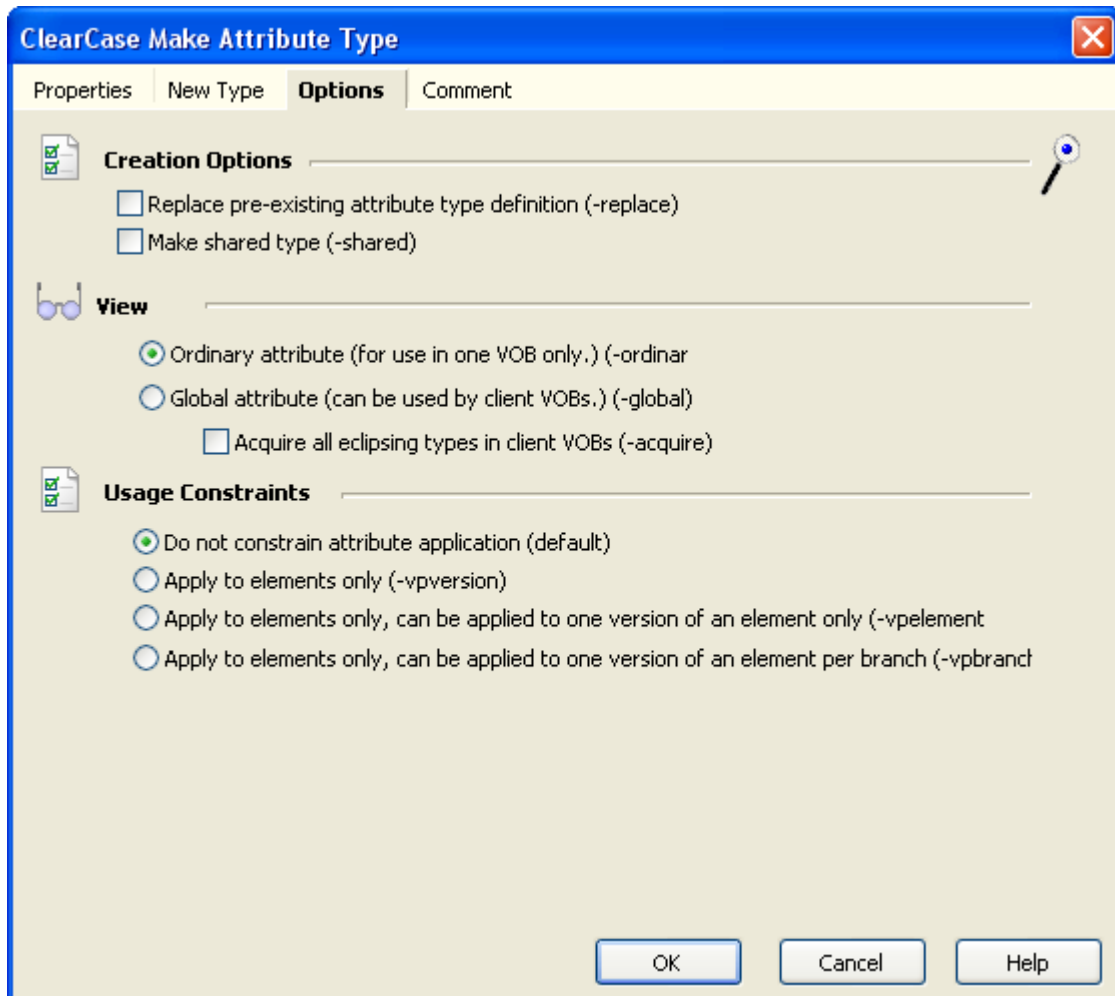
Click on the dropdown menu to see a list of VOBs.

**Allowable Values**

Attributes can be of type String, Integer, Real, Date or Opaque (a single unsigned byte.)

For cardinal types (all apart from String), minimum and maximum values can be

specified.



### Creation Options

If the attribute type already exists, 'Replace pre-existing...' must be checked or the action will fail at runtime.

### View

The attribute can be scoped either for a single VOB (Ordinary) or for all client VOBs as well (Global.)

### Usage Constraints

Application of the attribute type can be restricted in order to avoid ambiguity.

## 5.2.1.3.10 ClearCase Apply Attribute Action

Use this action to apply an attribute (with value) to one or more elements in a VOB.

For full details of the mkattr command, see the ClearCase manual page (type *cleartool man mkattr*.)

The Apply Attribute Action comes in two forms: Apply Attribute (Paths) and Apply Attribute (Objects.) The (Paths) form chooses the elements via ClearCase Pathnames, while the (Objects) form uses the object selector.

The screenshot shows the 'ClearCase Apply Attribute (Paths)' dialog box. It has a blue title bar and four tabs: 'Properties', 'Pathnames', 'Attribute' (which is selected), and 'Options'. The 'Attribute' tab contains the following fields and controls:

- Attribute Name:** A label with a folder icon and a text field. To the right is a magnifying glass icon.
- VOB:** A dropdown menu currently showing '\projects'.
- Name:** A dropdown menu currently showing 'PromotionLevel'.
- Attribute Value:** A label with a document icon and a large text input field.
- ☐ Use default value
- (Type: String, default value: "INITIAL")

At the bottom right, there are three buttons: 'OK', 'Cancel', and 'Help'.

**Attribute Name**

Click on the VOB dropdown menu to see a list of available VOBs. Once a VOB has been selected, click on the Name dropdown menu to see a list of available attribute type names.

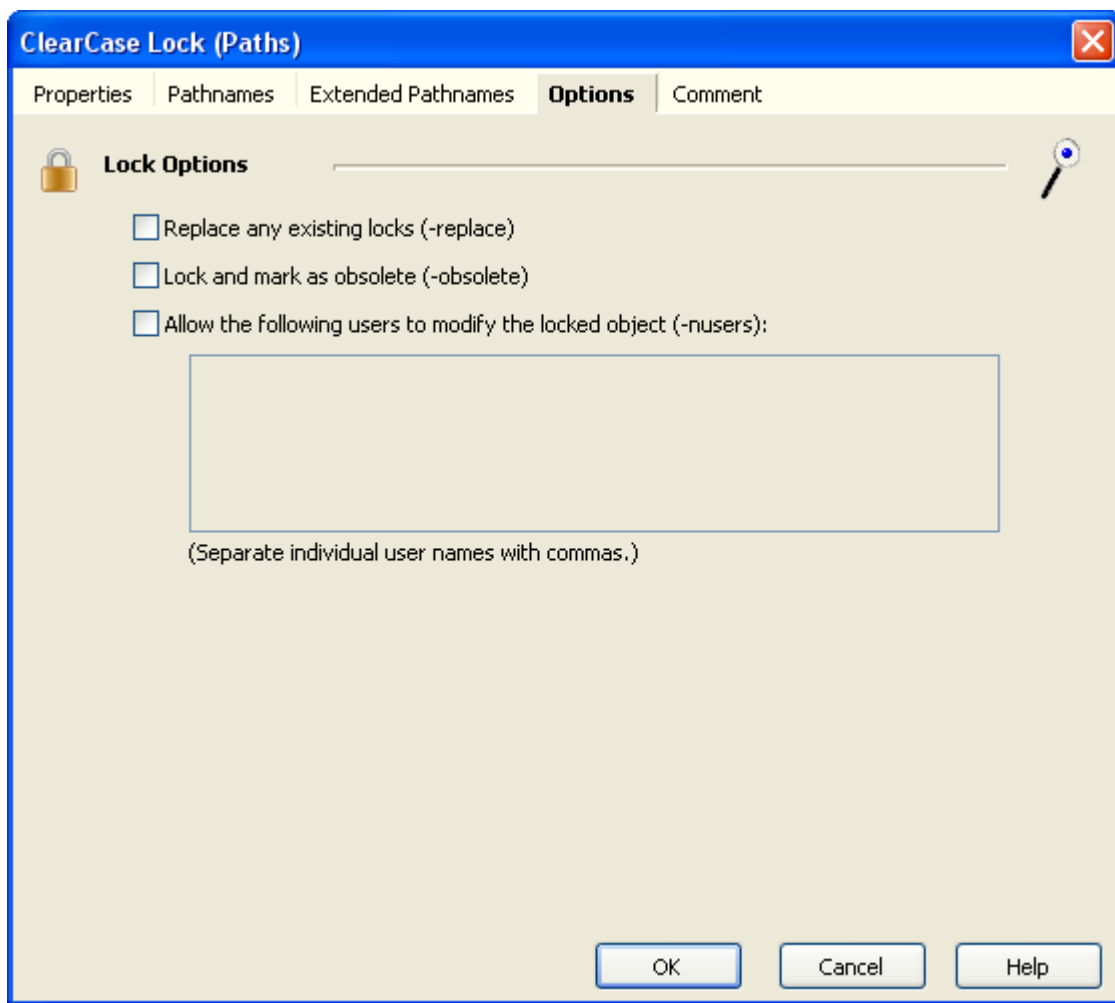
Once an attribute name has been chosen, Finalbuilder will try and determine the type and the default value (if applicable.)

## 5.2.1.3.11 ClearCase Lock Action

Use this action to lock one or more elements in a VOB.

For full details of the lock command, see the ClearCase manual page (type *cleartool man lock.*)

The Lock Action comes in two forms: Lock (Paths) and Lock (Objects.) The (Paths) form chooses the elements to lock via ClearCase Pathnames, while the (Objects) form uses the object selector.

**Replace any existing locks**

The action will fail if an element is already locked and this option is unchecked.

**Lock and mark as obsolete**

Obsolete elements are no longer used in the VOB.

**Allow the following users...**

Locks can provide access control by allowing only certain users to modify the locked contents.

**5.2.1.3.12 ClearCase Unlock Action**

Use this action to unlock one or more elements in a VOB.

For full details of the unlock command, see the ClearCase manual page (type *cleartool man unlock*.)

The Unlock Action comes in two forms: Unlock (Paths) and Unlock (Objects.) The (Paths) form chooses the elements to lock via ClearCase Pathnames, while the (Objects) form uses the object selector.

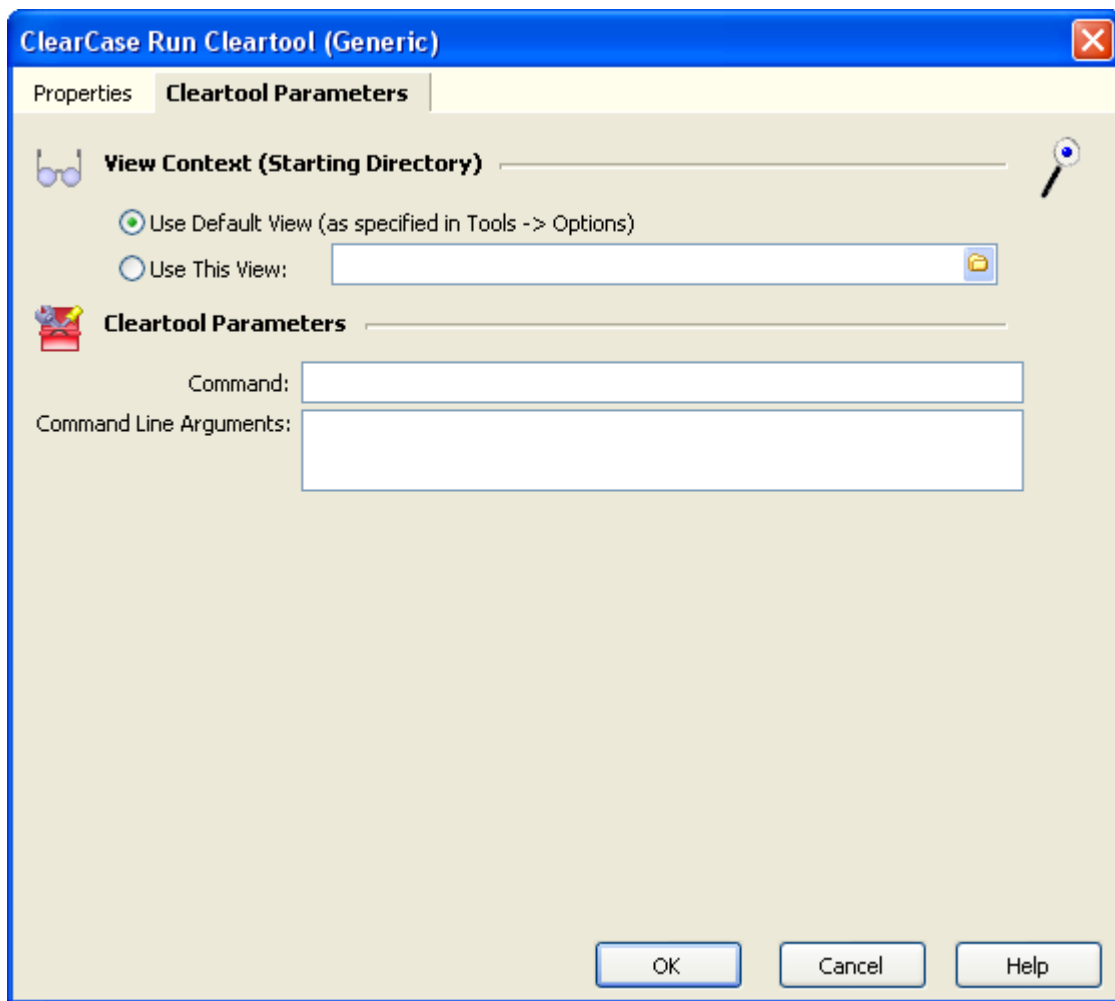
There are no specific options for this action.

**5.2.1.3.13 ClearCase Run Cleartool (Generic) Action**

The ClearCase Run Cleartool (Generic) action allows you to specify any cleartool command and any set of parameters to pass to cleartool.

For full details of Cleartool's capabilities, and a list of commands, type '*cleartool man*'





When the action is run, its behaviour will correspond to the following DOS commands:

```
cd <<View Context (Starting Directory)>>
cleartool <<Command>> <<Command Line Arguments>>
```

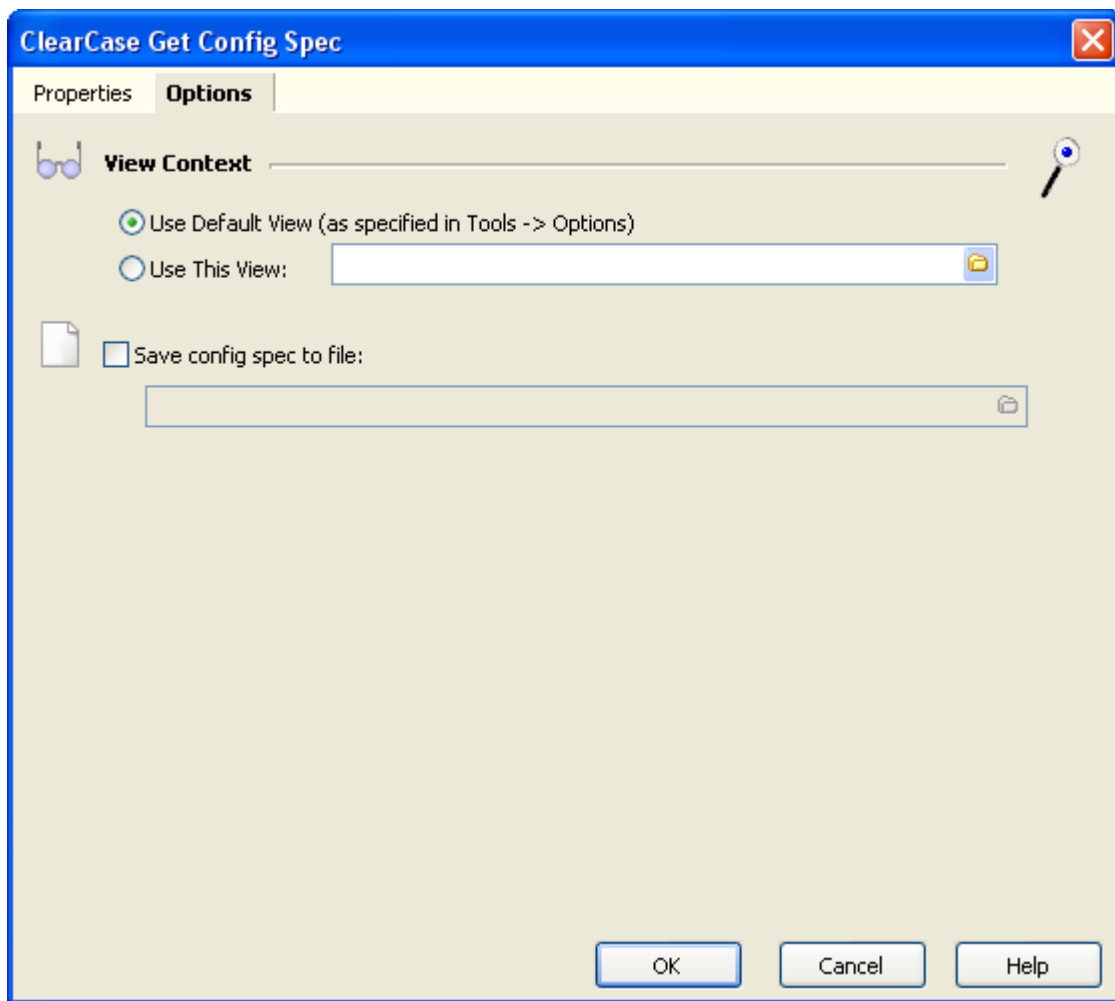
All output will be recording in the log. The action will fail if cleartool fails (ie returns a non-zero return code.)

**Note:** If you find you are using a certain command very frequently, and would like to see an action based on that command, then please [email us](#) and request it!

#### 5.2.1.3.14 ClearCase Get Config Spec Action

Use this action to output the config spec for a view.

For full details of the catcs command, see the ClearCase manual page (type *cleartool man catcs*.)



The config spec can be either that of the default view (specified on the Options panel) or a view specific to that action.

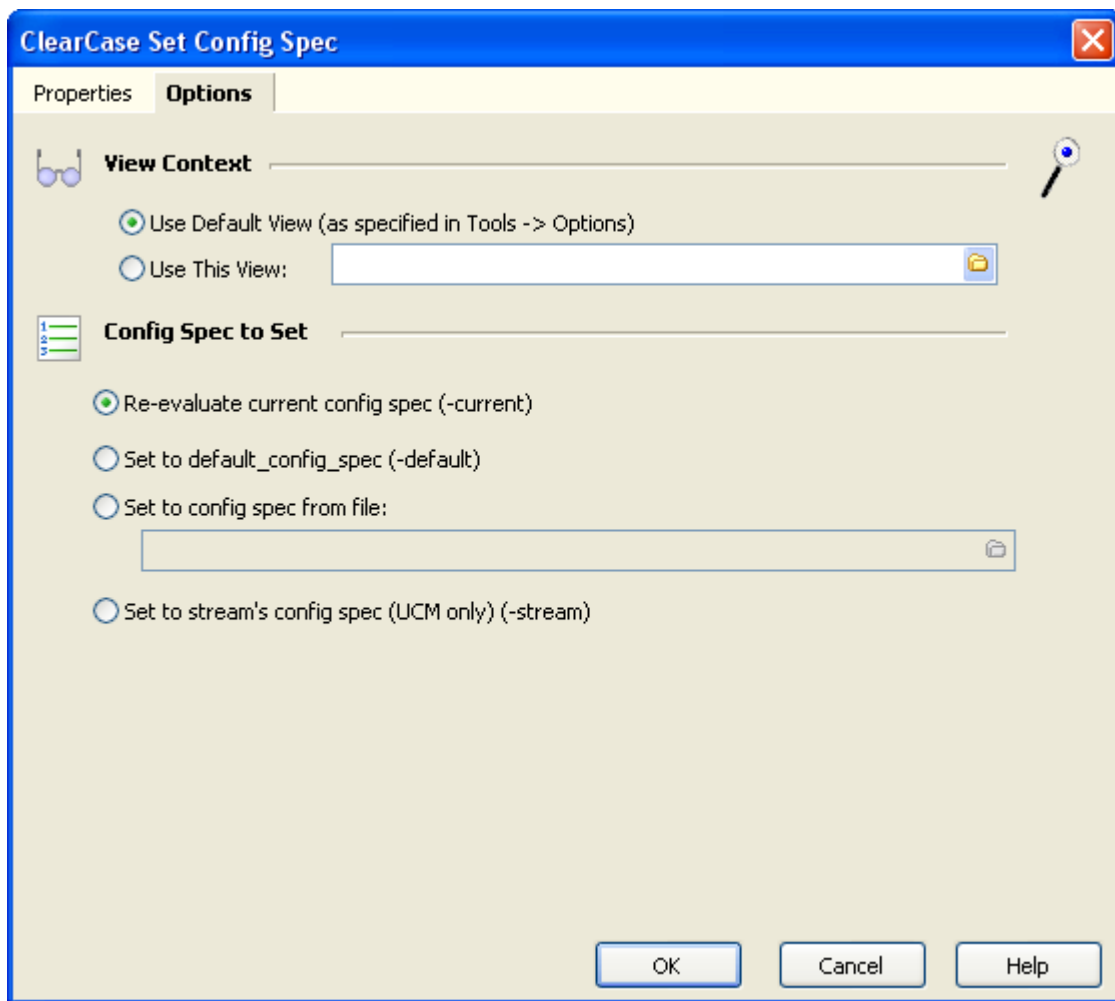
If the 'Save config spec to file' option is not checked, the config spec will be output to the log.

Config specs which have been saved to a file can be set with the Set Config Spec Action.

#### 5.2.1.3.15 ClearCase Set Config Spec Action

Use this action to set the config spec for a view.

For full details of the setcs command, see the ClearCase manual page (type *cleartool man setcs*.)



The config spec can be either that of the default view (specified on the Options panel) or a view specific to that action.

The config spec that is set can be one of:

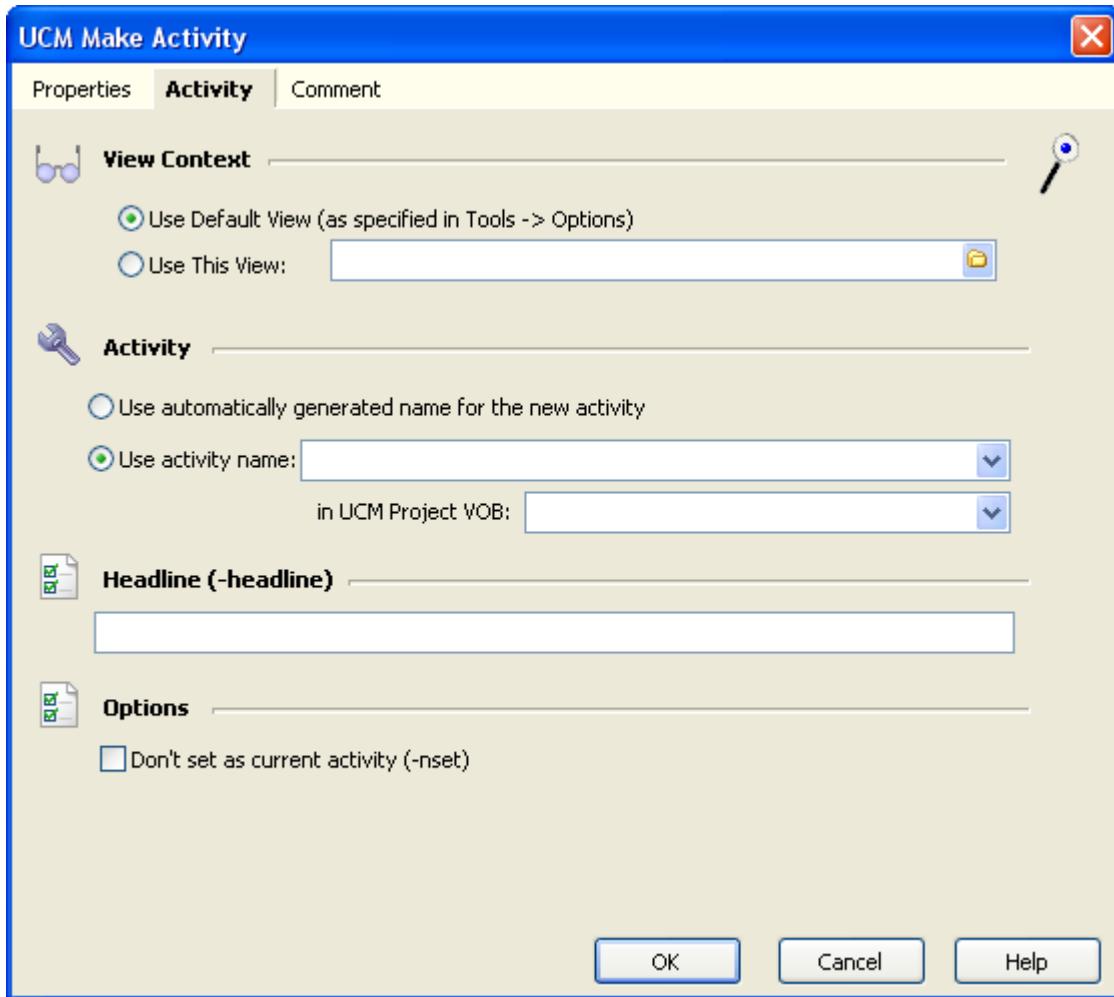
- The current config spec (which is re-evaluated.)
- The default config spec for that view
- A config spec given in a file
- The config spec for the View's UCM Stream

#### 5.2.1.4 UCM

##### 5.2.1.4.1 UCM Make Activity Action

Use this action to create a new activity in a UCM stream. By default, the new activity is set as the new current activity for that stream.

For full details of the `mkactivity` command, see the ClearCase manual page (type `cleartool man mkactivity`.)

The image shows a Windows-style dialog box titled "UCM Make Activity". It has three tabs: "Properties", "Activity", and "Comment", with "Activity" currently selected. The dialog is divided into four sections, each with an icon and a title: "View Context" (glasses icon), "Activity" (key icon), "Headline (-headline)" (document icon), and "Options" (document icon). The "View Context" section has two radio buttons: "Use Default View (as specified in Tools -> Options)" which is selected, and "Use This View:" followed by an empty text box and a folder icon. The "Activity" section has two radio buttons: "Use automatically generated name for the new activity" and "Use activity name:" which is selected, followed by a text box. Below this is "in UCM Project VOB:" followed by a dropdown menu. The "Headline (-headline)" section has a single large text box. The "Options" section has a checkbox labeled "Don't set as current activity (-nset)" which is unchecked. At the bottom right are three buttons: "OK", "Cancel", and "Help".

The stream to use can be taken from the default view (specified on the Options panel) or a view specific to that action.

### Activity

ClearCase can automatically generate names for new activities. If a headline is specified, the name will be based on the headline text.

### Headline

(Optional) a human-readable name for the activity.

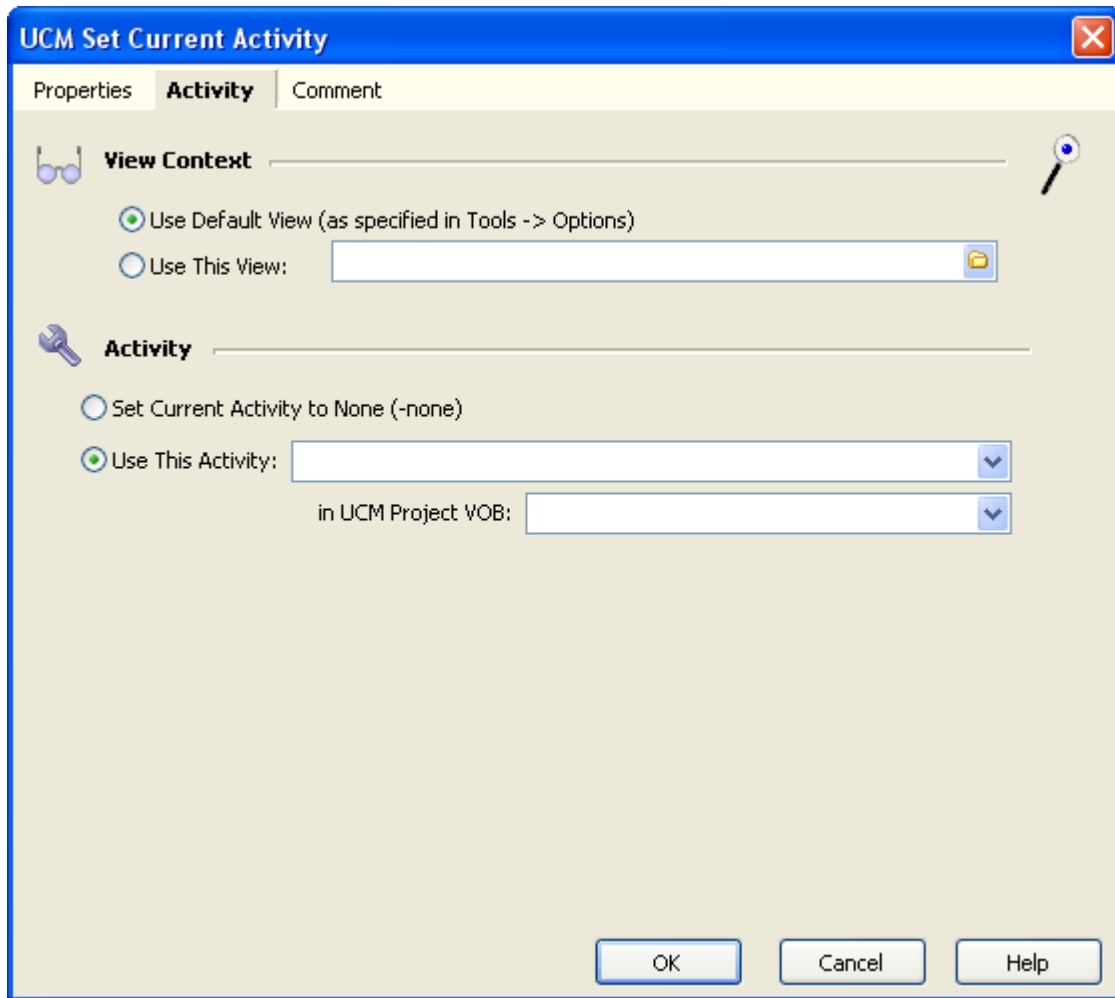
### Options

By default, the newly created activity is set as the current activity for the stream. To avoid this, check the "Don't Set..." check box. The current activity can also be set with the UCM Set Activity Action.

## 5.2.1.4.2 UCM Set Current Activity Action

Use this action to set the current activity for a UCM stream.

For full details of the setactivity command, see the ClearCase manual page (type *cleartool man setactivity*.)



The stream to use can be taken from the default view (specified on the Options panel) or a view specific to that action.

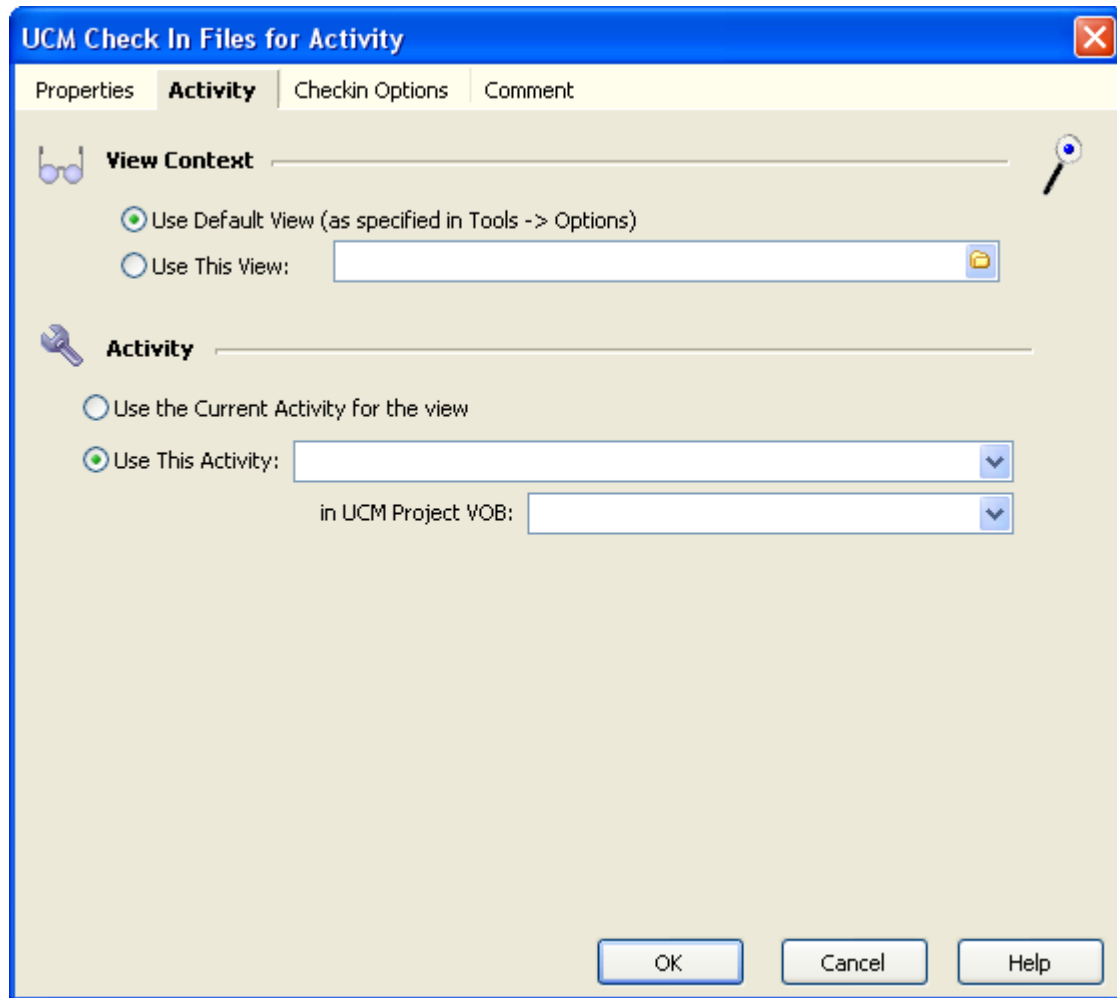
**Activity**

The current activity can be set to 'none', or to any activity from the Stream's UCM project VOB. Click on the VOB dropdown menu to see a list of VOBs, and the Activity dropdown menu to see a list of activities in that VOB.

## 5.2.1.4.3 UCM Check In Files For Activity Action

Use this action to check in all files checked out to a UCM activity.

For full details of the checkin command, see the ClearCase manual page (type *cleartool man checkin*.)



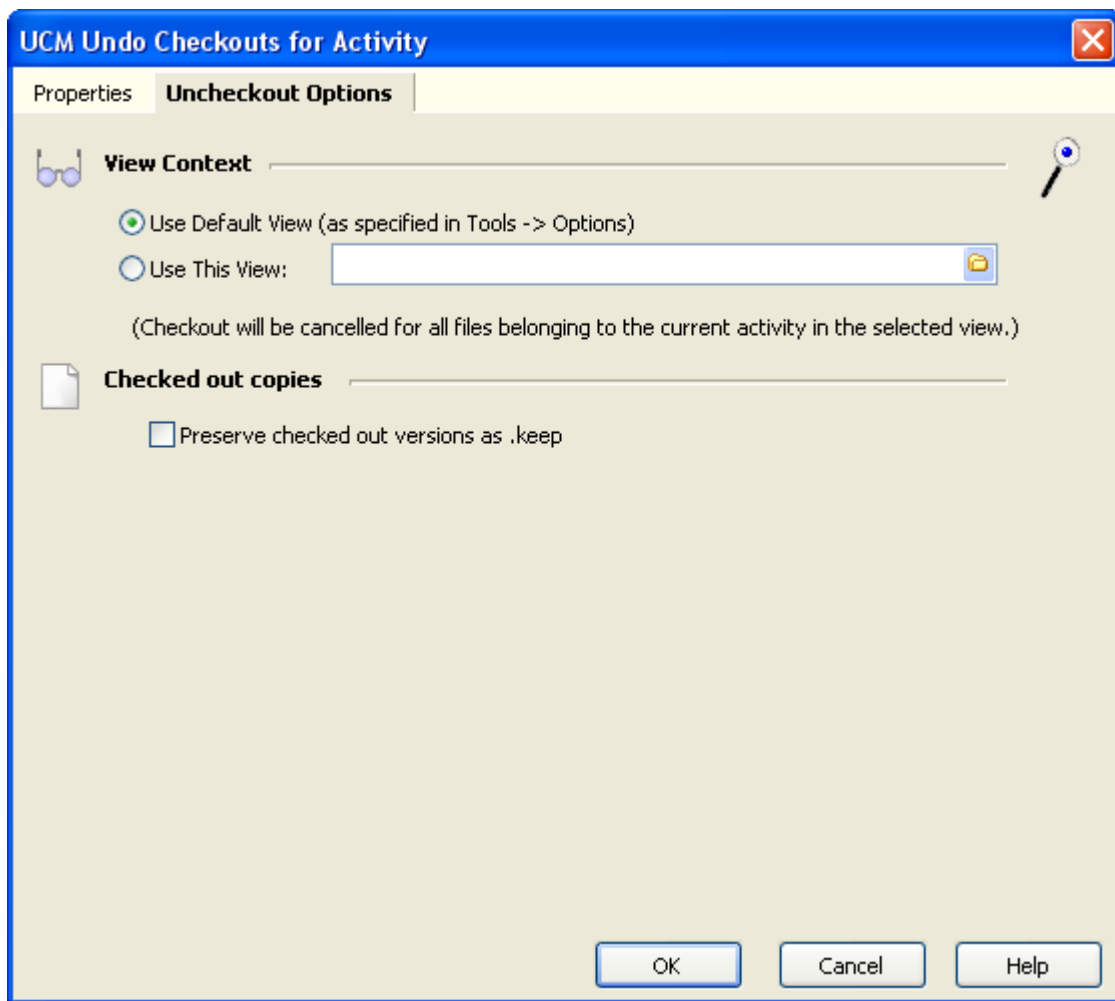
The stream to use can be taken from the default view (specified on the Options panel) or a view specific to that action.

The activity to check in files for can be either the current activity for that view, or a specific activity chosen from a UCM Project VOB.

## 5.2.1.4.4 UCM Undo Checkouts for Activity Action

Use this action to undo all checkouts registered to the current UCM activity for a view.

For full details of the uncheckout command, see the ClearCase manual page (type *cleartool man uncheckout*.)

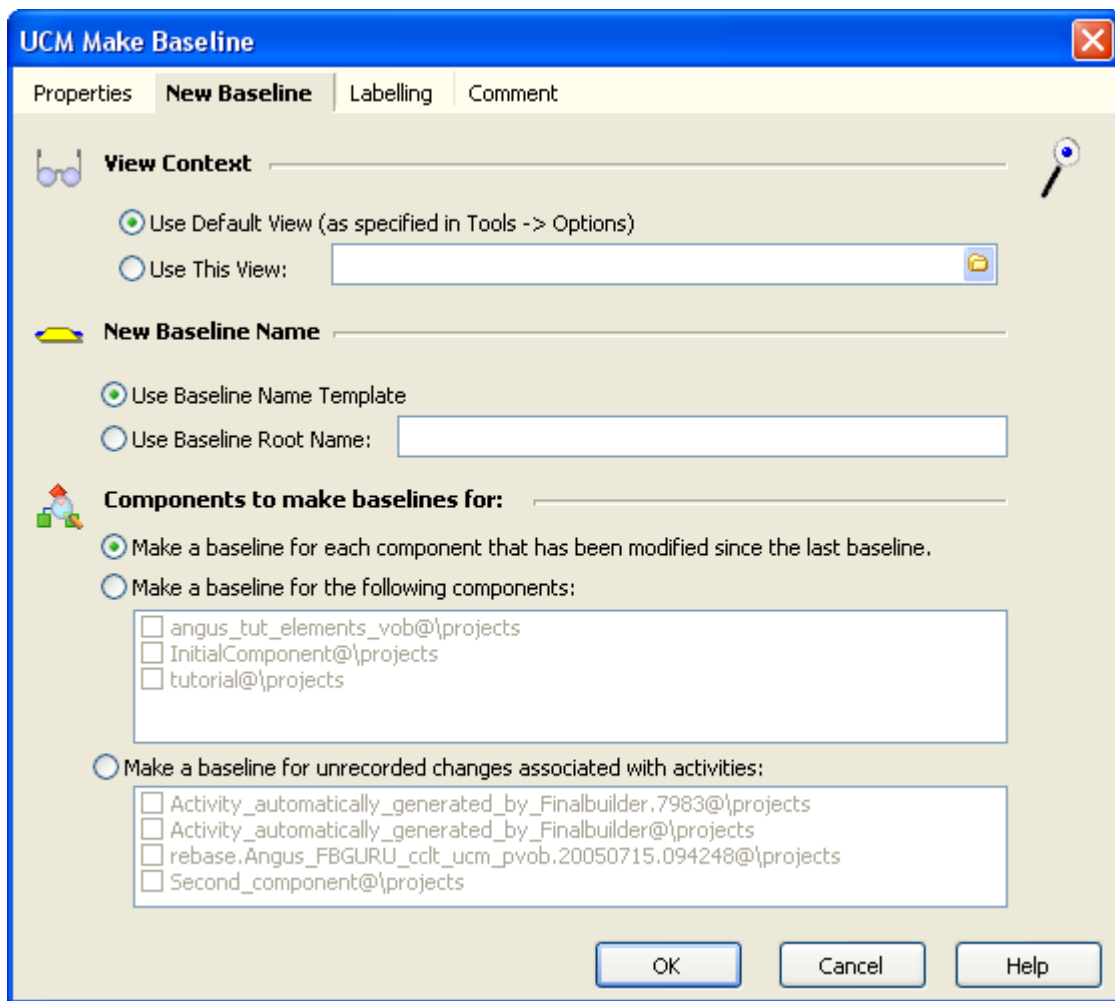


The stream to use can be taken from the default view (specified on the Options panel) or a view specific to that action.

#### 5.2.1.4.5 UCM Make Baseline Action

Use this action to create a new baseline in a UCM stream.

For full details of the `mkbl` command, see the ClearCase manual page (type *cleartool man mkbl*.)



The baseline can be created in either the default view (specified on the Options panel) or a view specific to that action.

### New Baseline Name

If a baseline name template is specified in the config spec, then the new baseline can be named from it.

### Components to make baselines for

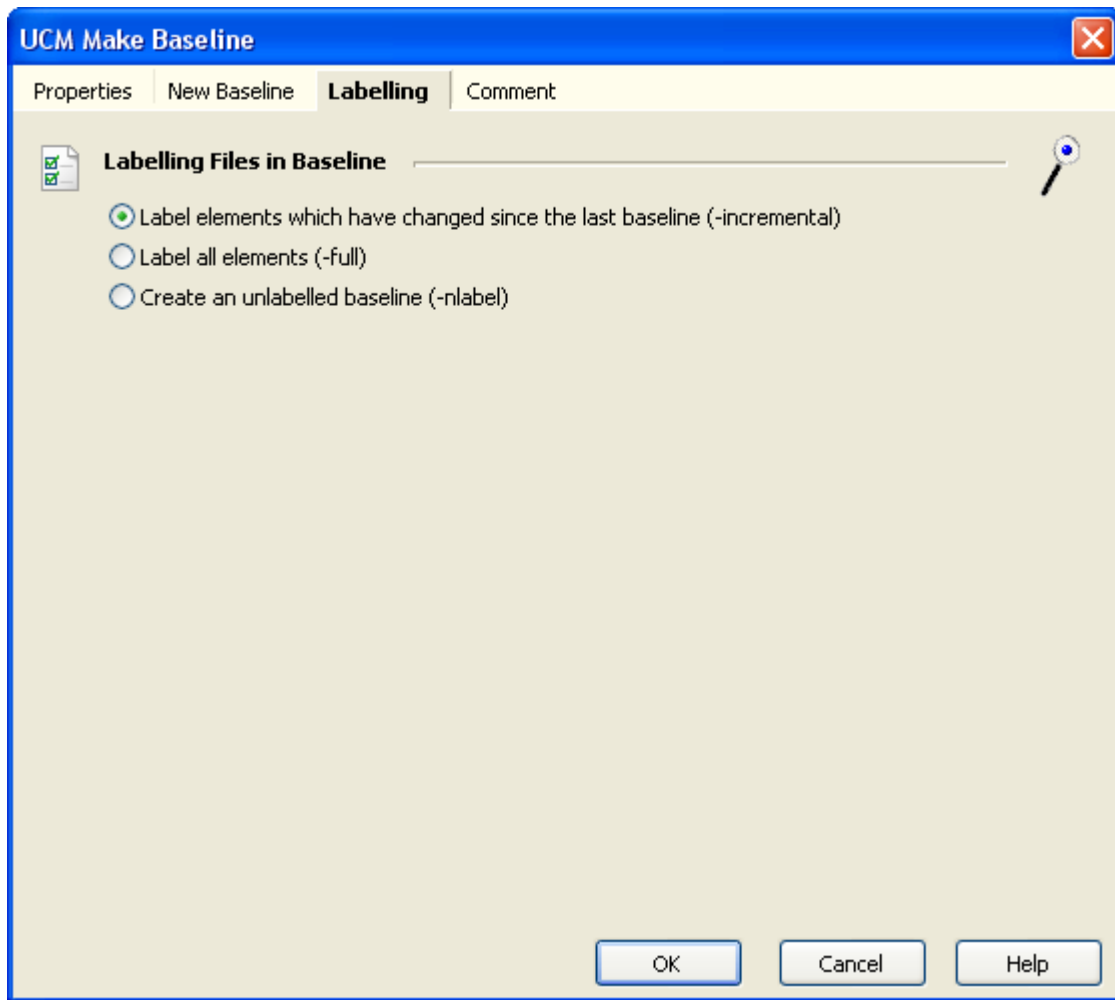
By default, a baseline contains all components that have been modified since the last baseline. However, baselines can be made to depend on components or activities. Baselines can also be made by importing a label type.

Both of the lists (components and activities) are updated with all of the components & activities for the current view. Check any which you wish to select.

If you cannot see the component or activity that you wish to select, change the View



Context to the view that contains that component or activity.

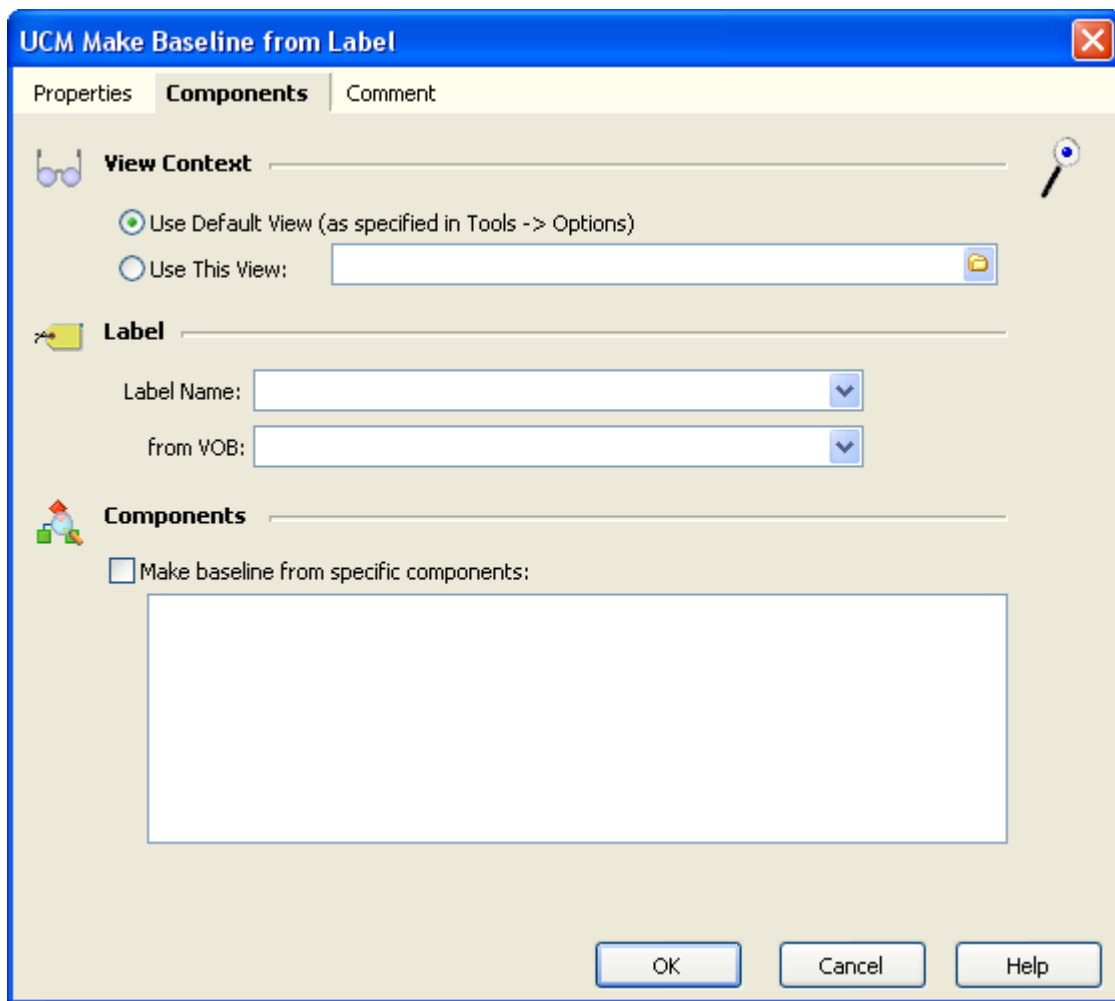


Elements in the baseline can be labelled with the name of the new baseline.

#### 5.2.1.4.6 UCM Make Baseline from Label Action

Use this action to create a new baseline in a UCM stream. The baseline will correspond to a set of labelled versions in the VOB.

For full details of the `mkbl` command, see the ClearCase manual page (type *cleartool man mkbl*.)



The baseline can be created in either the default view (specified on the Options panel) or a view specific to that action.

### Label

Choose a label name. All versions with that label name will be added to the new baseline.

### Components

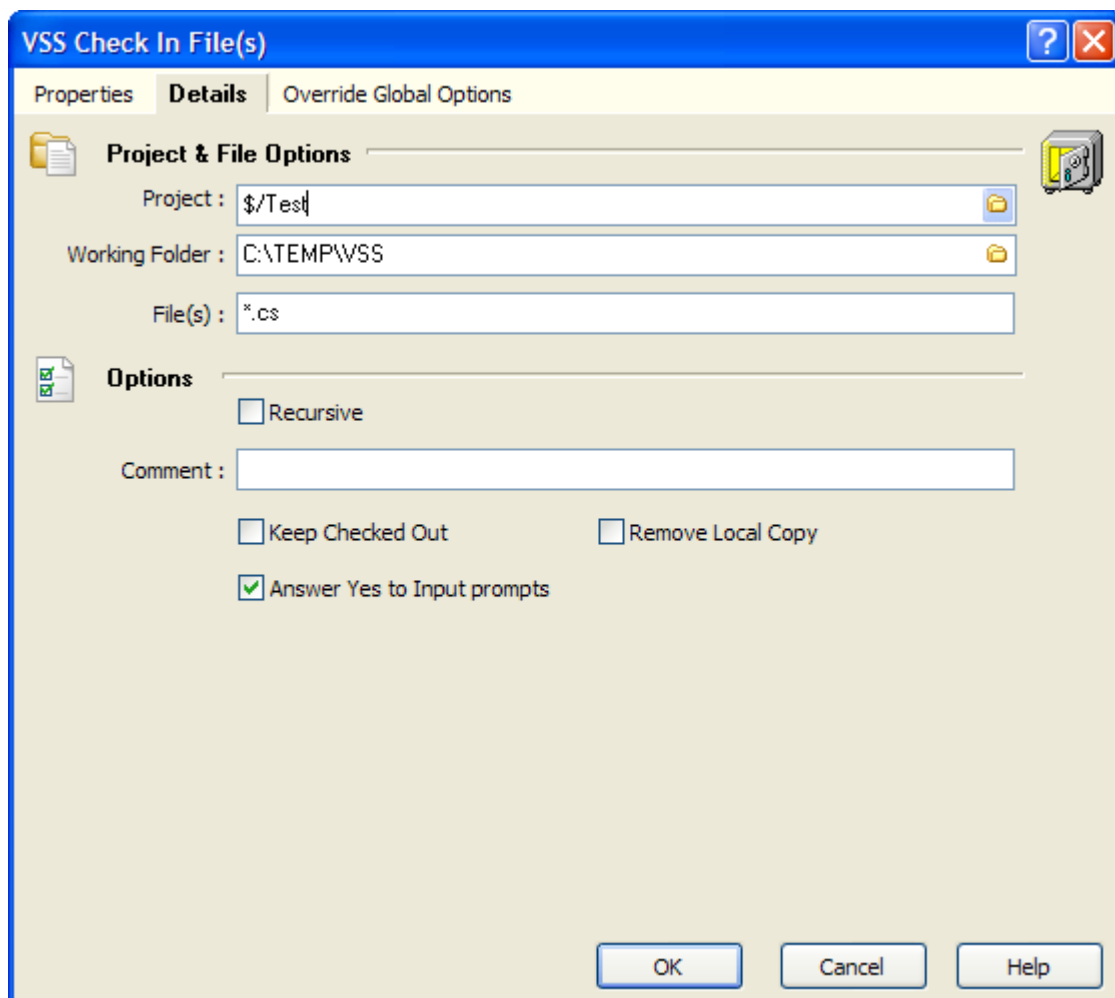
If a list of components is specified, the new baseline will only contain labelled elements from those components.

## 5.2.2 Source Safe

### 5.2.2.1 Source Safe Check In File(s) Action

This action allows you to check in files to a Visual Source Safe database. For detailed descriptions of the options available, see your Visual Source Safe 6.0 documentation or MSDN.

How to override Global Options for the Source Safe actions.



### Scripting Info

The Action properties available are :

- property** SSProject : WideString;// The name of the source safe project, eg. \$/test/subproject
- property** WorkingFolder : WideString;// The folder where the local copy will be placed.
- property** FileSpec : WideString;// The file specification for files to check out , you may use wildcards in this
- property** KeepCheckedOut : WordBool;// Keep File Checked Out.
- property** RemoveLocalCopy : WordBool;// Remove the local copy after check in.
- property** Comment; //Add a comment when checking file in.
- property** Recursive : WordBool;// Recurse sub projects, checking in files/project which match the filespec
- property** OverrideWorkingFolders; //override the working folders when recursively checking in projects
- property** UserID : WideString;// Source Safe User ID, only needed if you do not want to use the default
- property** Password : WideString;// Source Safe pwd for above user id.
- property** IniFile : WideString;// path to source safe ini file (if you do not want to use the default

### 5.2.2.2 Source Safe Check Out File(s) Action

This action allows you to check out files from a Visual Source Safe database. For detailed descriptions of the options available, see your Visual Source Safe 6.0 documentation or MSDN.

How to override Global Options for the Source Safe actions.

### Scripting Info

The Action properties available are :

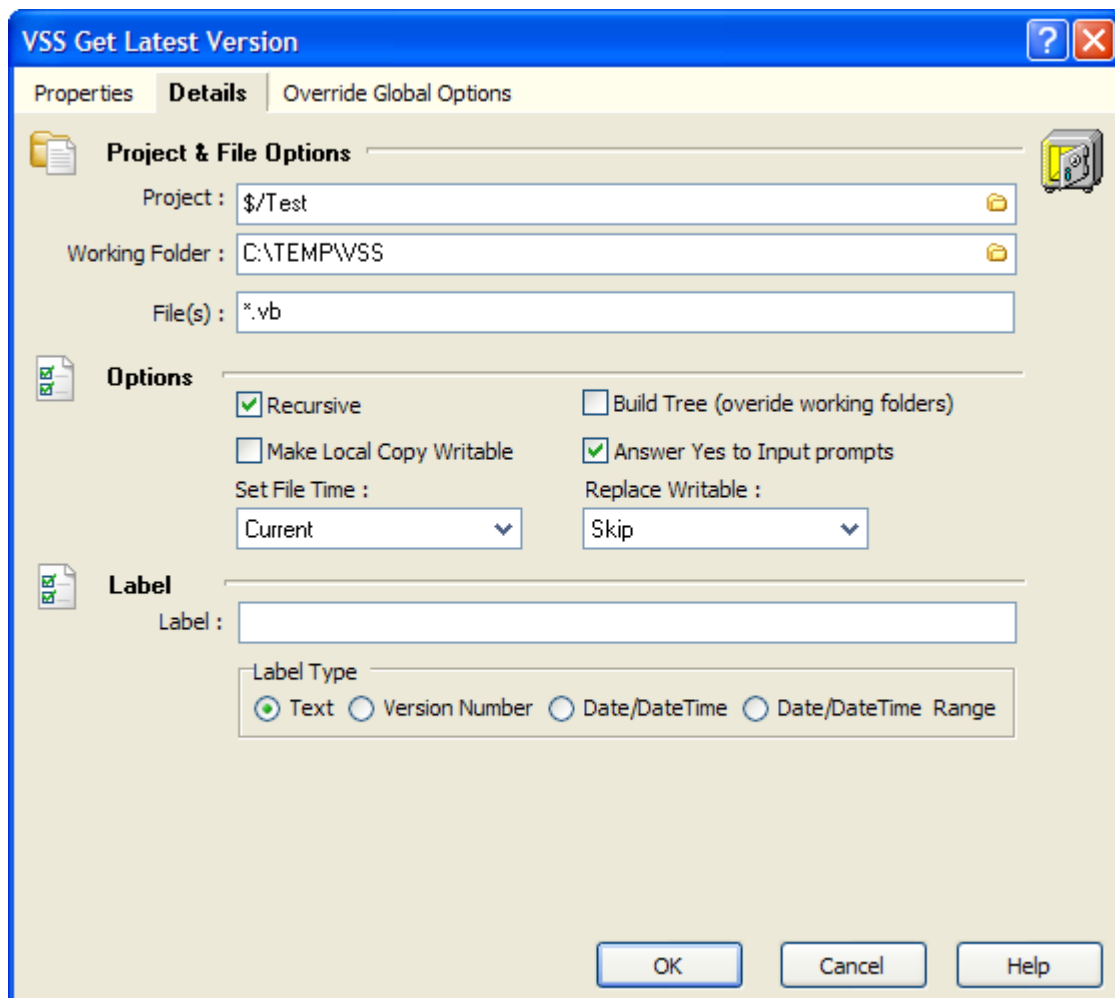
- property** Recursive : WordBool;// Recurse sub projects, checking out files/project which match the filespec
- property** FileSpec : WideString;// The file specification for files to check out , you may use wildcards in this
- property** WorkingFolder : WideString;// The folder where the local copy will be placed.
- property** SSProject : WideString;// The name of the source safe project, eg. \$/test/subproject
- property** BuildTree : WordBool;// Build the local folders using the project names, overrides the working folders
- property** FileTime : integer; // Determines which time to set on files,  
valid values are : ssDefault,ssCurrent,ssModification,ssCheckIn
- property** ReplaceWritable : integer;// Determines how to handle local files that are already writeable,

// valid values are : ssReplace, ssSkip, ssMerge  
**property** UserID : WideString;// Source Safe User ID, only needed if you do not want to use the default  
**property** Password : WideString;// Source Safe pwd for above user id.  
**property** IniFile : WideString;// path to source safe ini file (if you do not want to use the default

### 5.2.2.3 Source Safe Get Latest Version Action

This action allows you to Get the Latest Version of files or projects from a Visual Source Safe database. For detailed descriptions of the options available, see your Visual Source Safe 6.0 documentation or MSDN.

How to override Global Options for the Source Safe actions.



You can specify a date string in the Label field. If a date and time is specified, then source safe will get the versions of the specified files with that exact date & time, if there is no version with that date & time then the file will not be retrieved. If you specify just a date, then source safe will get the latest versions of the specified files as at the specified date.

### Scripting Info

The Action properties available are :

**property** SSProject : WideString;// The name of the source safe project, eg. \$/test/subproject  
**property** WorkingFolder : WideString;// The folder where the local copy will be placed.  
**property** FileSpec : WideString;// The file specification for files to check out , you may use wildcards in this  
**property** KeepCheckedOut : WordBool;// Keep File Checked Out.  
**property** RemoveLocalCopy : WordBool;// Remove the local copy after check in.  
**property** Comment; //Add a comment when checking file in.  
**property** Recursive : WordBool;// Recurse sub projects, checking in files/project which match the filespec  
**property** BuildTree : WordBool;// Build the local folders using the project names, overrides the working folders  
**property** FileTime : integer; // Determines which time to set on files,  
valid values are : ssDefault,ssCurrent,ssModification,ssCheckIn  
**property** ReplaceWritable : integer;// Determines how to handle local files that are already writeable,  
// valid values are : ssReplace, ssSkip, ssMerge  
**property** UserID : WideString;// Source Safe User ID, only needed if you do not want to use the default  
**property** Password : WideString;// Source Safe pwd for above user id.  
**property** IniFile : WideString;// path to source safe ini file (if you do not want to use the default

#### 5.2.2.4 Source Safe Label File(s)

This action allows you to Label the latest version of a project or files in a Visual Source Safe database. For detailed descriptions of the options available, see your Visual Source Safe 6.0 documentation or MSDN.

How to override Global Options for the Source Safe actions.

The screenshot shows the 'VSS Label File(s)' dialog box with the 'Details' tab selected. The dialog has three tabs: 'Properties', 'Details', and 'Override Global Options'. The 'Project & File Options' section contains a 'Project' field with the value '\$/Test' and a 'Filespec' field with the value '\*. \*'. The 'Options' section contains a 'New Label' field with the value 'Build %BUILDNO%', an 'Apply To Label' field, a 'Label Type' section with radio buttons for 'Text' (selected), 'Version Number', 'Date/DateTime', and 'Date/DateTime Range', a 'Comment' field, and a checked checkbox for 'Answer Yes to Input prompts'. The dialog has 'OK', 'Cancel', and 'Help' buttons at the bottom.

### Scripting Info

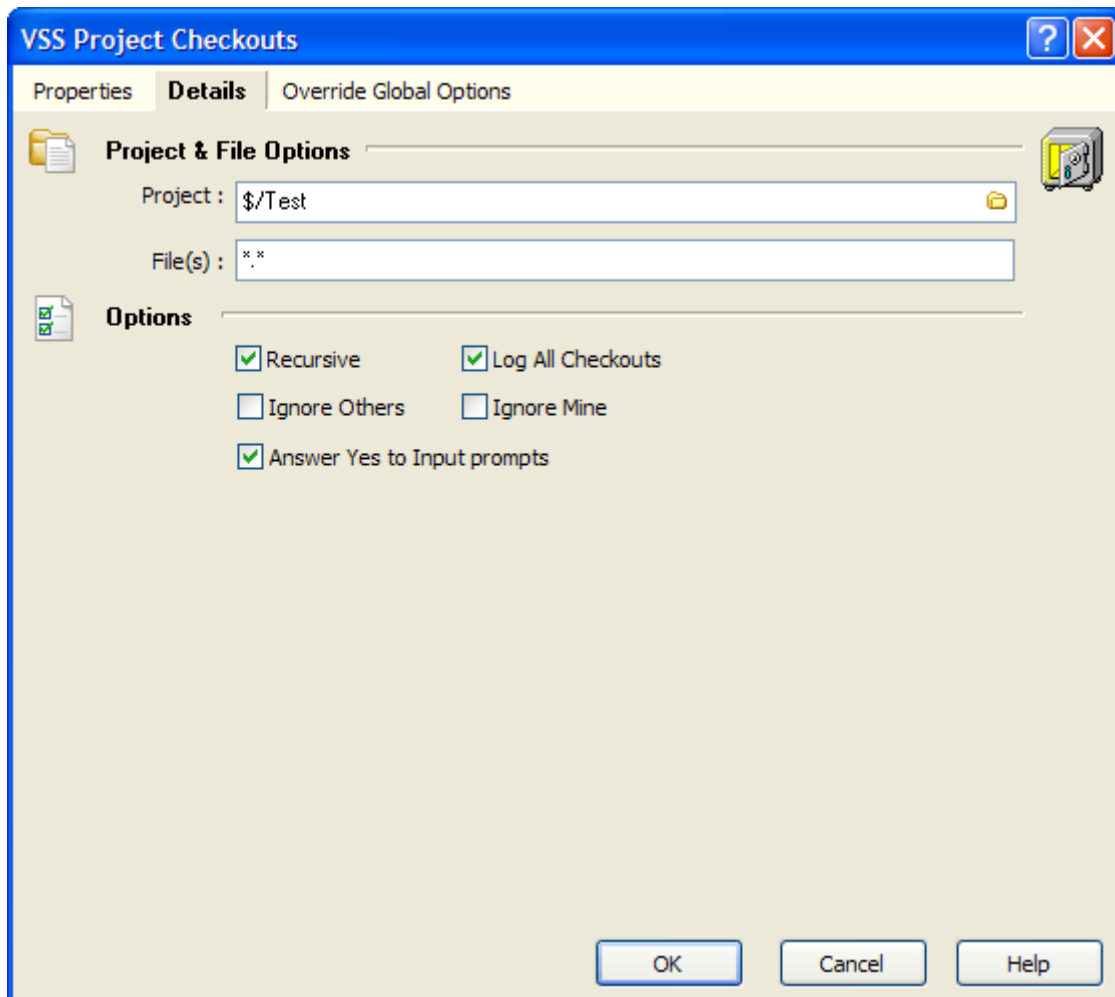
The Action properties available are :

- property** SSProject : WideString;// The name of the source safe project, eg. \$/test/subproject
- property** SSLabel : WideString;// The Label to apply
- property** FileSpec : WideString;// The file specification for files to label , you may use wildcards in this after check in.
- property** Comment; //Add a comment when Labelling a file or project.

#### 5.2.2.5 Source Safe Project Checkouts

This action checks a Source Safe project to see if any files are checked out. If it finds any files checked out then it will fail. You can choose to ignore files checked out by yourself.

How to override Global Options for the Source Safe actions.

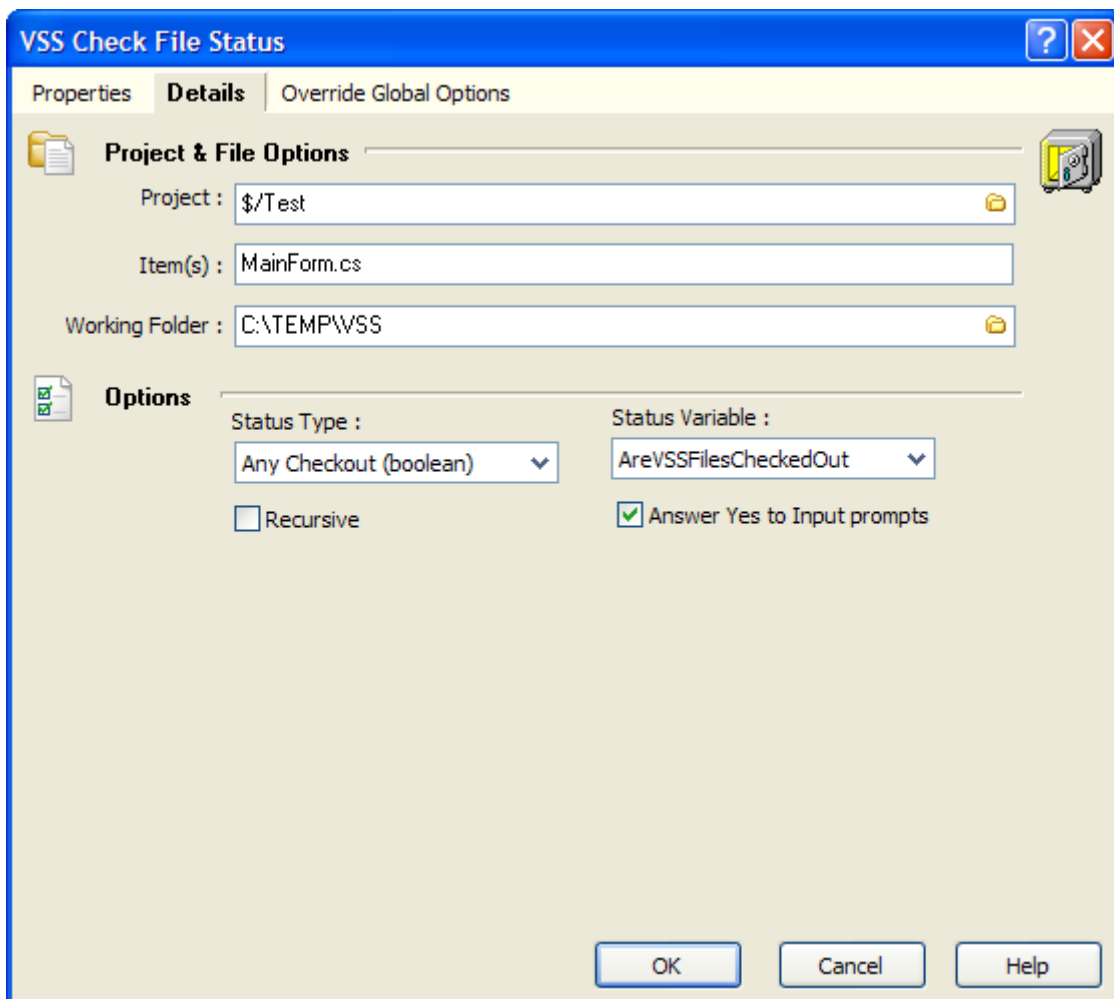


#### 5.2.2.6 Source Safe Check File Status

This Action allows you to check the status of files in a source safe project. The status is read into a FinalBuilder variable which can then be used elsewhere in the build.

How to override Global Options for the Source Safe actions.

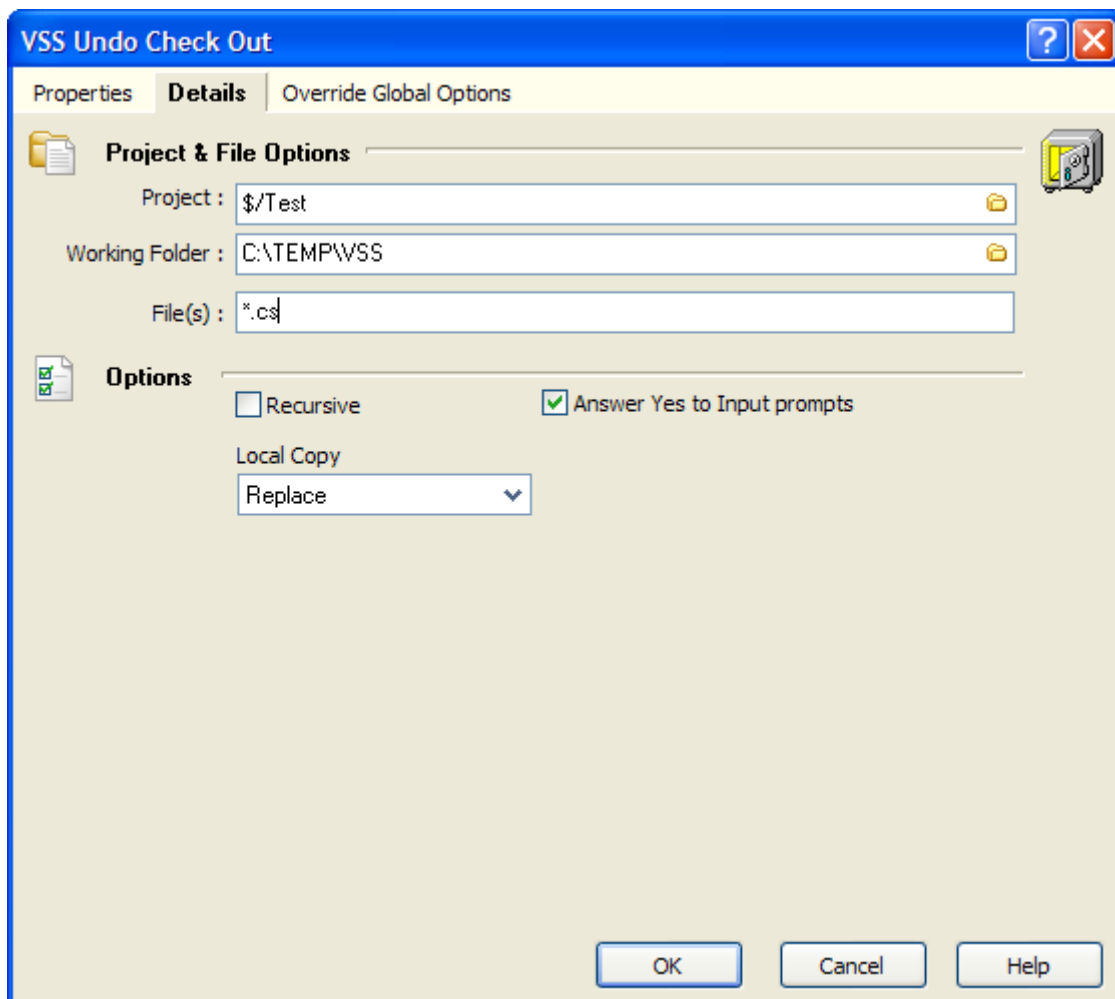




#### 5.2.2.7 Source Safe Undo CheckOut Action

This action allows you to undo the check out of files from a Visual Source Safe database. For detailed descriptions of the options available, see your Visual Source Safe 6.0 documentation or MSDN.

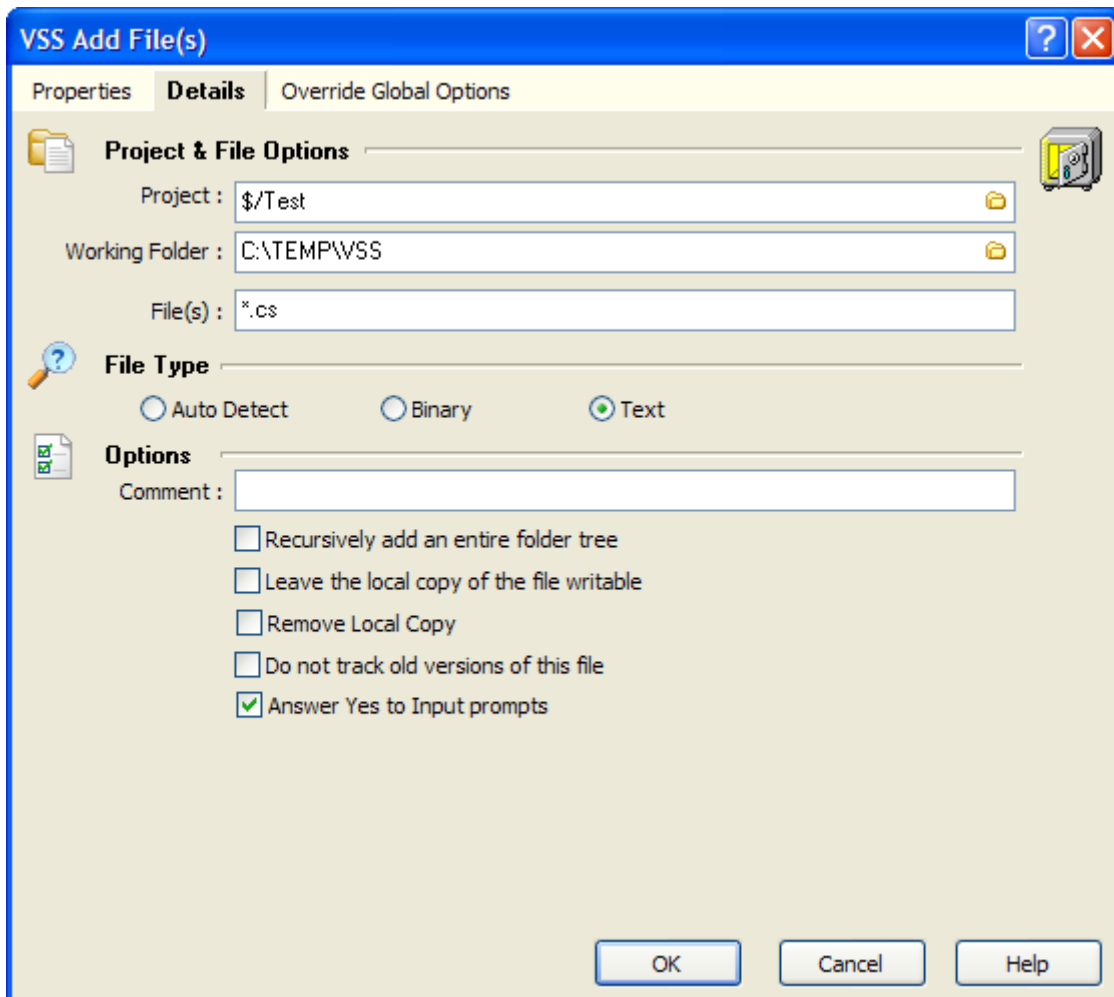
How to override Global Options for the Source Safe actions.



#### 5.2.2.8 Source Safe Add Files Action

Adds Files to a Source Safe Project.

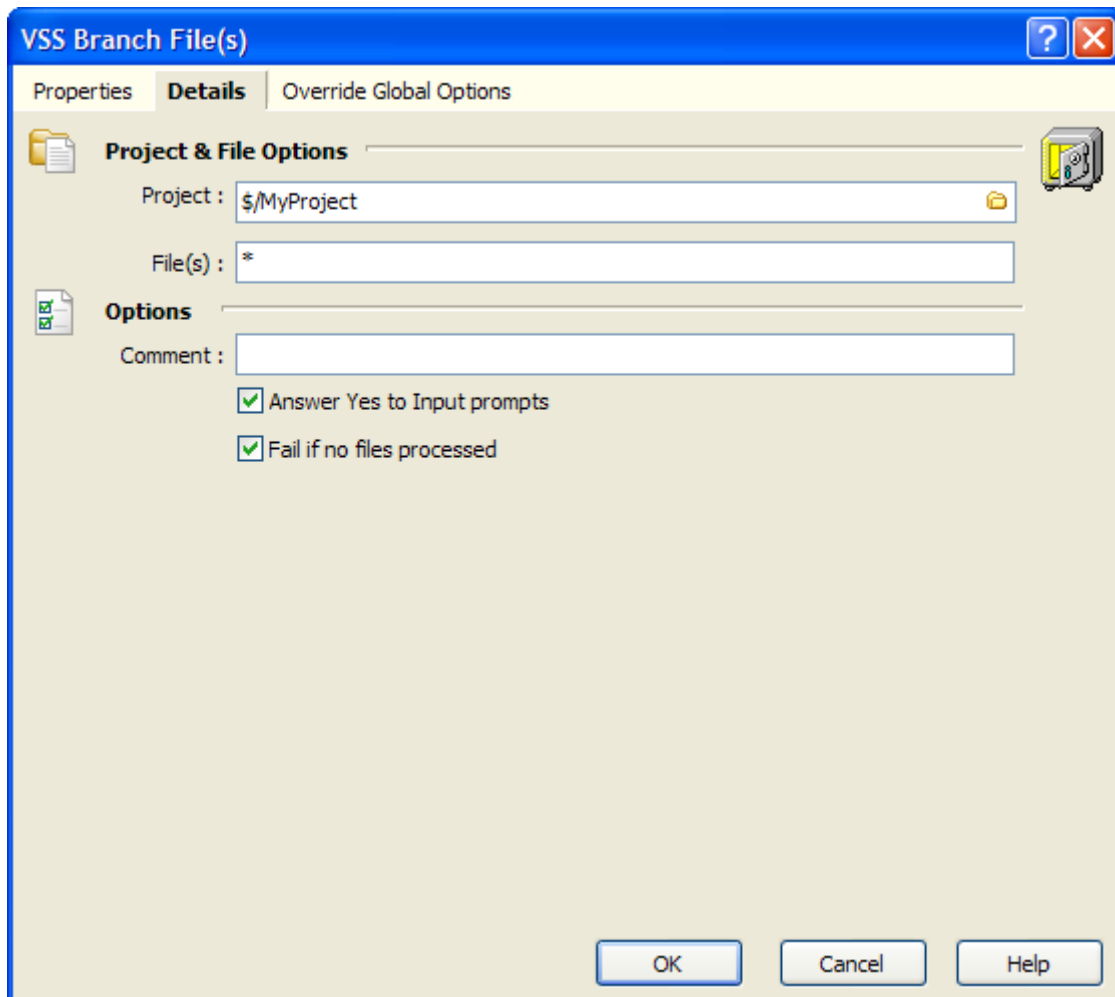
How to override Global Options for the Source Safe actions.



#### 5.2.2.9 Source Safe Branch

The VSS Branch Files action will branch shared files in the specified project.

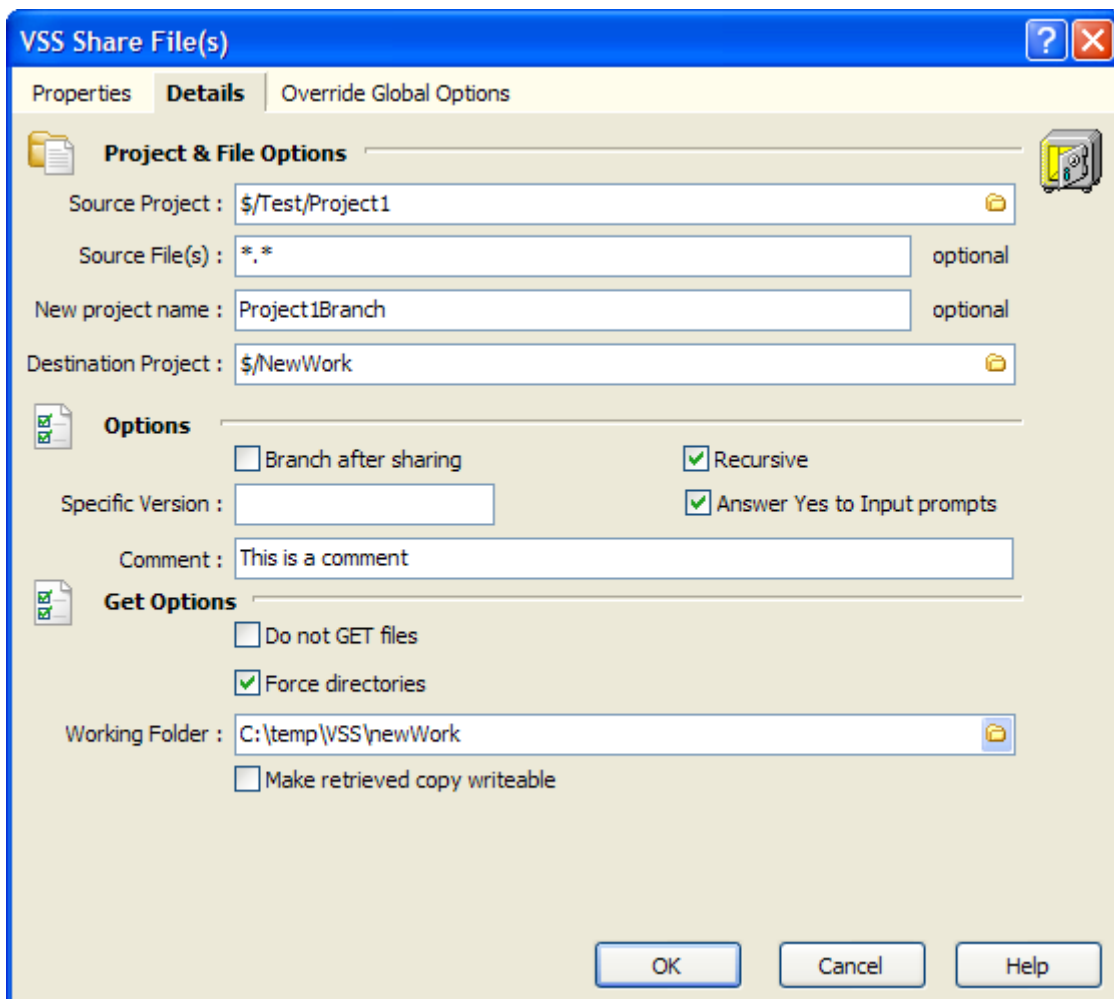
Branching severs the link between the original copy of the file and the shared file. See Source Safe Share



#### 5.2.2.10 Source Safe Share

The VSS Share files action enables you to share a project or files to a new project in Source Safe.

Sharing files does not create a new file, but creates a link back to the original file. You can branch the shared files immediately by specifying the "Branch after sharing" option. Or you can branch later using the Source Safe Branch action.



#### 5.2.2.11 Source Safe Get Working Directory

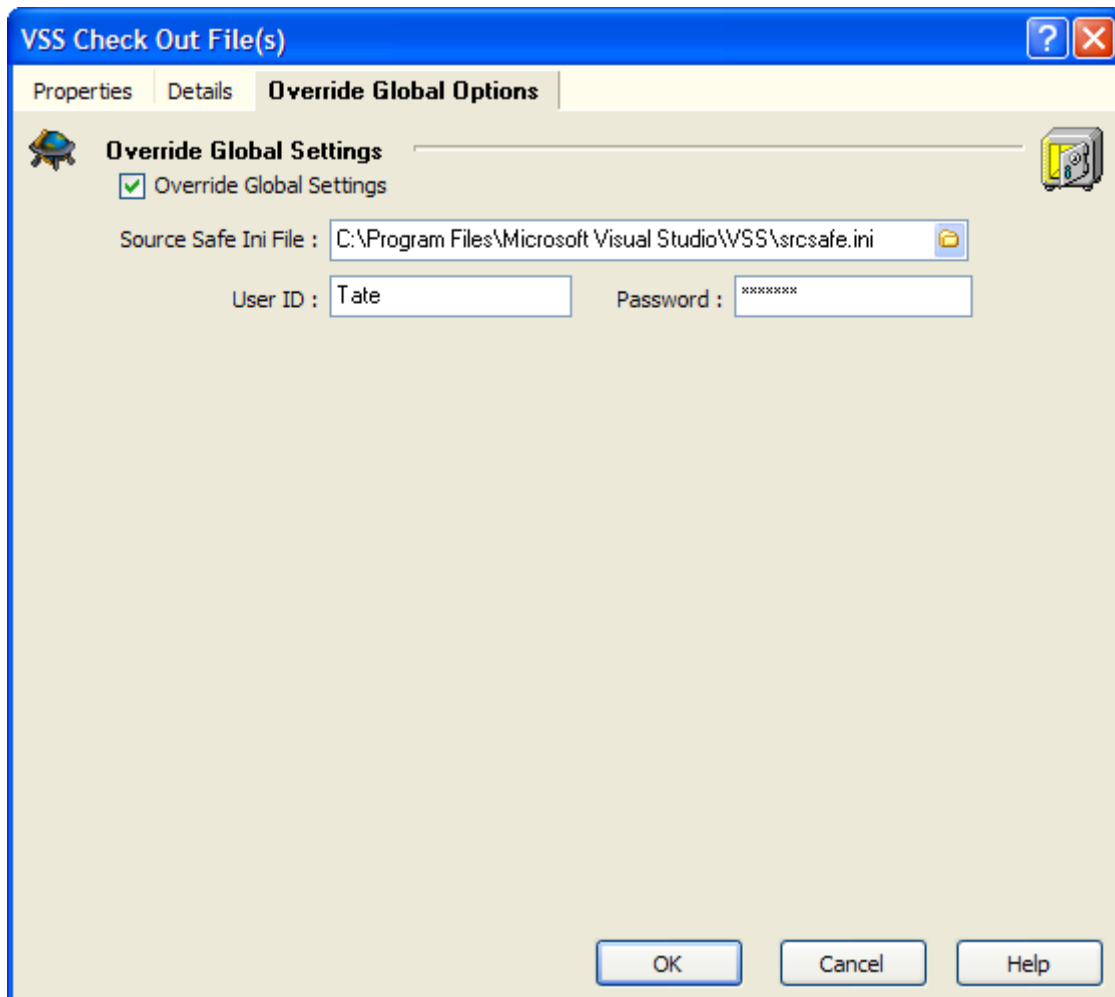
The Get Working Directory action attempts to retrieve the working directory for the specified project using the source safe ini files.

The working directory will be output to the FB Log and optionally set a variable with the value.

#### 5.2.2.12 Source Safe Override Global Options

Each of the source safe actions allow you to override the global options. This can be used to access more than one Source Safe database.

To set the global options, open the Options dialog (Tools menu) and then open the Version Control Systems category and select Visual Source Safe. The global options will then be used for any VSS action which doesn't have the "Override Global Settings" turned on.



### 5.2.3 Perforce

#### 5.2.3.1 Perforce Options

Perforce help coming soon.

Note: it is advised not to use Perforce environment variables in the various Perforce fields (in Perforce options or in the Perforce actions). Instead, define your own FinalBuilder variables and set them to the required values.

#### 5.2.3.2 Perforce Create Branch

Perforce help coming soon.

#### 5.2.3.3 Perforce Delete Branch

Perforce help coming soon.

#### 5.2.3.4 Perforce Sync

Perforce help coming soon.

**5.2.3.5 Perforce Submit**

Perforce help coming soon.

**5.2.3.6 Perforce open for Edit**

Perforce help coming soon.

**5.2.3.7 Perforce Opened**

Perforce help coming soon.

**5.2.3.8 Perforce Create Label**

Perforce help coming soon.

**5.2.3.9 Perforce Update Label**

Perforce help coming soon.

**5.2.3.10 Perforce Delete Label**

Perforce help coming soon.

**5.2.3.11 Perforce Labelsync**

Perforce help coming soon.

**5.2.3.12 Perforce open for Delete**

Perforce help coming soon.

**5.2.3.13 Perforce Tag**

Perforce help coming soon.

**5.2.3.14 Perforce Lock**

Perforce help coming soon.

**5.2.3.15 Perforce Unlock**

Perforce help coming soon.

**5.2.3.16 Perforce Revert**

Perforce help coming soon.

**5.2.3.17 Perforce Create Changelist**

Perforce help coming soon.

**5.2.3.18 Perforce Delete Changelist**

Perforce help coming soon.

**5.2.3.19 Perforce Generic**

Perforce help coming soon.

### 5.2.3.20 Perforce Old Actions

The old set of Perforce actions has been deprecated.

The help topics and actions are still available, but we encourage Perforce users to use the new set of actions.

#### 5.2.3.20.1 Perforce Synchronise with View Action

Copy files from the depot into the workspace.

The sync command can sync to either the Head Revision, Revision Number, Label, Change Number or a specific Date/Time.

**Sync to Revision**

Properties **Sync To Revision**

Perforce path :   
(eg. //depot/myproject)

Filespec :

Force : ☐ (resynchronisation and clobbers writable files)

Version

Head Revision : ☒

Revision Number : ☐

Label : ☐

Change Number : ☐

Date/Time : ☐

Page : Sync To Revision

For all other Perforce commands use the "Perforce Command" and enter the command line options manually.

For more information on the sync command, see:

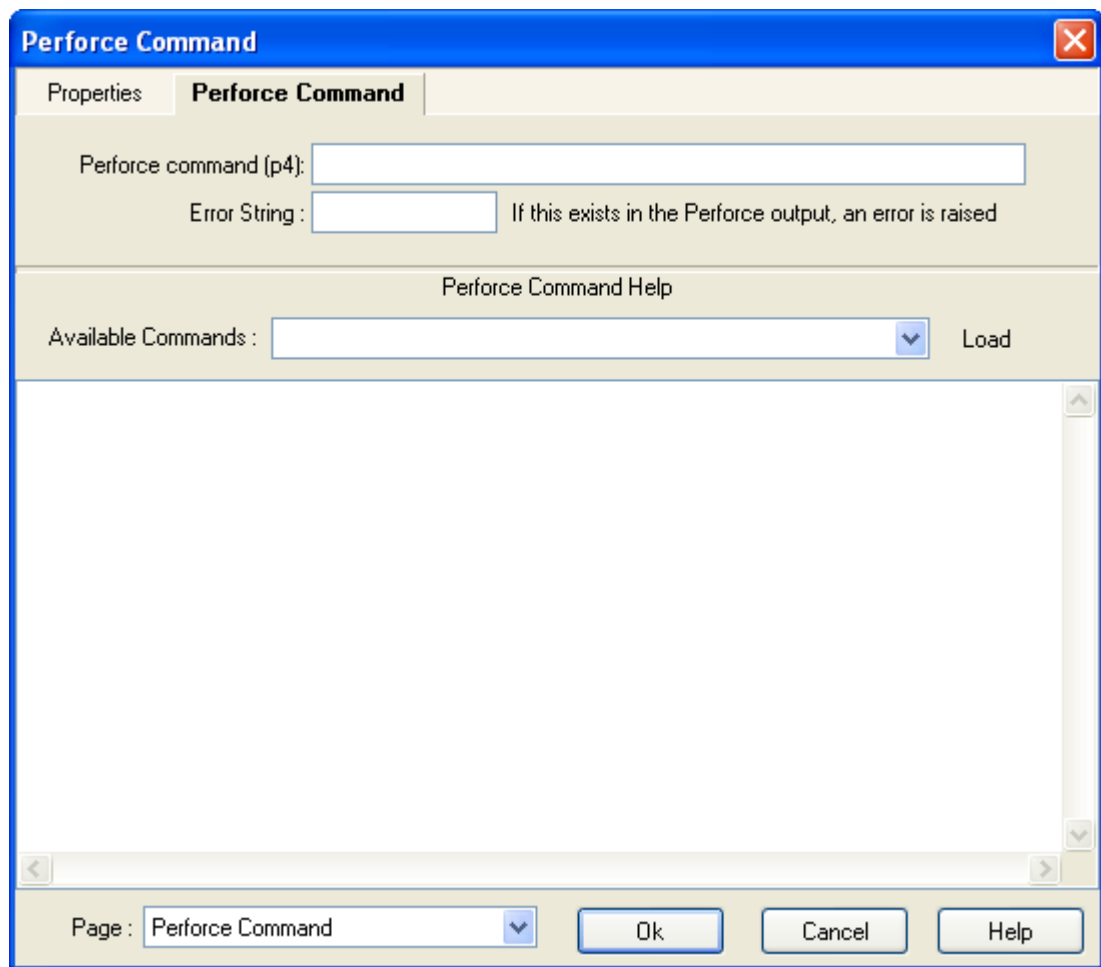


<http://www.perforce.com/perforce/doc.032/manuals/cmdref/sync.html>

This action was kindly provided by Tate Needham from t8software.com

#### 5.2.3.20.2 Perforce Command Action

This action allows you to run most Perforce commands. Note that command which prompt you for a response will not work and cause the action to hang.



The Command Reference can be found here:

<http://www.perforce.com/perforce/doc.032/manuals/cmdref/index.html>

This action was kindly provided by Tate Needham from t8software.com

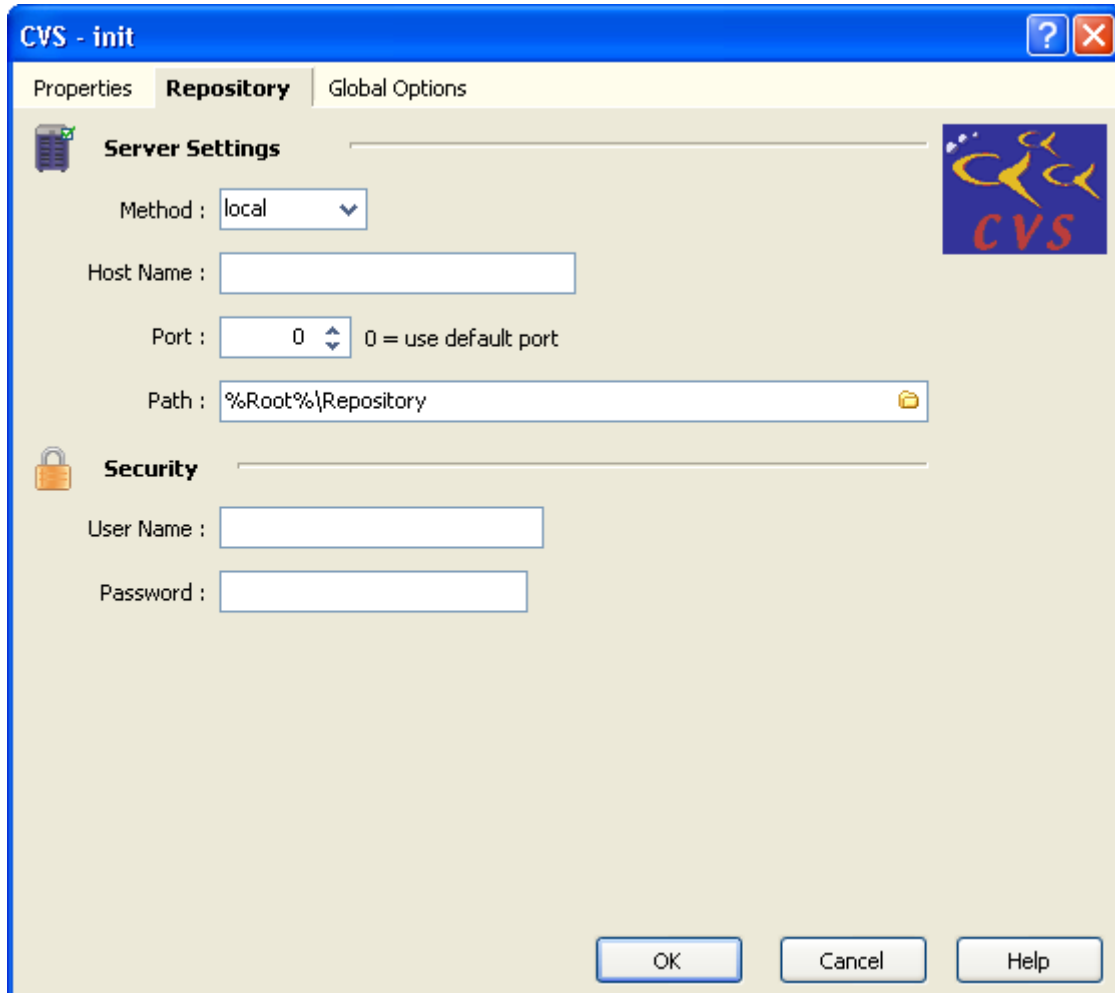
## 5.2.4 CVS Actions

The CVS Actions provide a basic interface over the CVS command line tool, cvs.exe. Not all options are exposed in these action's property pages, however they do allow you to

specify additional options to be passed to the command line.

All of the CVS actions have the Repository and Global Options property pages. These pages expose the options that control how you connect to CVS (either remotely or locally) and provide access to some properties that are common across most of the CVS actions.

For information on how to use CVS, see the CVS documentation available here : <http://www.cvshome.org>



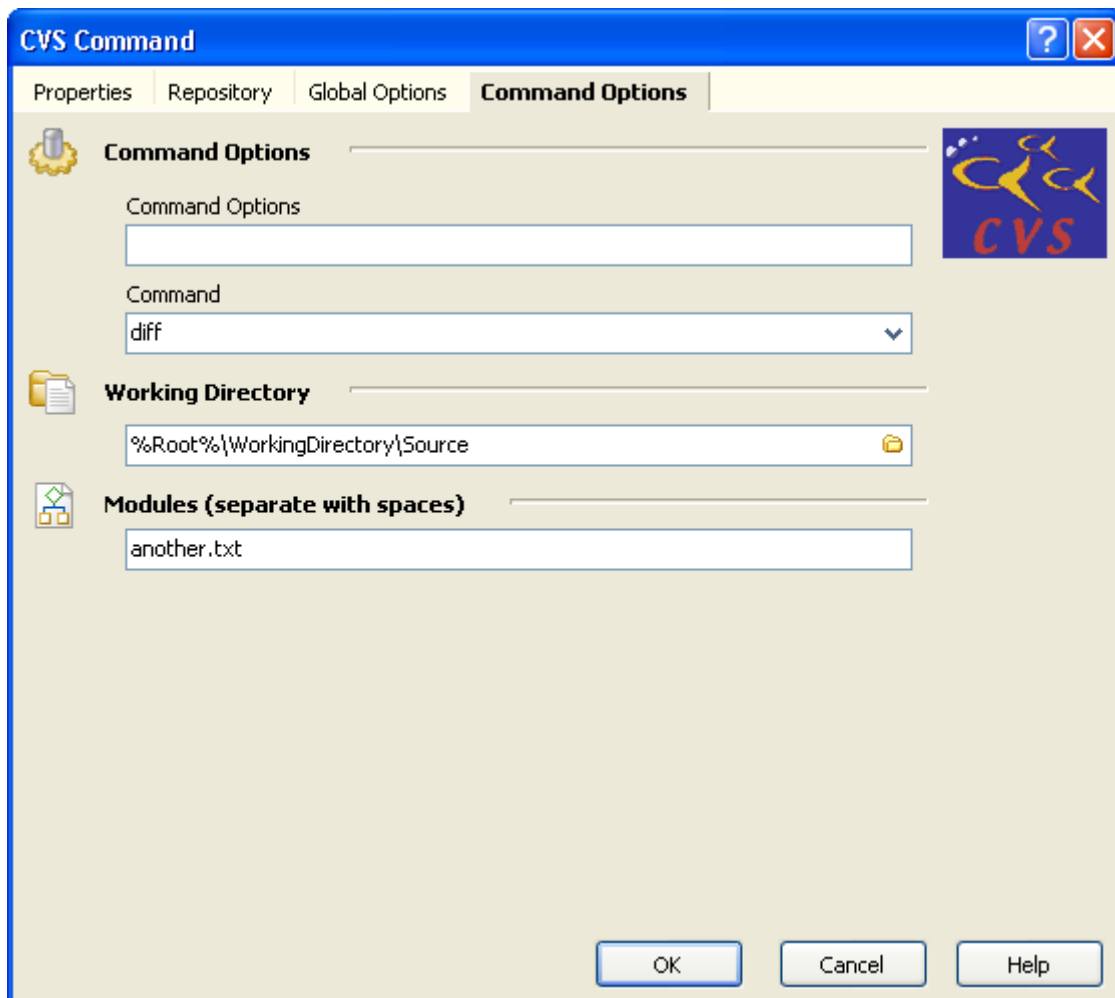
There is an example Build script that demonstrates the usage of some of the CVS actions in the Finalbuilder\Examples\CVS Directory installed with FinalBuilder.

**Note :** CVS.exe requires that two environment variables HOMEDRIVE and HOMEPATH are set before running it. When running from the scheduler, make sure you run under a user that has these environment variables set, or add them as FinalBuilder project variables and check the "Make available as Environment variable" option when adding the variables. Typical values for these variables are :

HOMEDRIVE=E:  
HOMEPATH=\Documents and Settings\Vincent

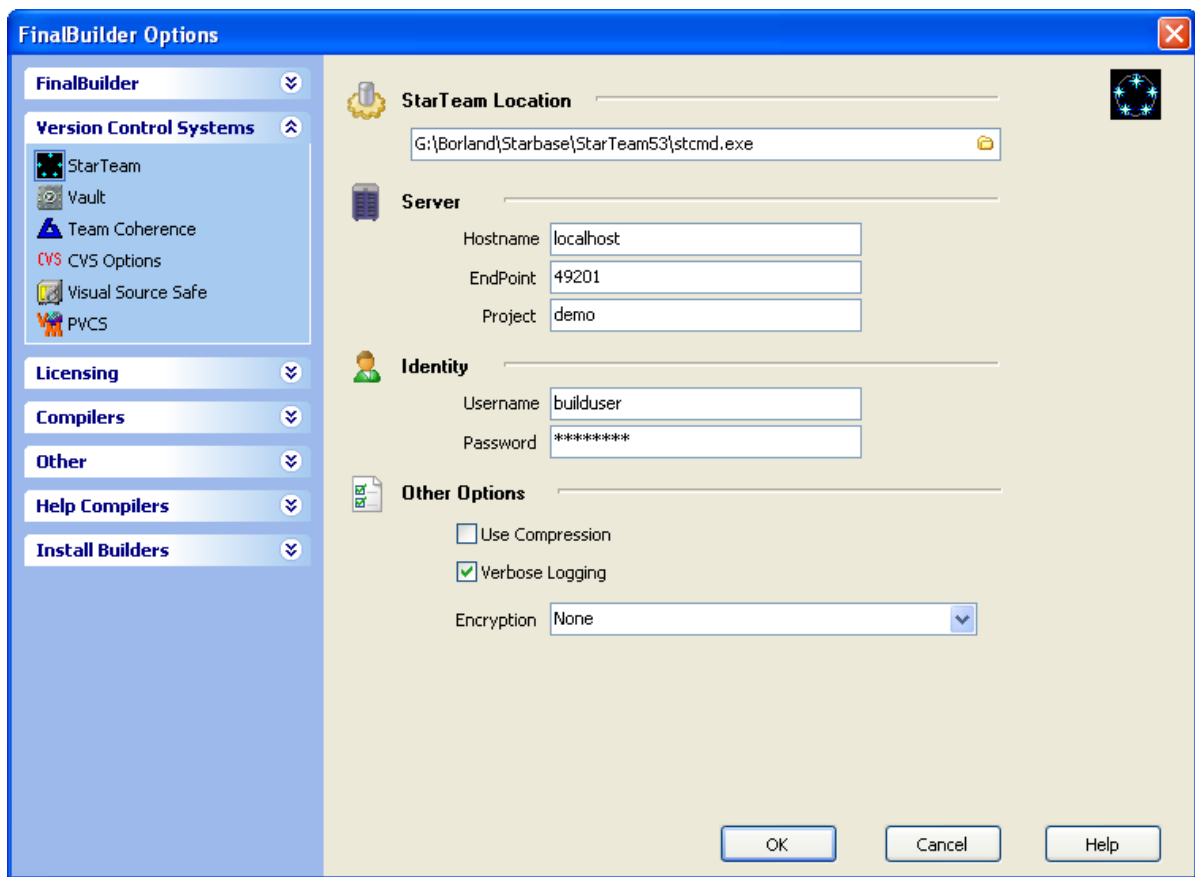
### 5.2.4.1 CVS Command Action

This action provides a generic CVS command interface. The list of commands in the Command drop down list is not the only commands that can be called, you can type the name of the command you wish to call into the combo box.



### 5.2.5 Borland StarTeam Actions

These Actions support Borland StarTeam 5.2.x or higher. They provide an easy to use wrapper around the StarTeam command line tool. Before using these actions you need to check that the path to the StarTeam command line tool is set in the FinalBuilder Options Dialog. FinalBuilder will attempt to find the path automatically however it is not always possible (depends on your installation).



You can also set defaults which will be used for the StarTeam actions in your projects, you can override the defaults if needed in the individual Actions.

The screenshot shows a Windows-style dialog box titled "Check In". It has four tabs: "Properties", "Check In", "More Options", and "Override Global Options", with the last tab being active. At the top left of the dialog is a checked checkbox labeled "Override Default Options". To the right of this checkbox is a small icon of a starburst. Below this, the dialog is divided into three sections, each with a header and a horizontal line separator. The first section is "Server", indicated by a server rack icon, and contains three text boxes: "Hostname" with "localhost", "EndPoint" with "49201", and "Project" with "finalbuilder2". The second section is "Identity", indicated by a person icon, and contains two text boxes: "Username" with "builduser" and "Password" with "\*\*\*\*\*". The third section is "Other Options", indicated by a document icon with checkmarks, and contains two checkboxes: "Use Compression" (unchecked) and "Verbose Logging" (checked), followed by an "Encryption" dropdown menu set to "None". At the bottom right of the dialog are three buttons: "OK", "Cancel", and "Help".

#### 5.2.5.1 StarTeam Check In Action

This action calls the StarTeam command line tool to check in files.

#### 5.2.5.2 StarTeam Check Out Action

This action calls the StarTeam command line tool to check out files.

#### 5.2.5.3 StarTeam Lock/Unlock Files Action

This action calls the StarTeam command line tool to Lock/Unlock files.

#### 5.2.5.4 StarTeam Create Label Action

This action calls the StarTeam command line tool to create a Label

#### 5.2.5.5 StarTeam Apply Label Action

This action calls the StarTeam command line tool to Apply a Label

**5.2.5.6 StarTeam Update Status Action**

This action calls the StarTeam command line tool to Update the status of files.

**5.2.5.7 StarTeam Delete Files Action**

This action calls the StarTeam command line tool to delete files.

**5.2.5.8 StarTeam Add Files Action**

This action calls the StarTeam command line tool to Add files.

**5.2.5.9 StarTeam List Files Action**

This action calls the StarTeam command line tool to List files.

**5.2.6 QVCS Actions****5.2.6.1 QVCS Add File Action**

The QVCS Add File action enables you to add files to your source code repository using QVCS as part of your build process.

For more information on QVCS, see <http://www.qumasoft.com/>

**5.2.6.2 QVCS Check In File(s) Action**

The QVCS Check In File(s) action enables you to check in files to your source code repository using QVCS as part of your build process.

For more information on QVCS, see <http://www.qumasoft.com/>

**5.2.6.3 QVCS Check Out File(s) Action**

The QVCS Check Out File(s) action enables you to check out files from your source code repository using QVCS as part of your build process.

For more information on QVCS, see <http://www.qumasoft.com/>

**5.2.6.4 QVCS Get Latest Version Action**

The QVCS Get Latest Version action enables you to synchronise your local files to the latest version from your source code repository using QVCS as part of your build process.

For more information on QVCS, see <http://www.qumasoft.com/>

**5.2.6.5 QVCS Labels File(s) Action**

The QVCS Label File(s) action enables you to assign version labels to your files in your source code repository using QVCS as part of your build process.

For more information on QVCS, see <http://www.qumasoft.com/>

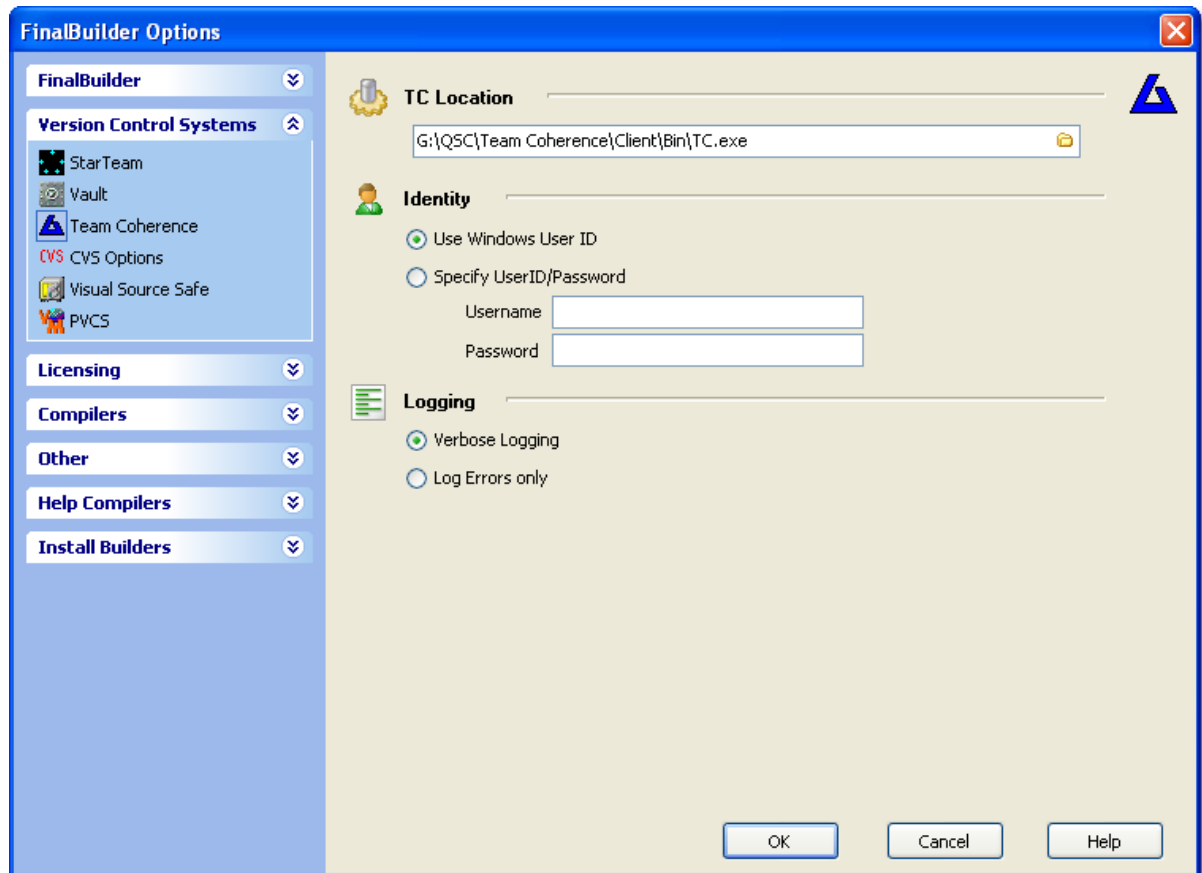
**5.2.6.6 QVCS Undo Check Out File(s) Action**

The QVCS Undo Check Out File(s) action enables you to undo checkouts of specified files using QVCS as part of your build process.

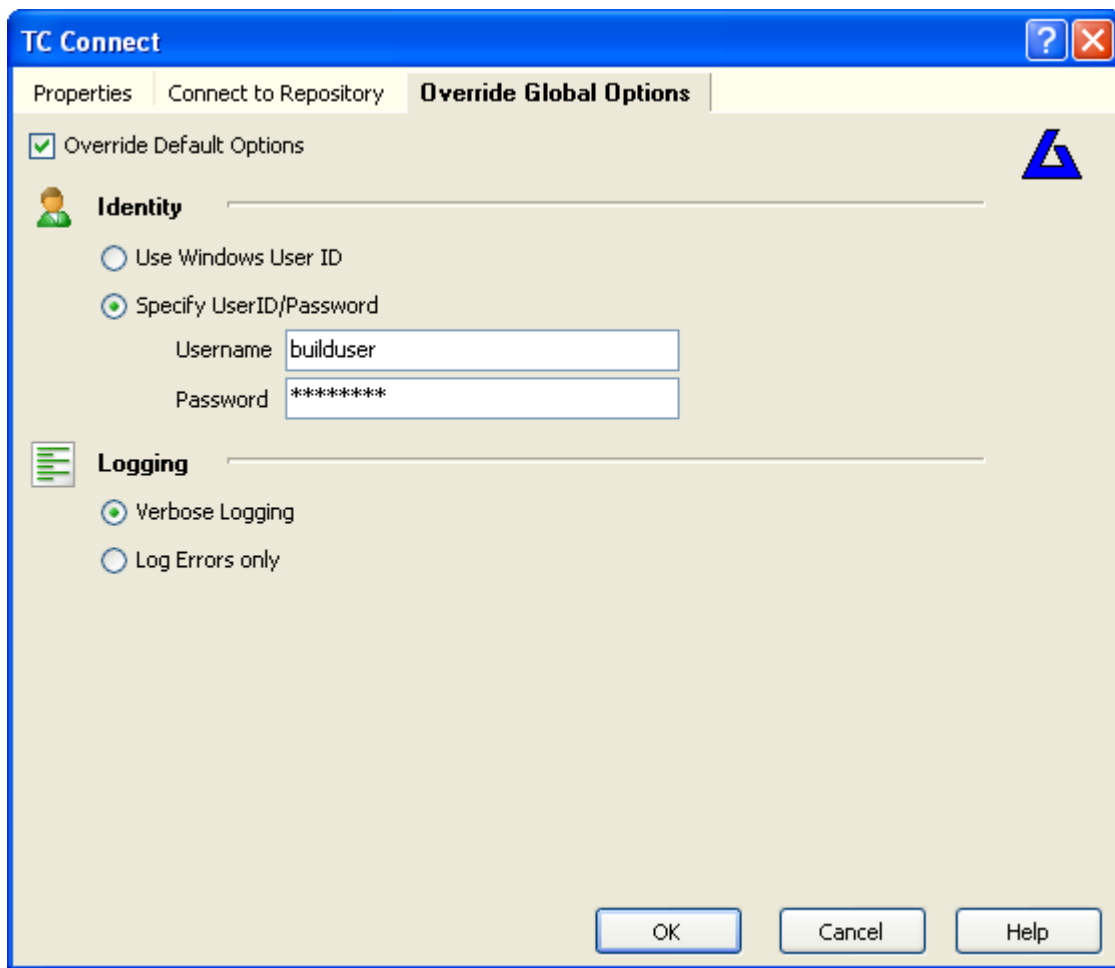
For more information on QVCS, see <http://www.qumasoft.com/>

## 5.2.7 QSC Team Coherence Actions

These Actions support Team Coherence 7.1 or higher. They provide an easy to use wrapper around the TC command line tool. Before using these actions you need to check that the path to the TC command line tool is set in the FinalBuilder Options Dialog. FinalBuilder will attempt to find the path automatically however it is not always possible (depends on your installation).



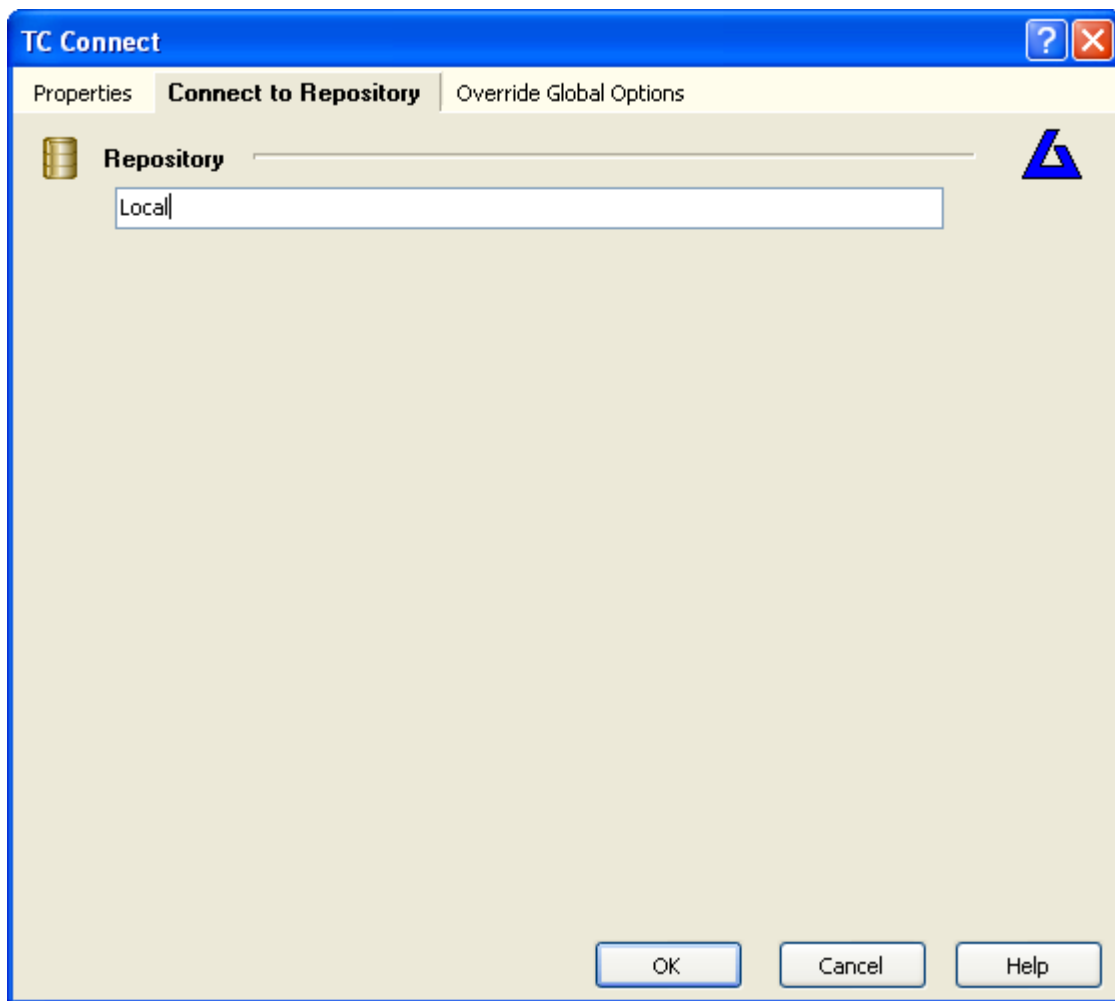
The default options can be overridden in the individual actions.



#### 5.2.7.1 Team Coherence Connect Action

Connects to a different repository.





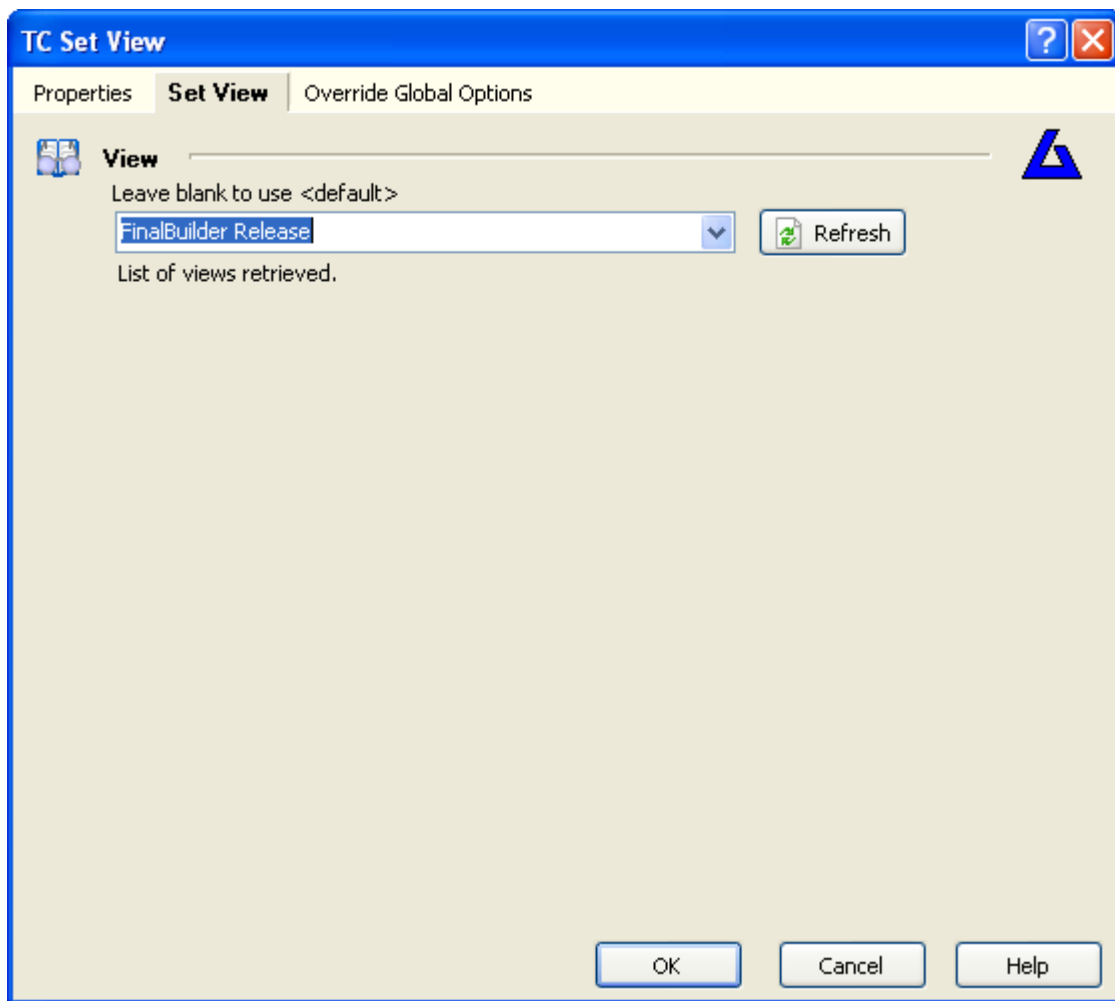
### Remarks

When you connect to a different repository, it becomes the default for subsequent commands.

Note that you must be connecting to a previously created repository connection. You can define connections using the Team Coherence Version Manager.

#### 5.2.7.2 Team Coherence Set View Action

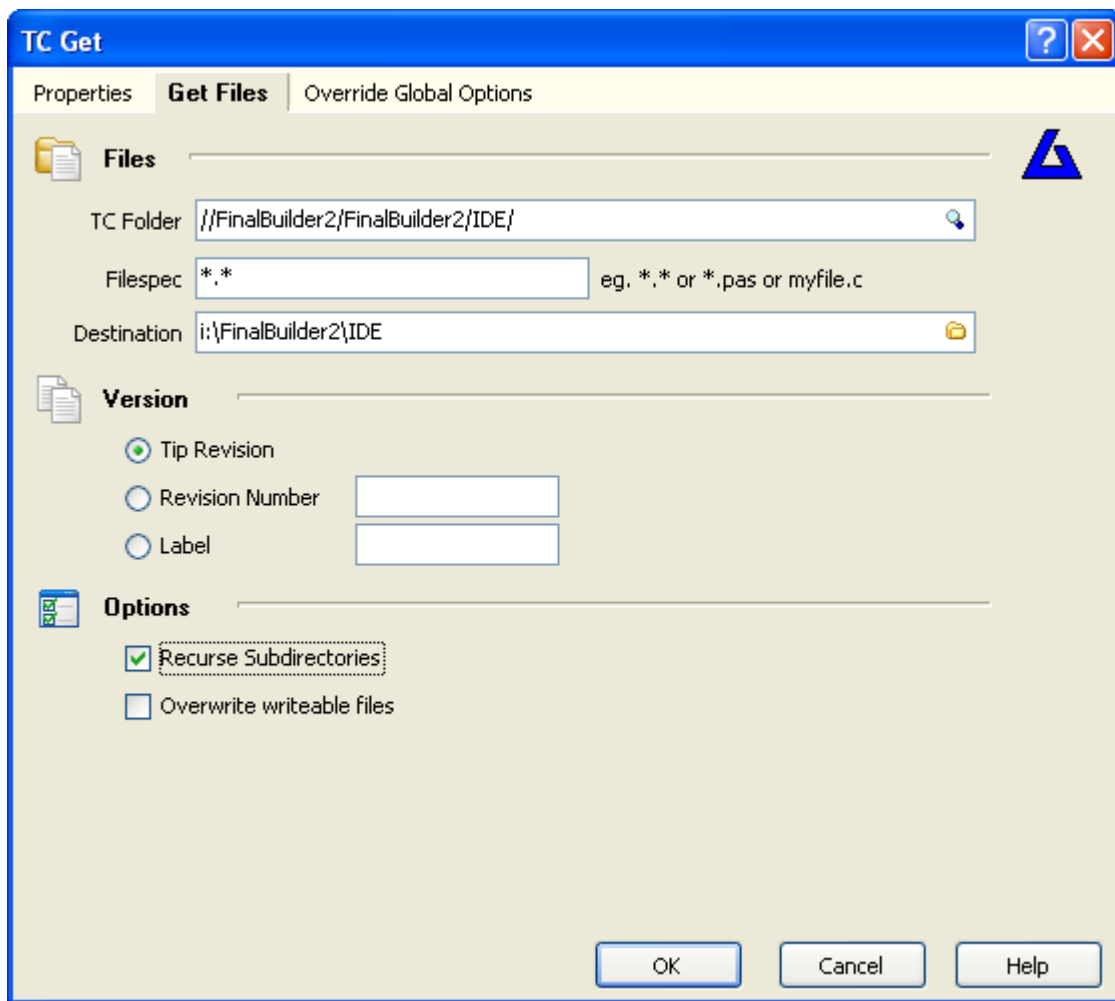
Selects a different view and makes it current.

**Remarks**

For more information on Views, see the main Team Coherence help file.

**5.2.7.3 Team Coherence Get Action**

Gets a read-only copy of the specified files.

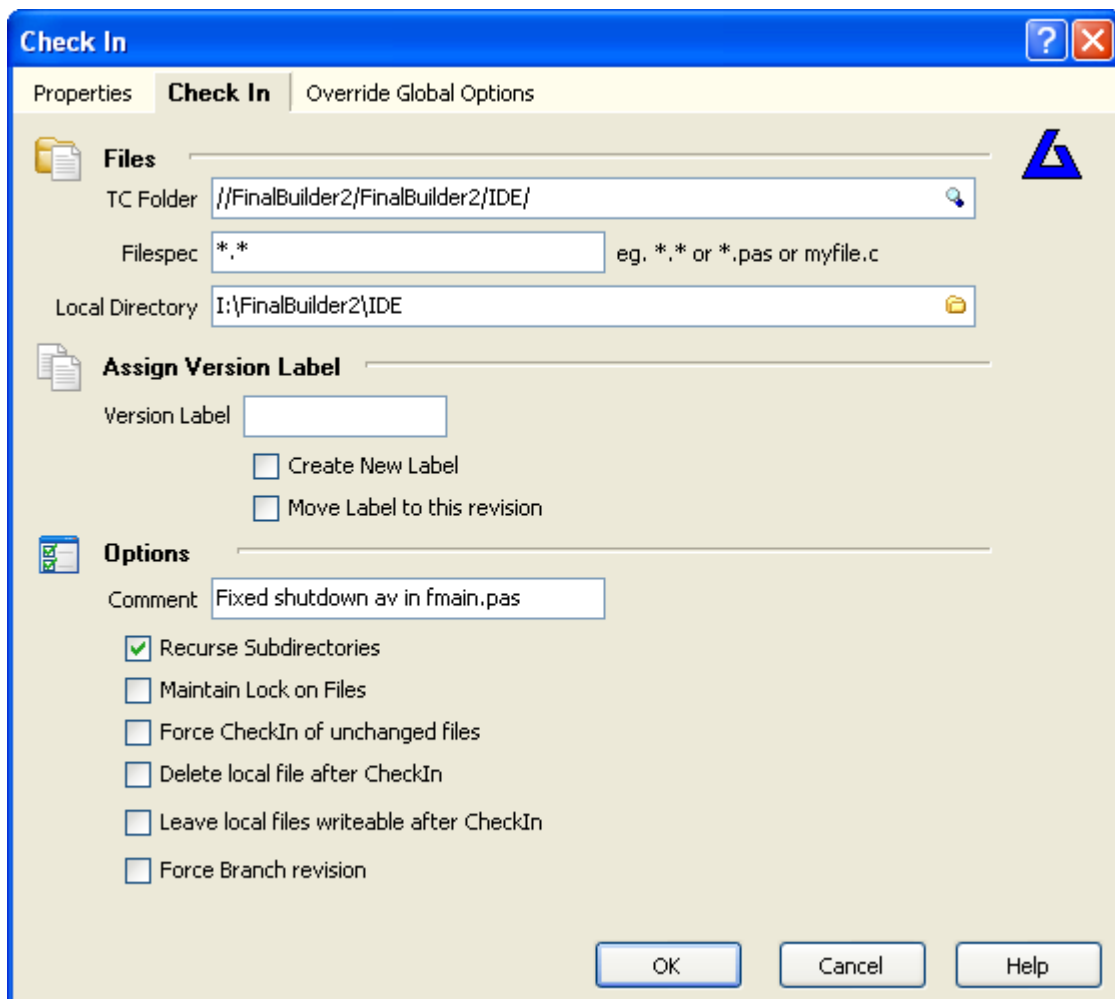


### Remarks

If the local copy of the file is not read-only the action will fail for that file. To override this default and to cause the local file to be overwritten with the new read-only copy, use the -W command-line option.

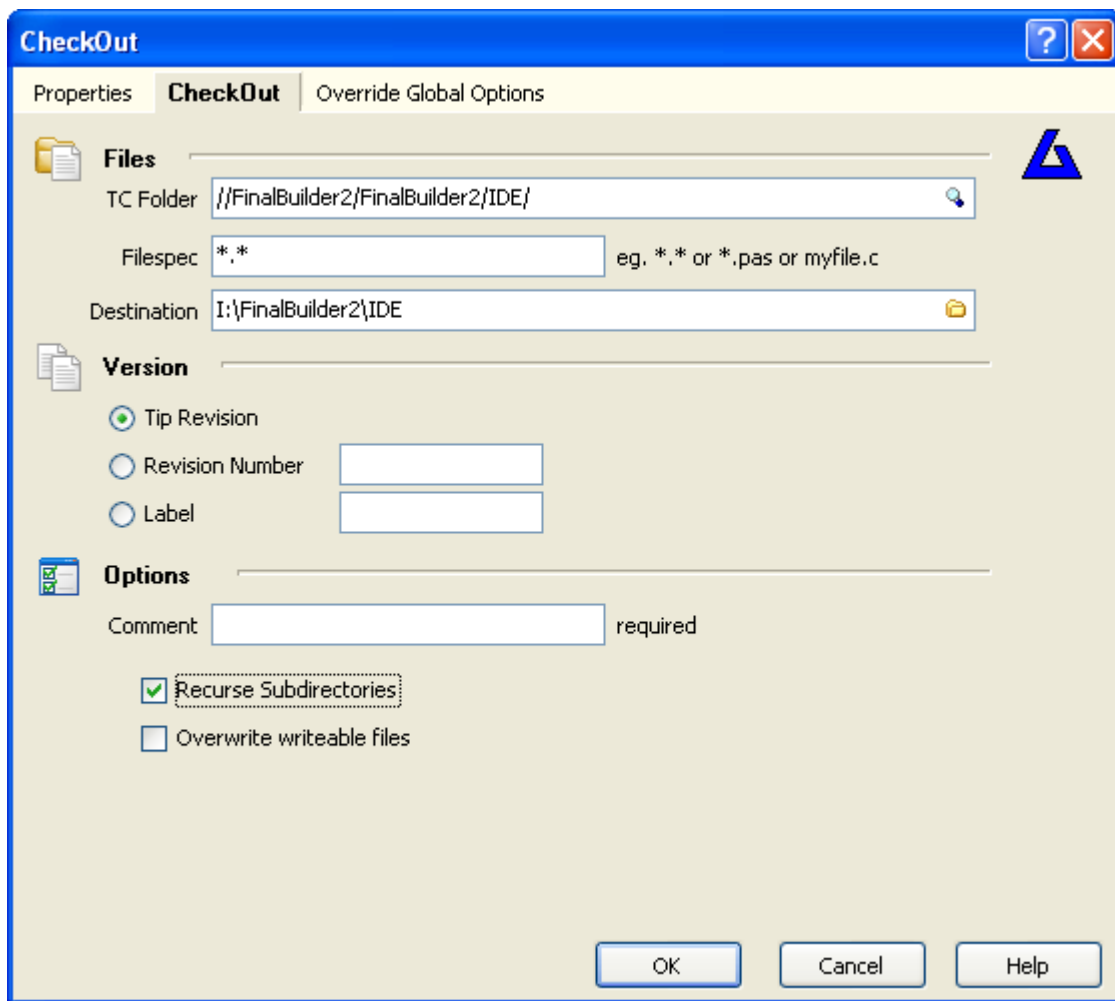
#### 5.2.7.4 Team Coherence Check In Action

Updates Team Coherence with changes made to a checked out file and unlocks the archive.



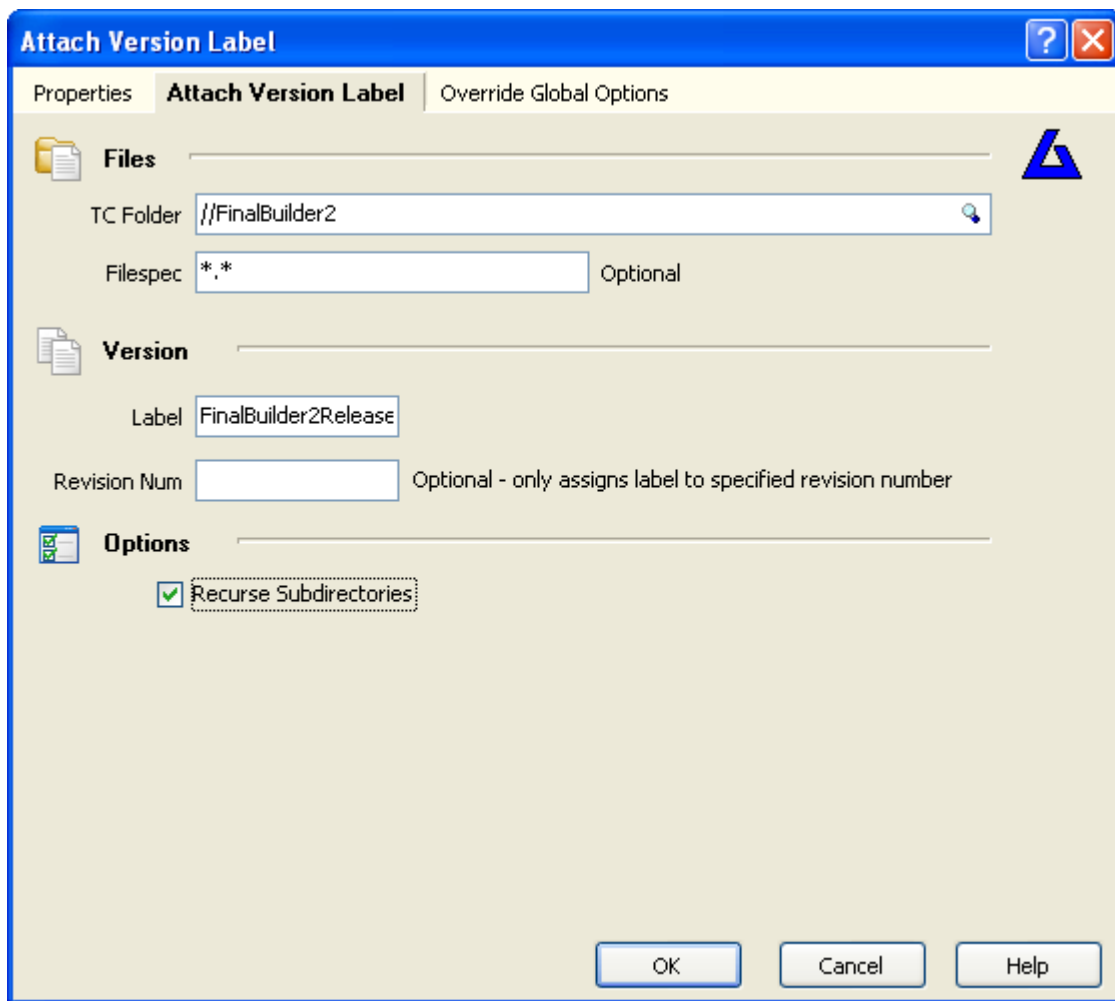
#### 5.2.7.5 Team Coherence Check Out Action

Extracts a file from the current folder to the current directory for the purposes of editing.



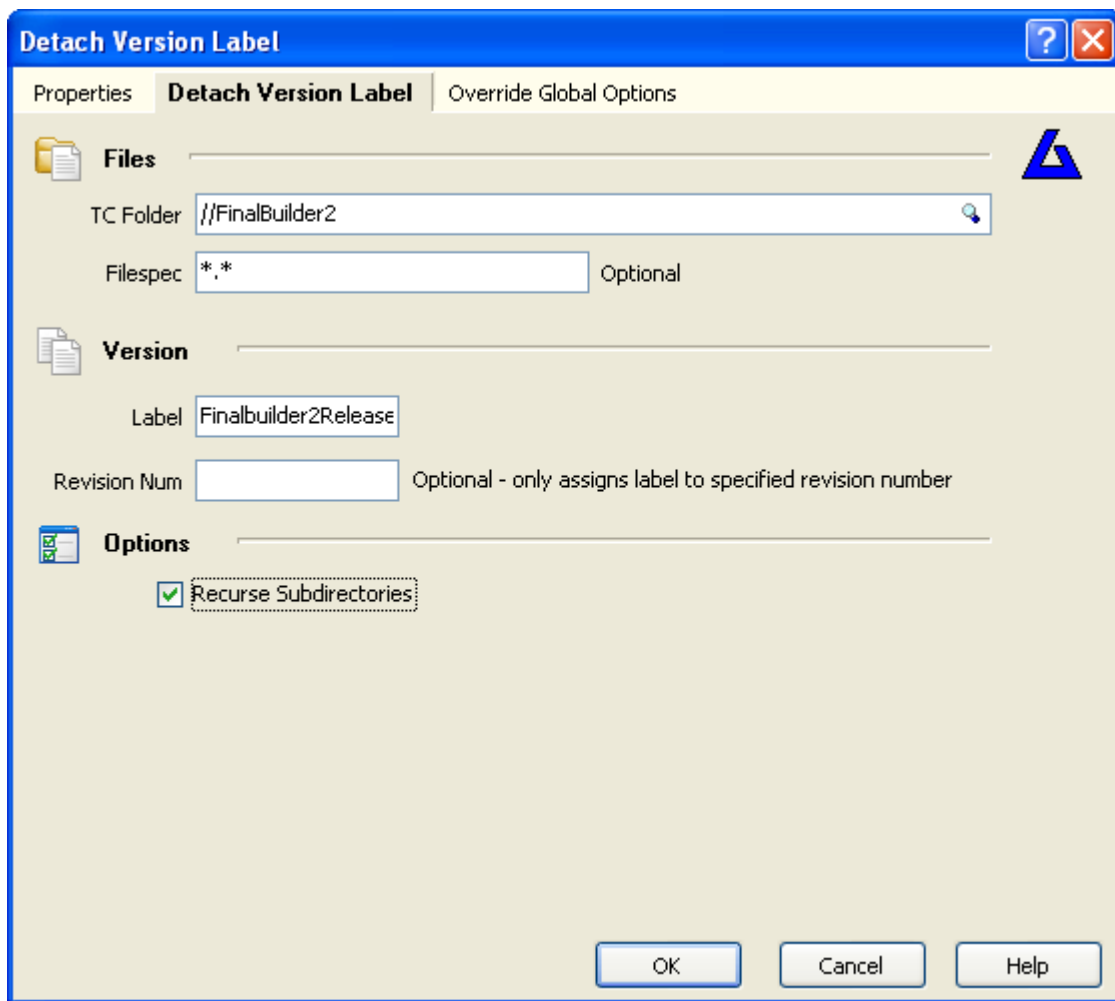
#### 5.2.7.6 Team Coherence Attach Label Action

Attaches a Version Label to the latest revision of a file or to a specified revision.



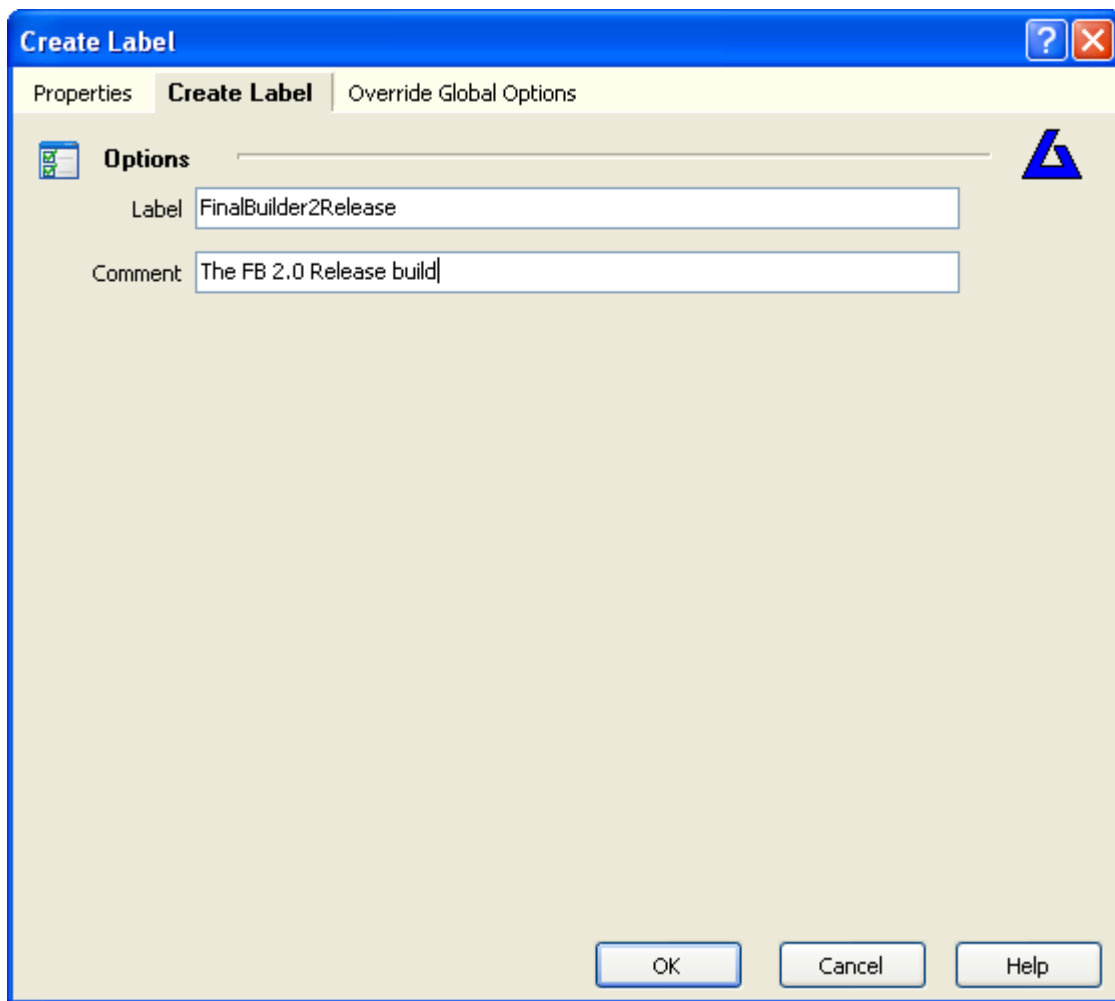
#### 5.2.7.7 Team Coherence Detach Label Action

Detaches the specified label from the specified objects.



#### 5.2.7.8 Team Coherence Create Label Action

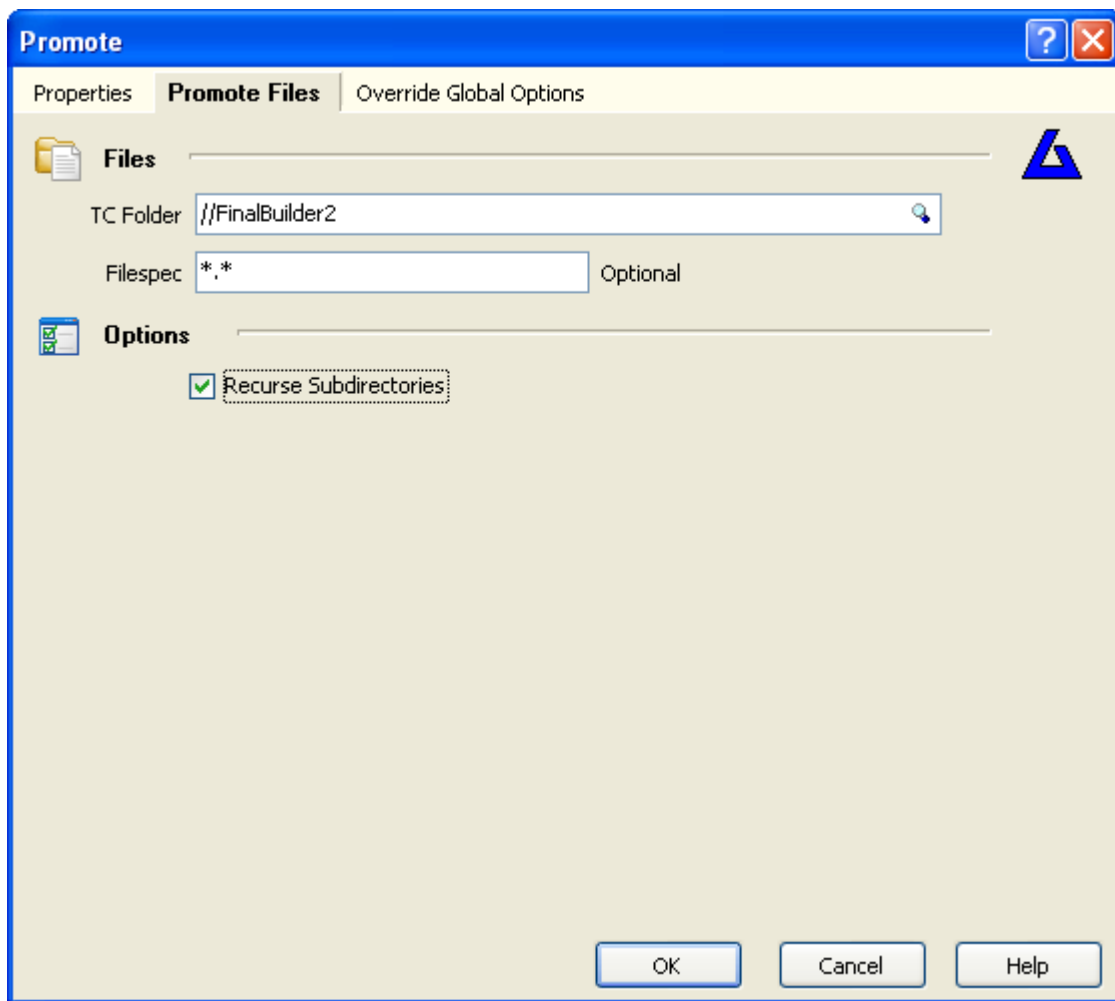
Creates a new Version Label.



#### 5.2.7.9 Team Coherence Promote Action

Promotes the specified files to the next Promotion Level. This action will promote the tip revision (as defined by the current View) to the next higher Promotion Level.



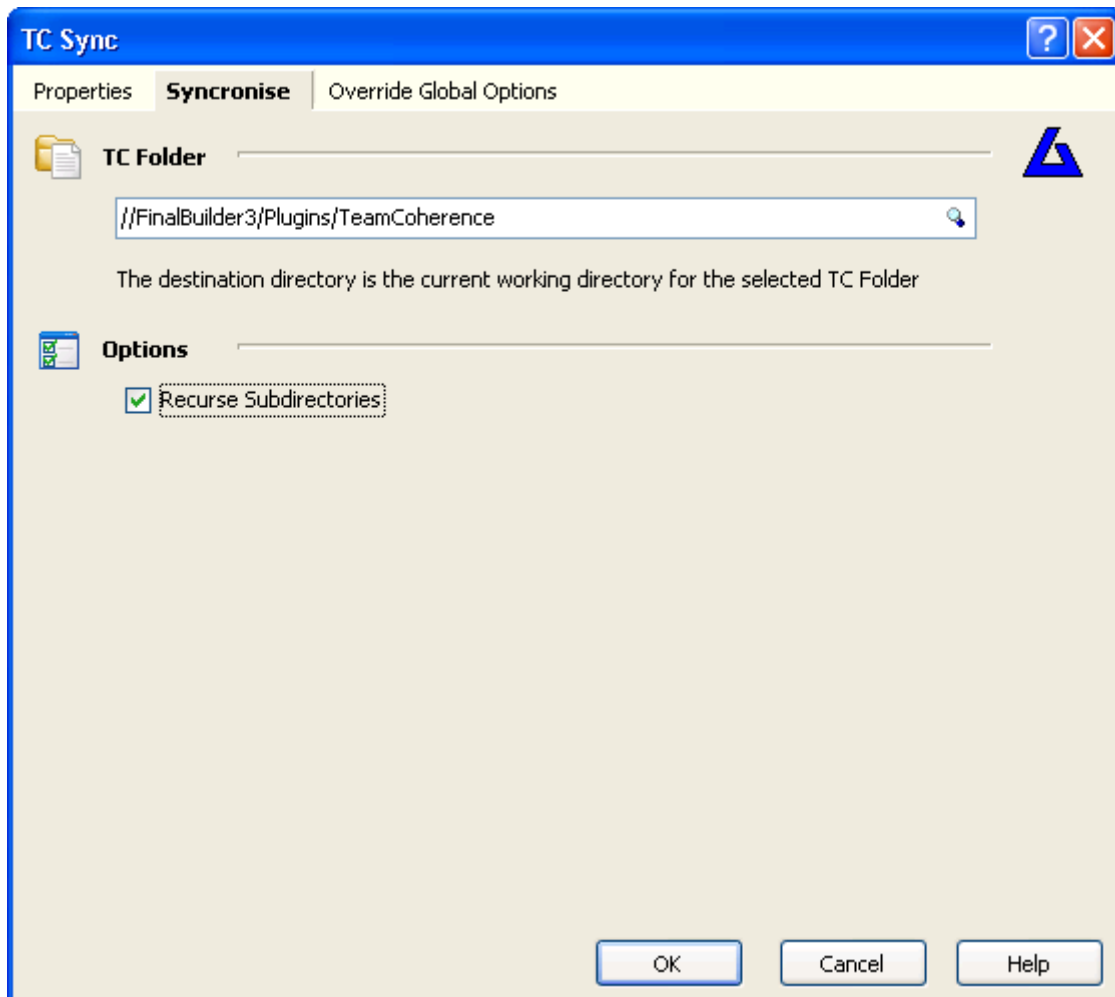


Both the Tip Revision and the next higher Promotion Level are defined by the currently selected View. If the current View is based on a Version Label, the revision that has the Version Label attached will be promoted to the first Promotion Level. If the current View is based on a Promotion Level, the revision that is currently at that level will be promoted to the next higher Promotion Level.

If the <default> View is current, or the current View is based on neither a Version Label or Promotion Level, the tip revision of each file will be promoted.

#### 5.2.7.10 Team Coherence Sync

The Sync command will synchronise your out of date or missing local files with the current files in the repository.



#### 5.2.7.11 Team Coherence Create View

Create a new View in the repository.

**TC Create View**

Properties **Create View** Override Global Options

**New View Details**

View Name: FB3.0.0

Description: FinalBuilder 3.0.0

Projects:   
//FinalBuilder3

If no projects are specified then view will apply to all projects

One project per line

**Options**

☒ Shared View

☐ Base view on version  
Version:

☐ Base view on promotion level  
Promotion Level:

OK Cancel Help

If you want the view to only apply to certain projects, then you must specify them in the Projects section.

**Shared View** - Shared Views are available to all users connected to a repository, however shared views can only be created by Admin users.

**Base view on version** - Specify the version which the view will be based on.

**Base view on promotion level** - Specify the predefined promotion level to base the view on. Promotional views are used to view all files at a particular level in the promotional hierarchy. Users can get a read-only copy of all the files at a particular promotional level but cannot modify them.

#### **More about TC Views, from the TC help file:**

A View is perhaps the most powerful tool available in Team Coherence when it comes to managing multiple versions of a project. At their simplest, Views allow you to filter which projects, files, and revisions are displayed in Version Manager.

Working with previous versions of a project then simply becomes a case of selecting a View. Once defined and selected, as far as the user is concerned they will be working on

the latest version of the project. All branching, synchronizing, reassignment of Version Labels, etc are handled automatically by Team Coherence whether working through Version Manager, or your selected IDE interface.

In addition, if required, the default working path for the files in any View can be different for each View, or can be based on the <default> View.

#### **5.2.7.12 Team Coherence Update View**

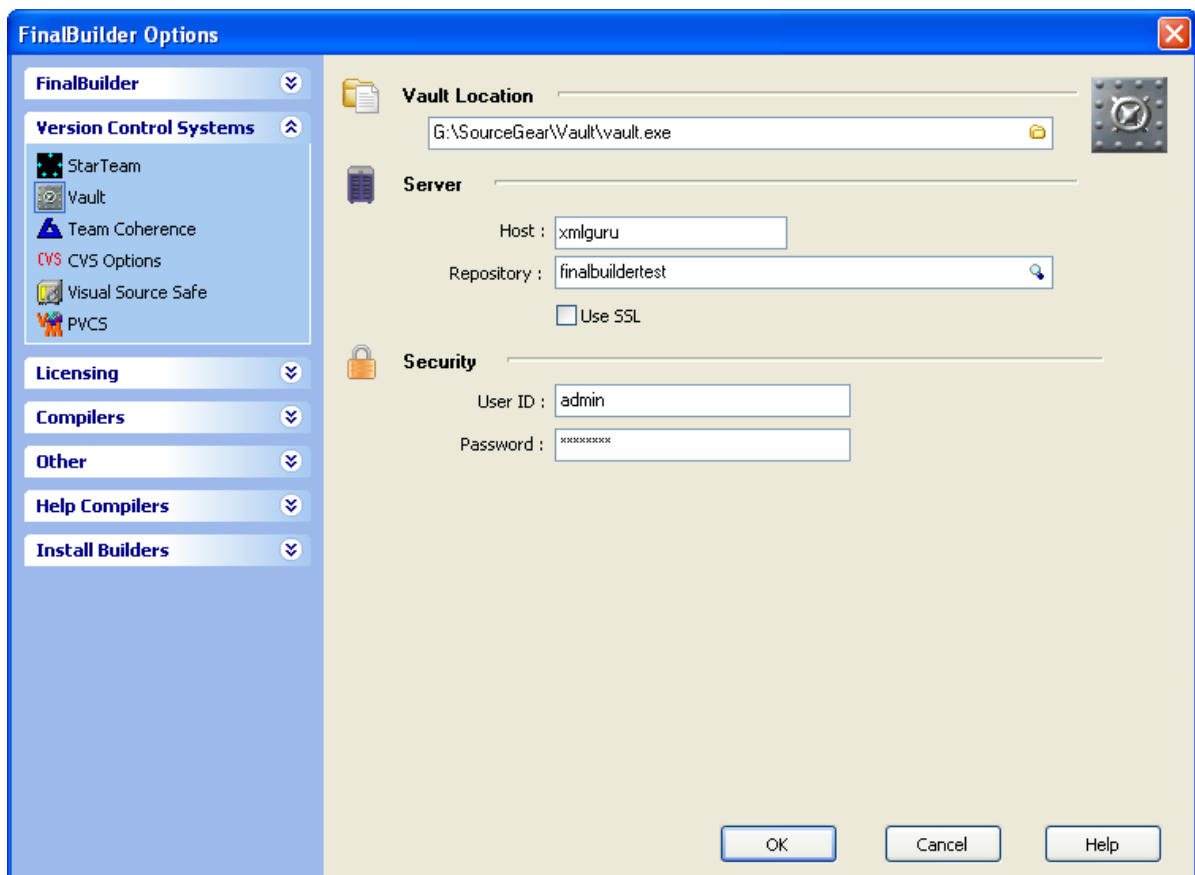
Update View enables you to change the definition of an existing view. See Team Coherence Create View for more information on the available properties.

#### **5.2.7.13 Team Coherence Delete View**

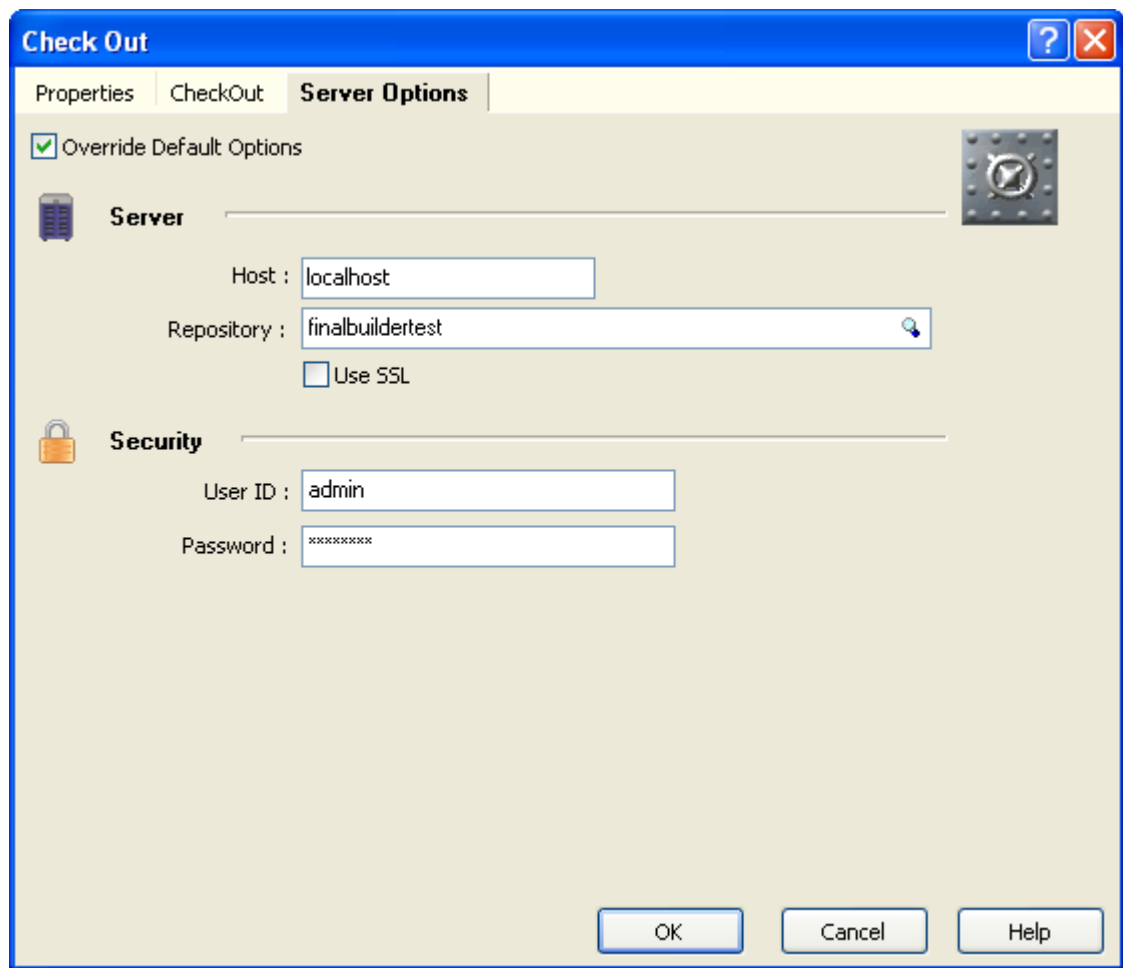
Delete a view in the TC repository. See Team Coherence Create View for more information on views.

### **5.2.8 SourceGear Vault Actions**

These Actions support SourceGear Vault 1.2 or higher. They provide an easy to use wrapper around the Vault command line tool. Before using these actions you need to check that the path to the Vault command line tool is set in the FinalBuilder Options Dialog. FinalBuilder will attempt to find the path automatically however it is not always possible (depends on your installation).

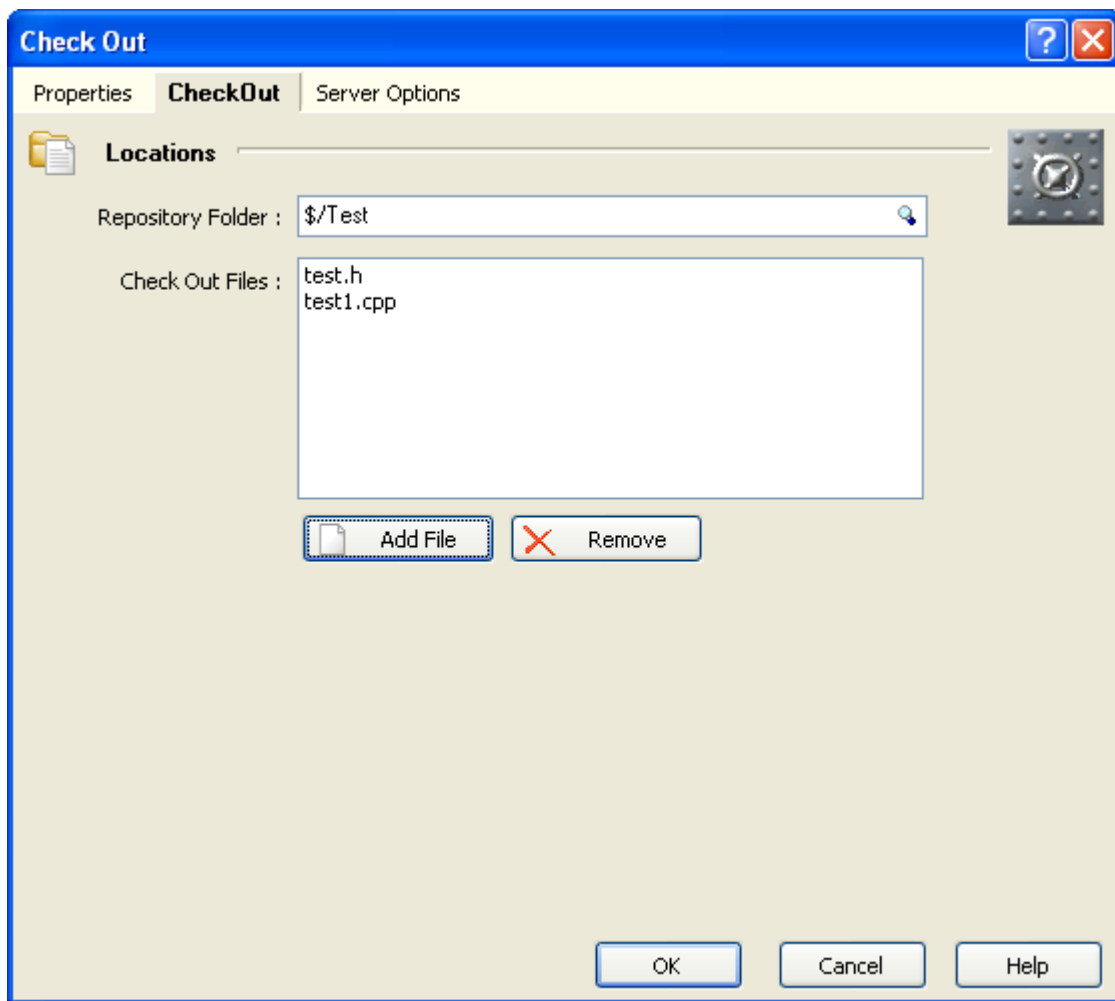


The default server and security options can be overridden in the individual actions.



#### 5.2.8.1 Vault Check Out Action

CHECKOUT will checkout files from the repository.

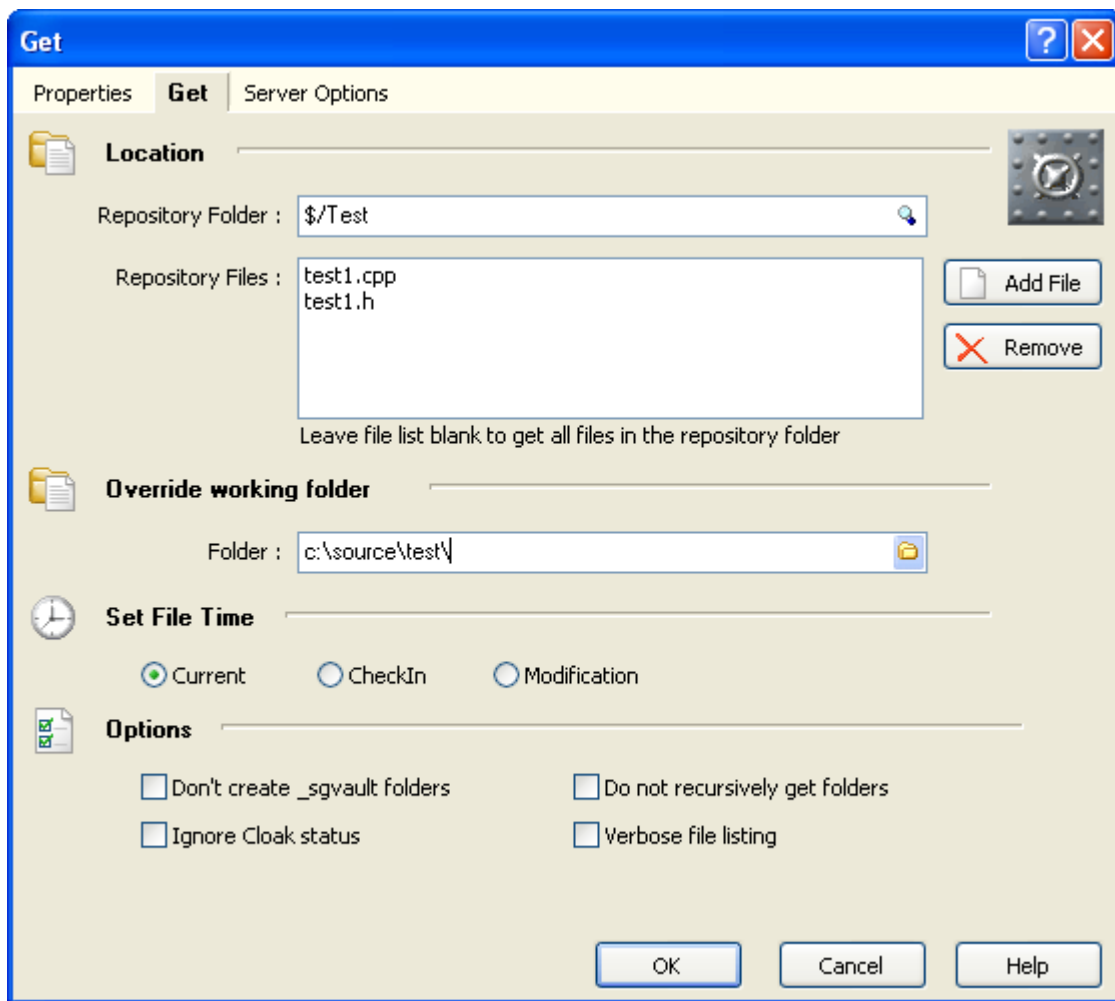


Server and authentication information is specified by:

- host host  
Hostname of the vault server to connect to - also see SERVER
- ssl  
Enables SSL for server connection
- user username  
Username to use when connecting to server - also see USERNAME
- password password  
Password to use when connecting to server
- repository repositoryname  
Repository to connect to

#### 5.2.8.2 Vault Get Action

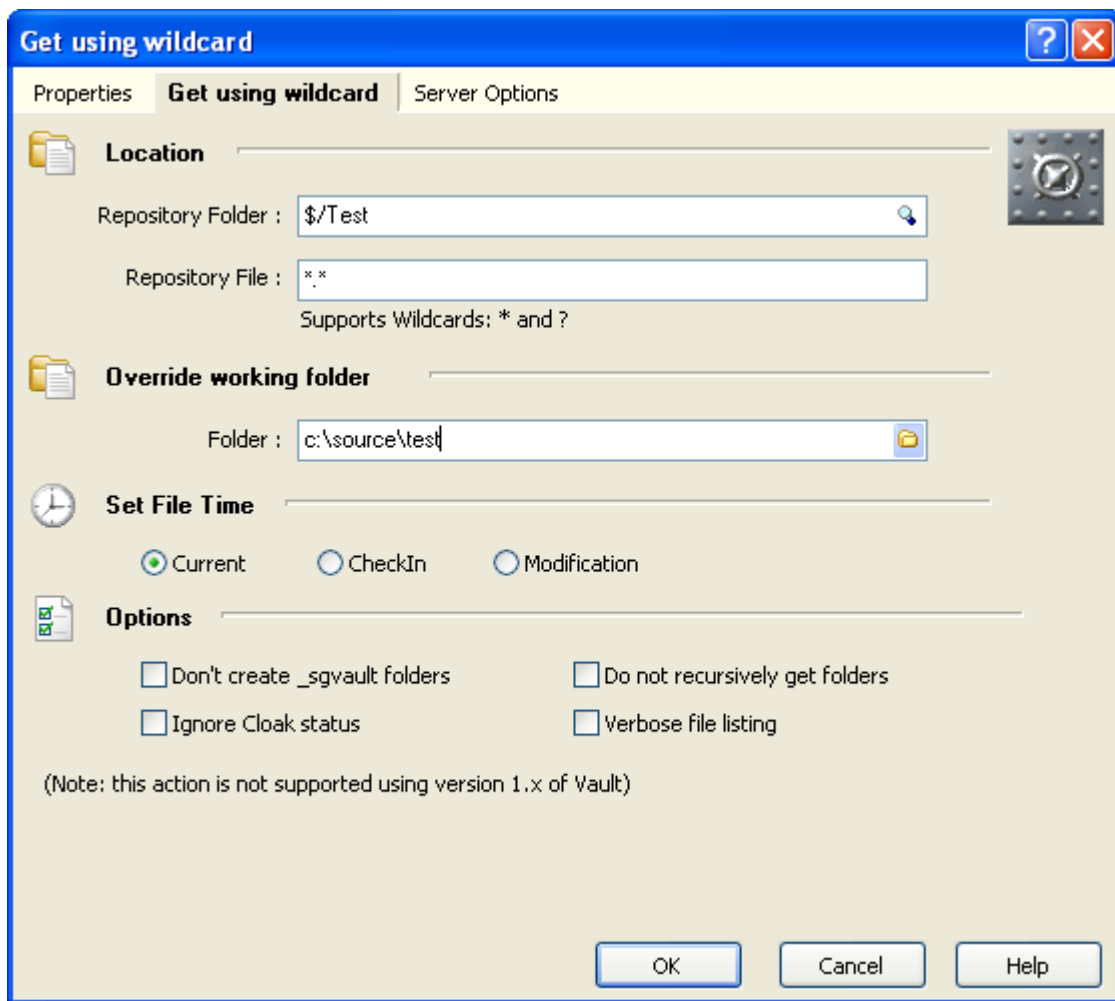
GET will retrieve the latest version of files or folders in the repository as specified by repositorypath(s). The files will be stored in their corresponding working folders on the local system.



### 5.2.8.3 Vault Get using Wildcards Action

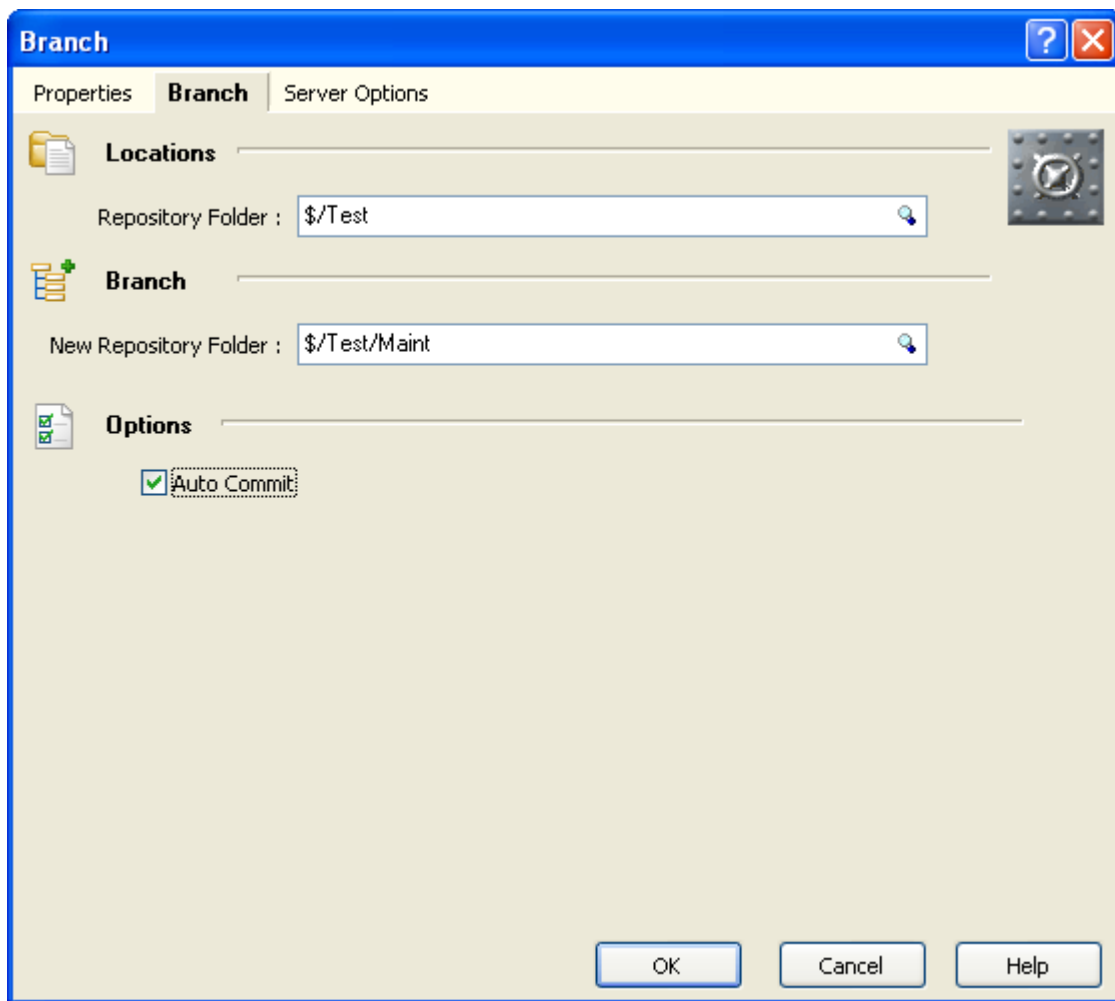
GETWILDCARD will retrieve all files within the folder specified by repositoryfolder whose name matches one of the wildcards specified. You may use '?' to match a single character or '\*' to match a range of characters.





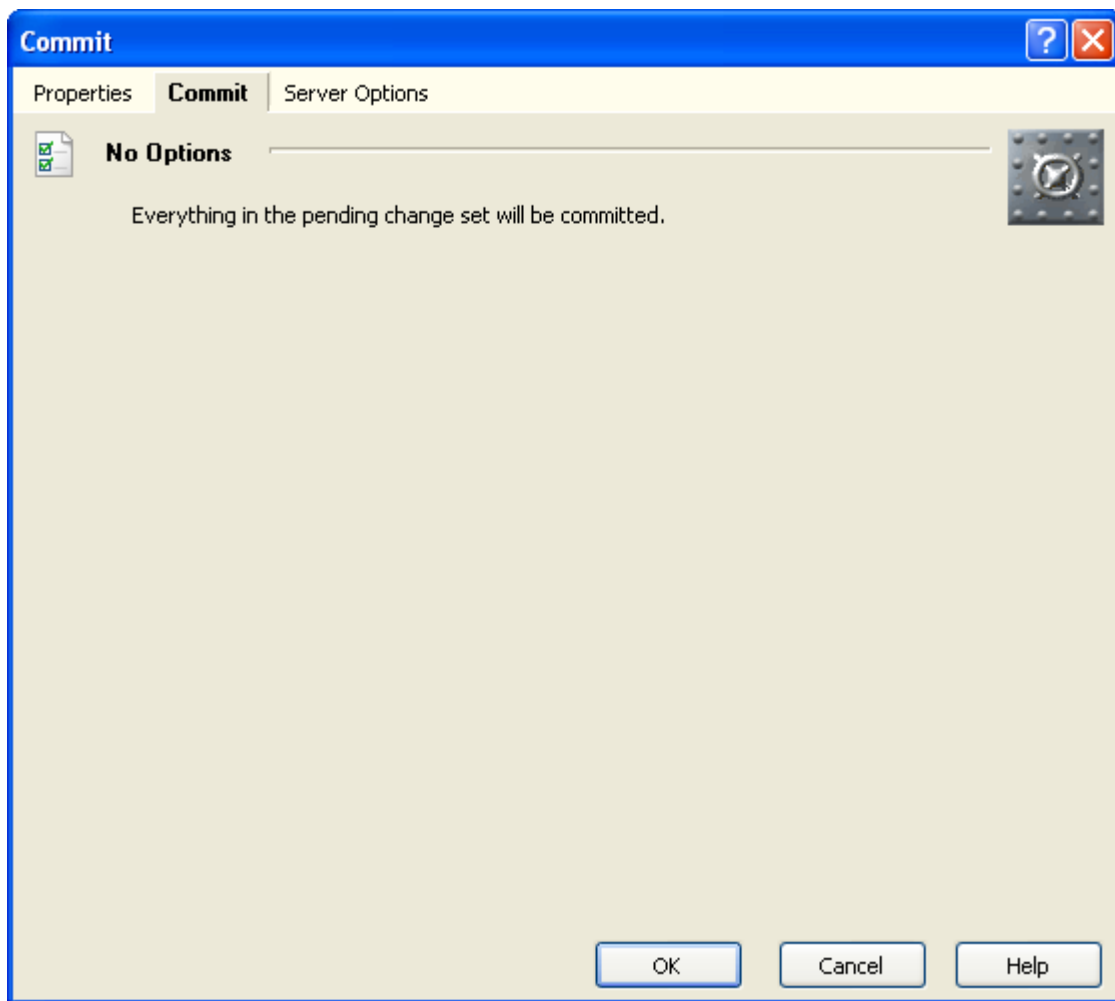
#### 5.2.8.4 Vault Branch Action

BRANCH will create a branch for the repository folder specified by repository folder at New repository folder.



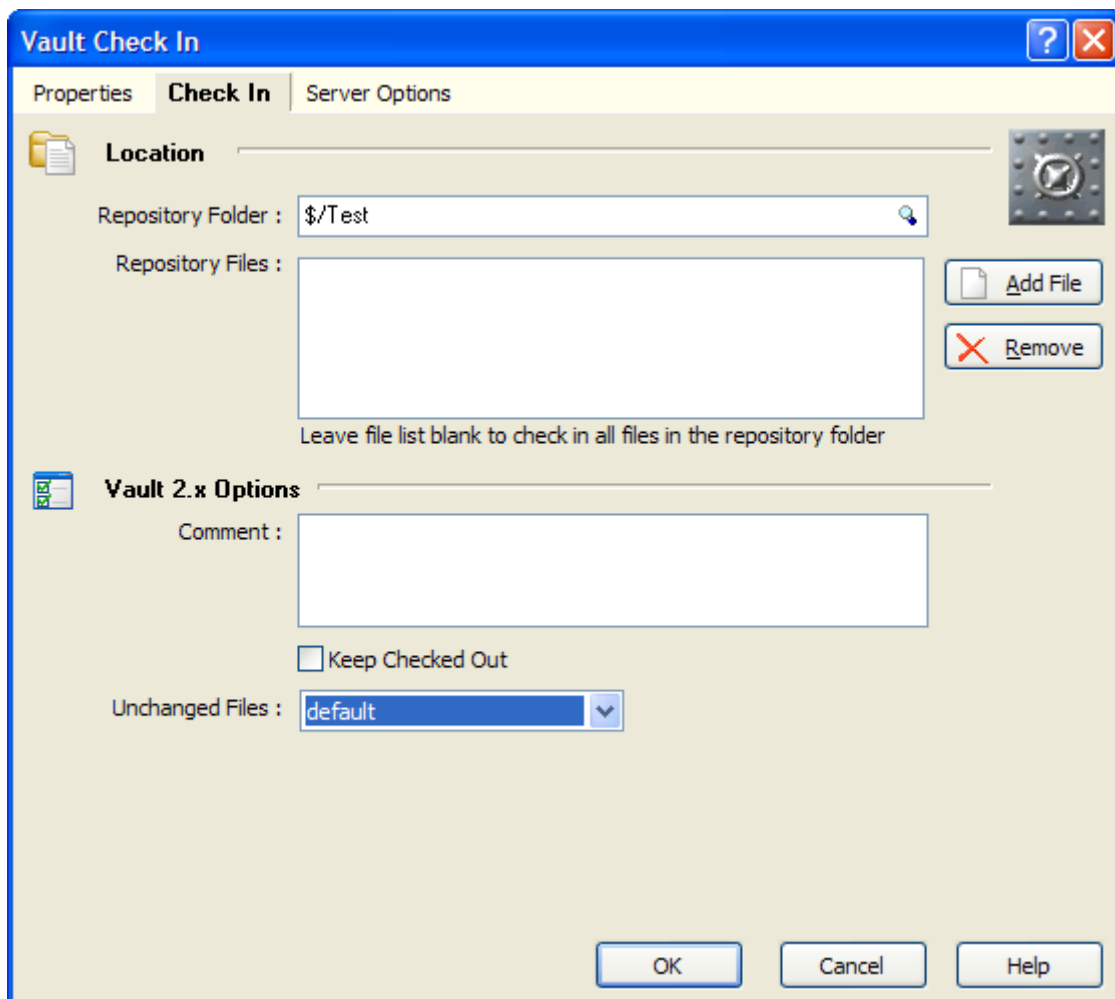
#### 5.2.8.5 Vault Commit Action

Commits all the items in the pending changeset list.



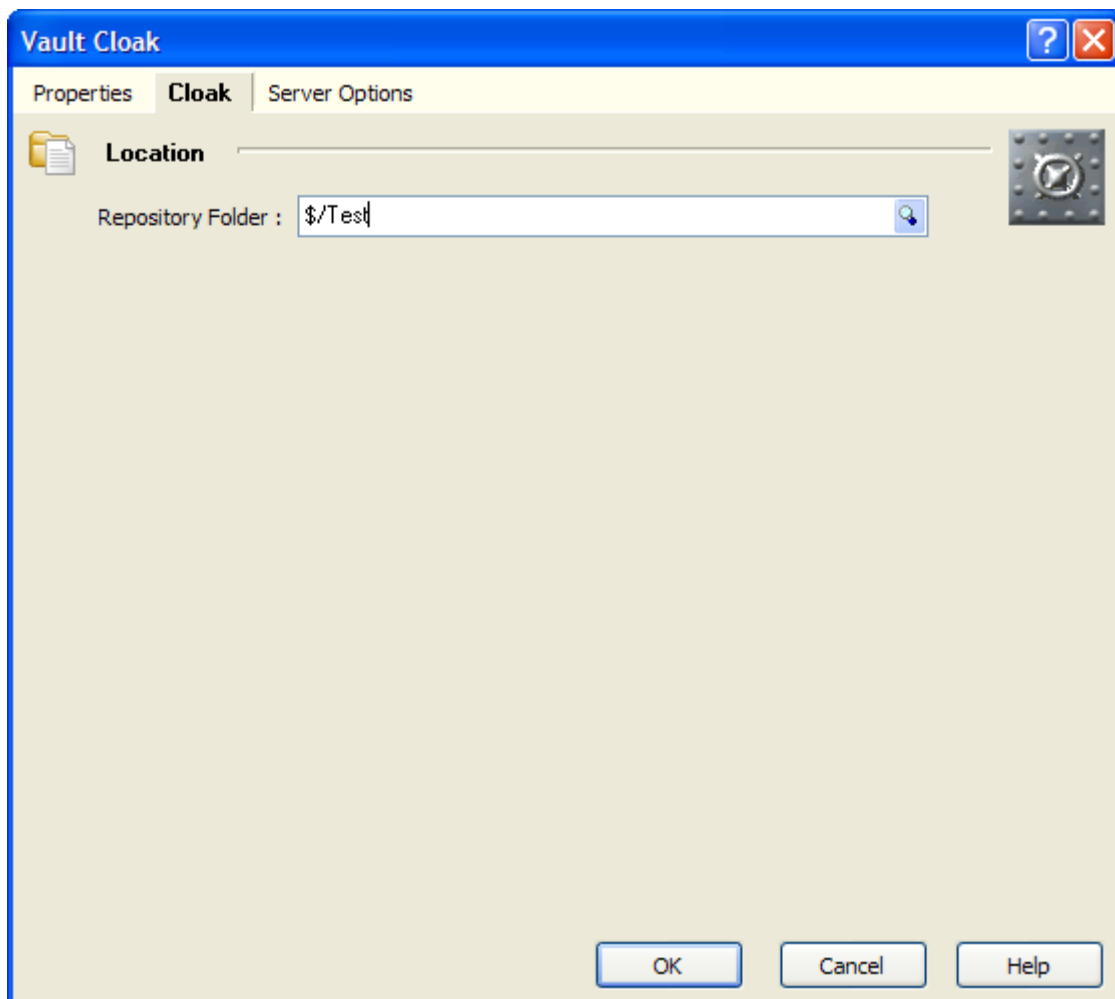
#### 5.2.8.6 Vault Check In Action

CheckIn uses the COMMIT command, it will commit the items in the pending changeset list specified by files(s).



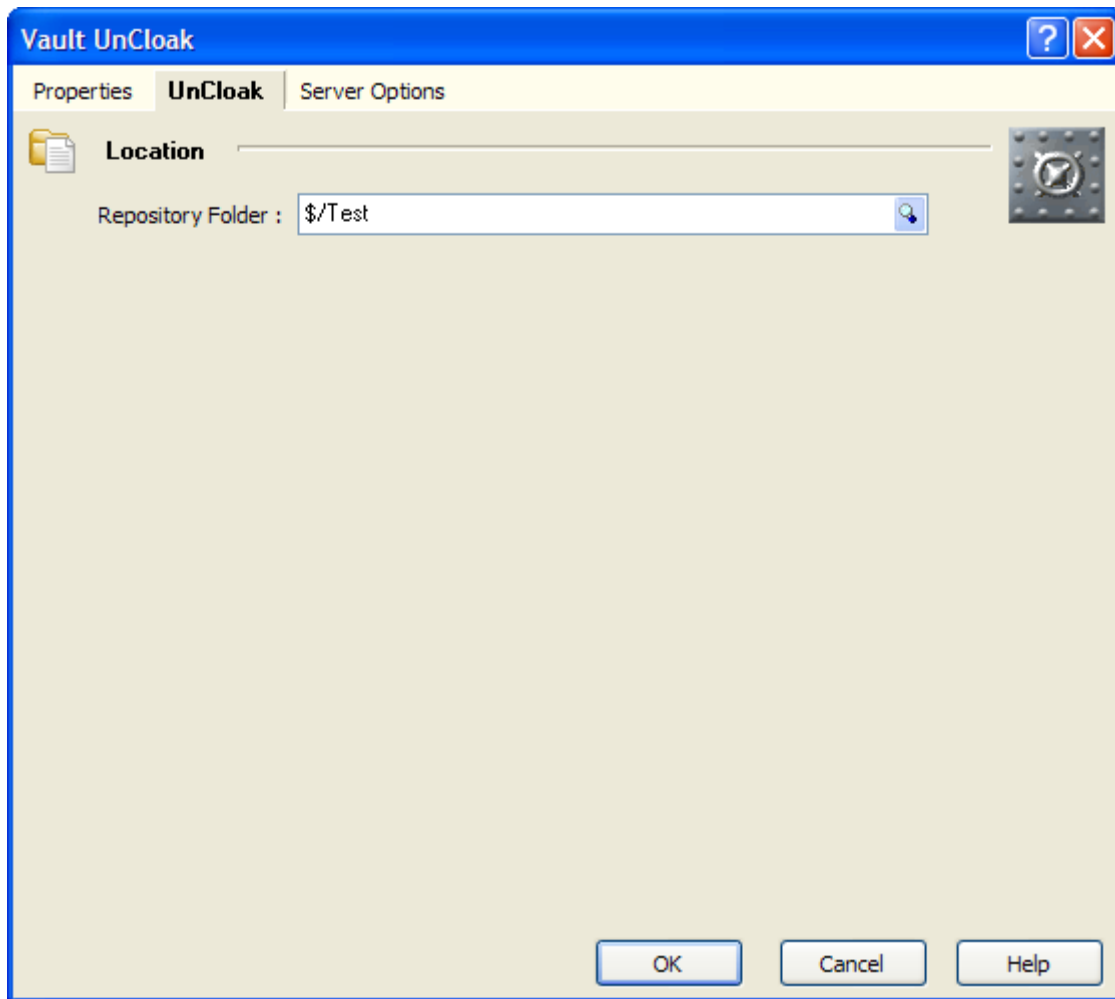
#### 5.2.8.7 Vault Cloak Action

The Cloak action will cloak a repository folder.



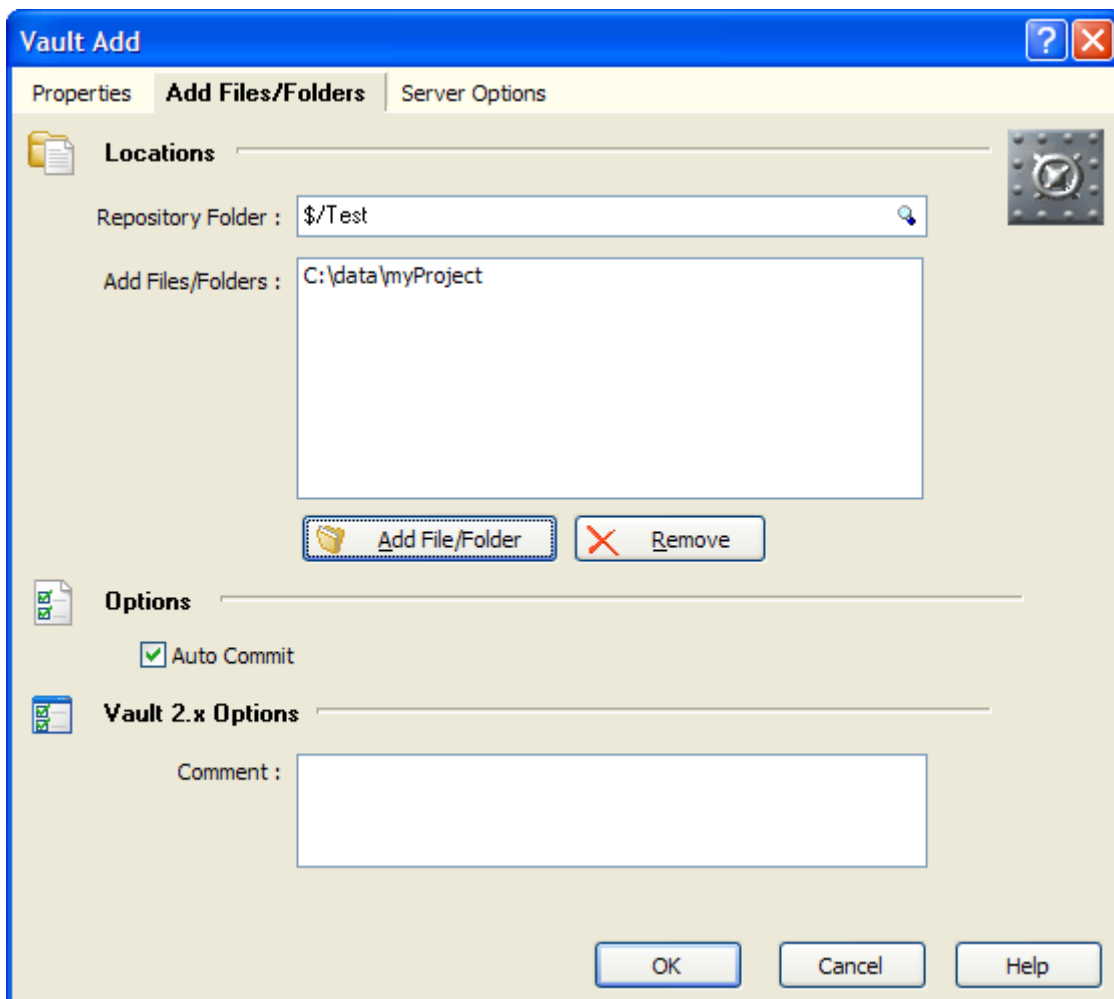
#### 5.2.8.8 Vault UnCloak Action

The UnCloak action will uncloak a repository folder.



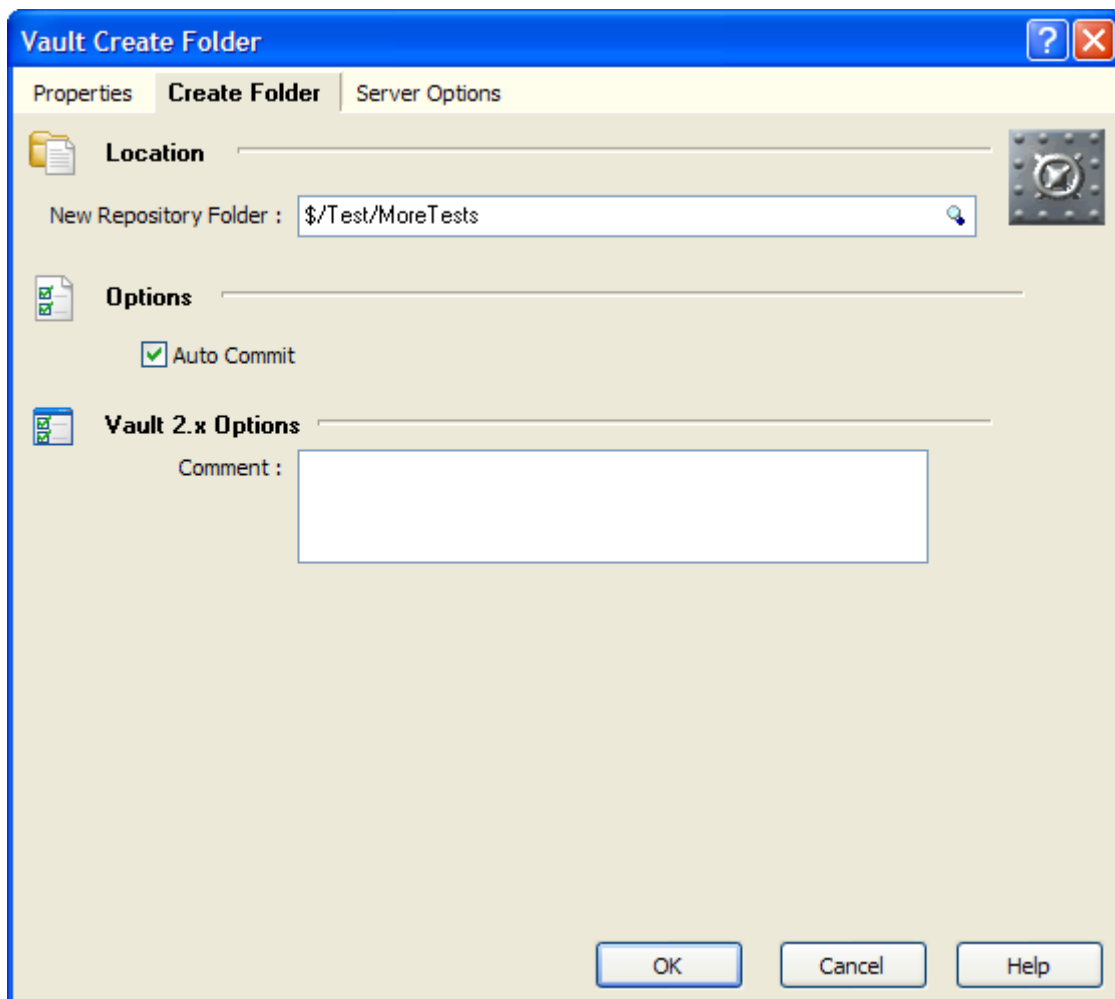
#### 5.2.8.9 Vault Add Action

The Vault Add action will add the specified files and directories into the selected repository folder.



#### 5.2.8.10 Vault Create Folder Action

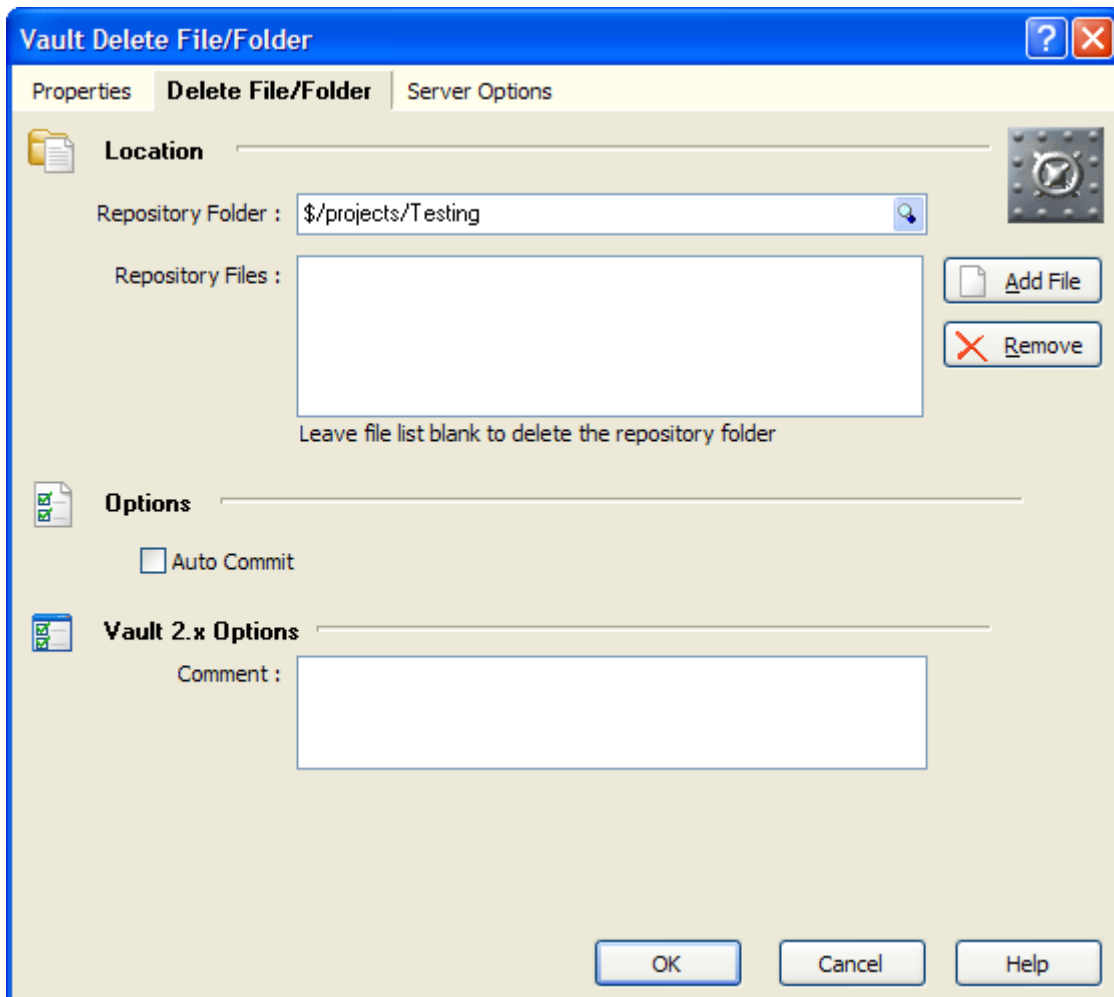
The Vault Create Folder action will create a new vault folder at the specified location in the repository



#### 5.2.8.11 Vault Delete File/Folder Action

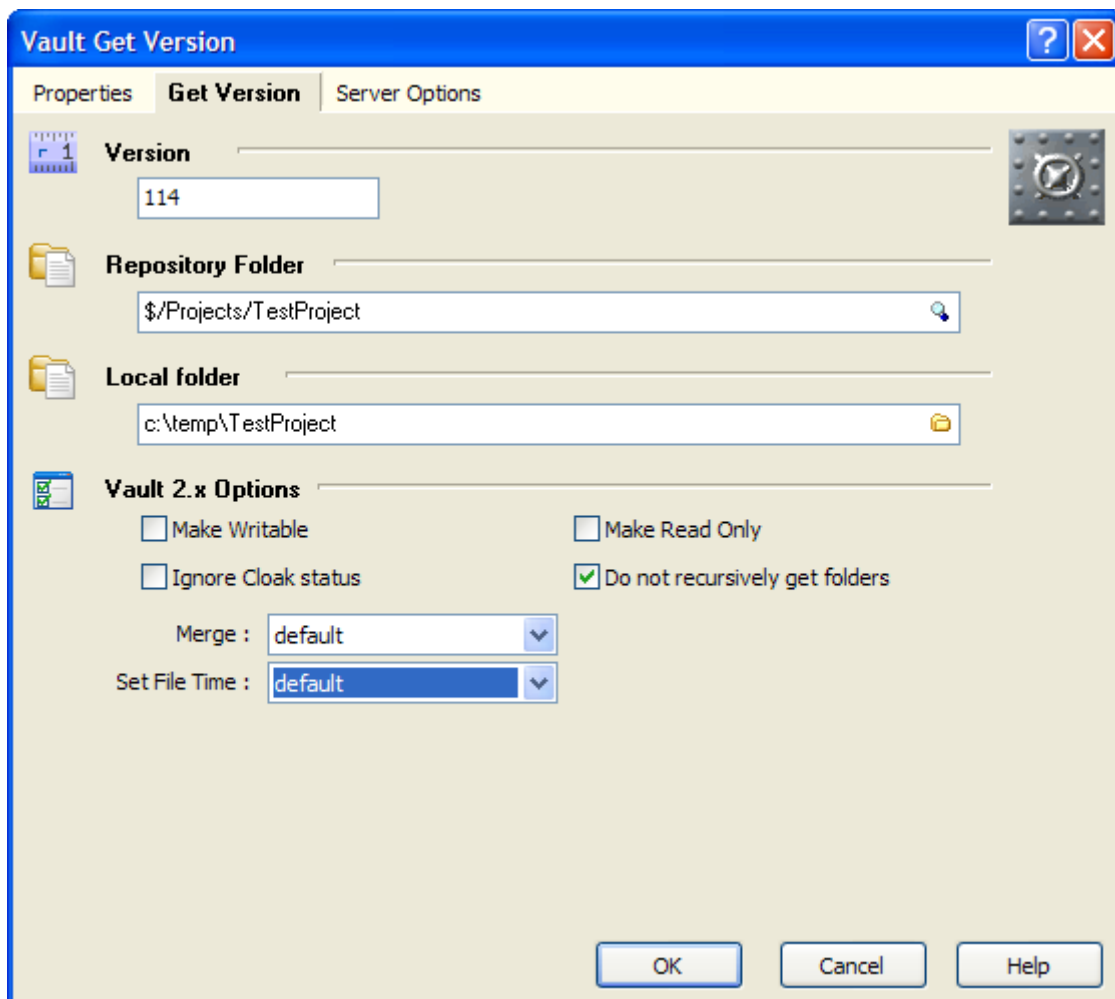
The Vault delete file/folder action will delete the specified files in the repository folder, or it will delete a repository folder if no files have been specified.





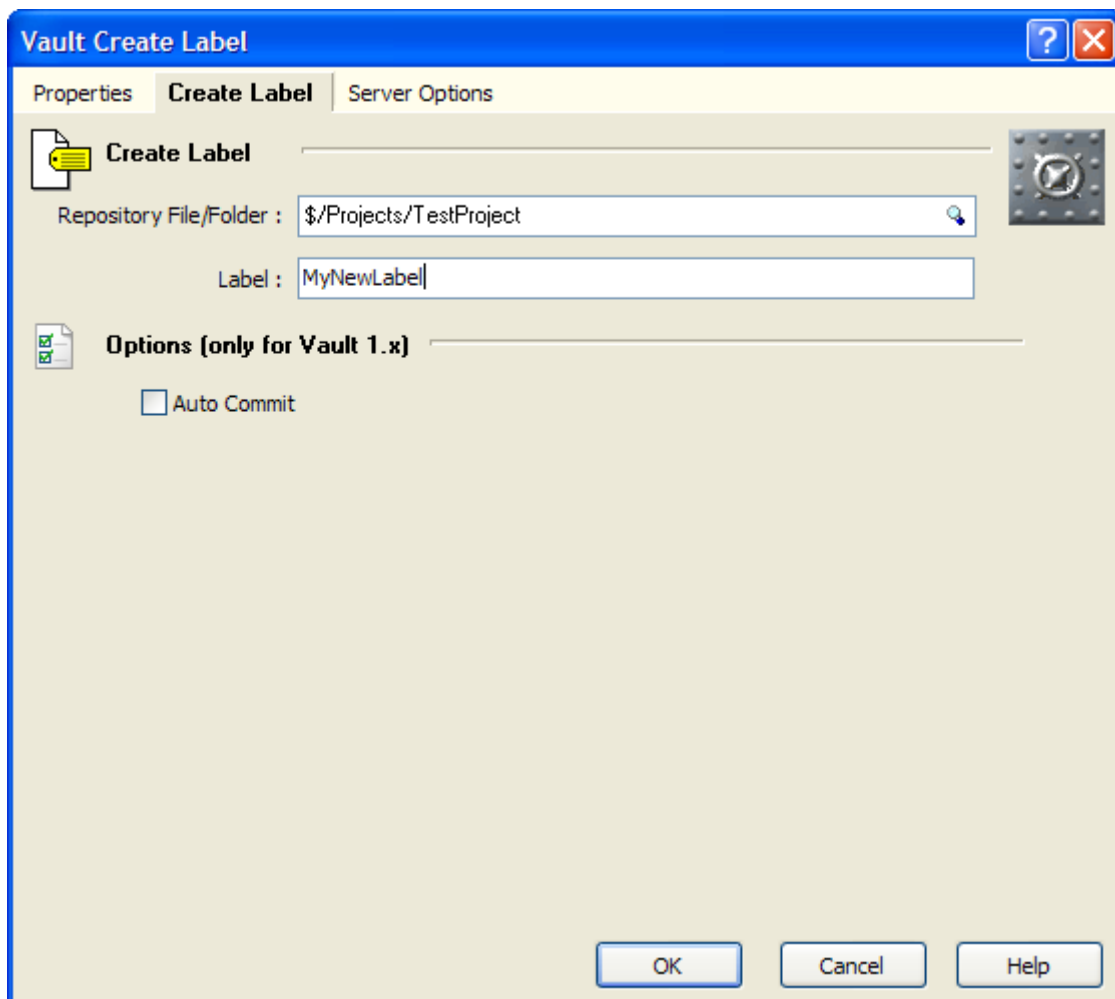
#### 5.2.8.12 Vault Get Version Action

The Vault Get Version action will get files from the vault repository to the specified local folder at a specific version number.



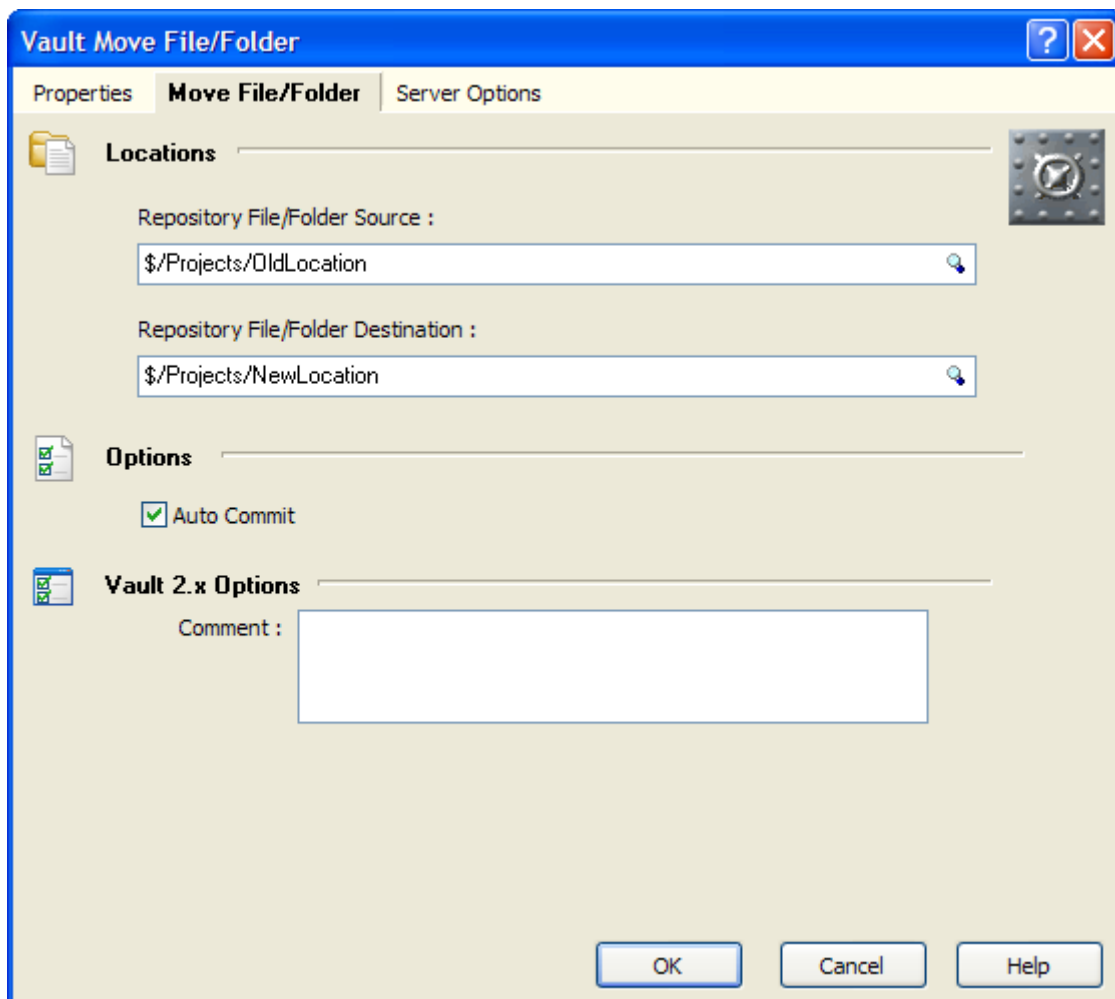
#### 5.2.8.13 Vault Create Label Action

The Vault Create Label action will create a label at the specified repository file/folder.



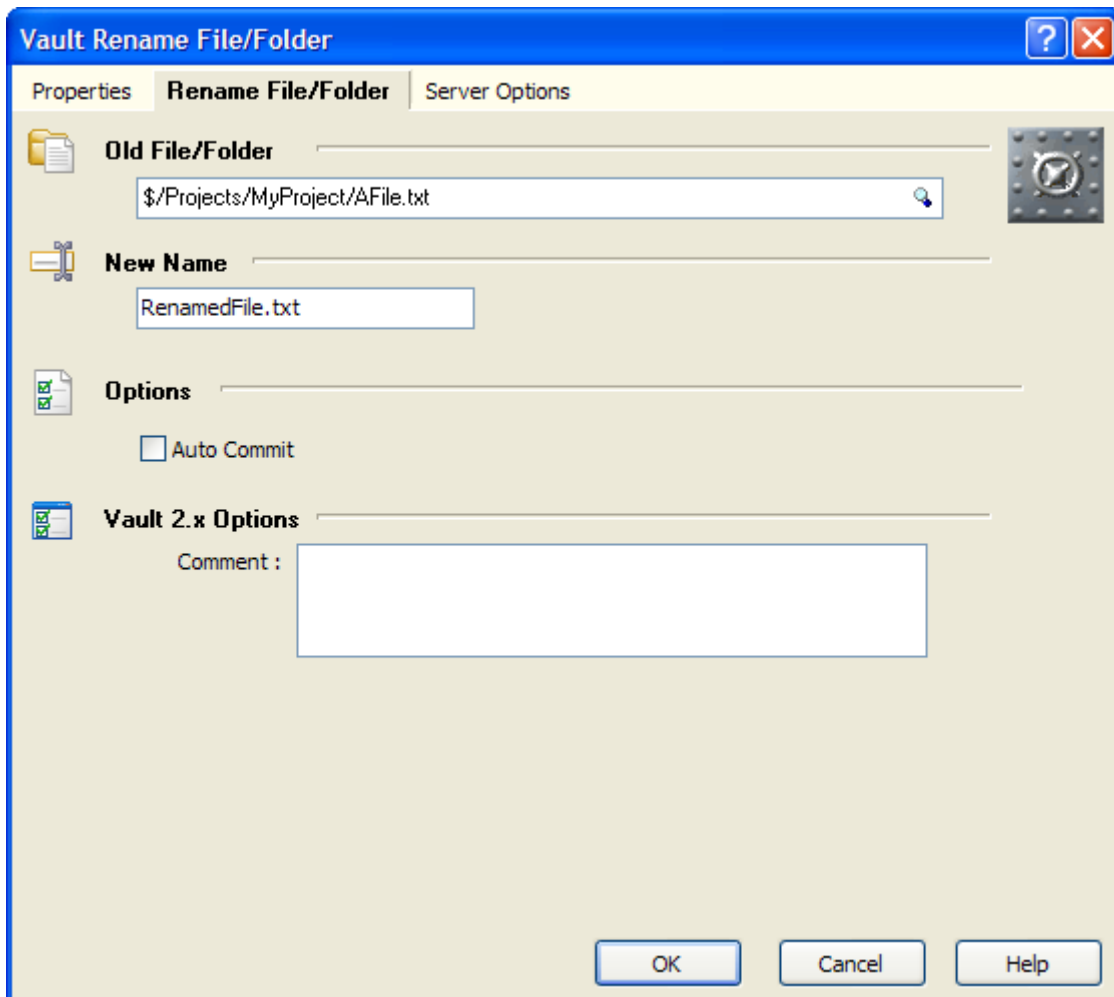
#### 5.2.8.14 Vault Move File/Folder Action

The Vault Move File/Folder action allows you move a file or folder to a new repository path.



#### 5.2.8.15 Vault Rename File/Folder Action

The vault rename file/folder action allows you to rename a file or folder in the vault repository.



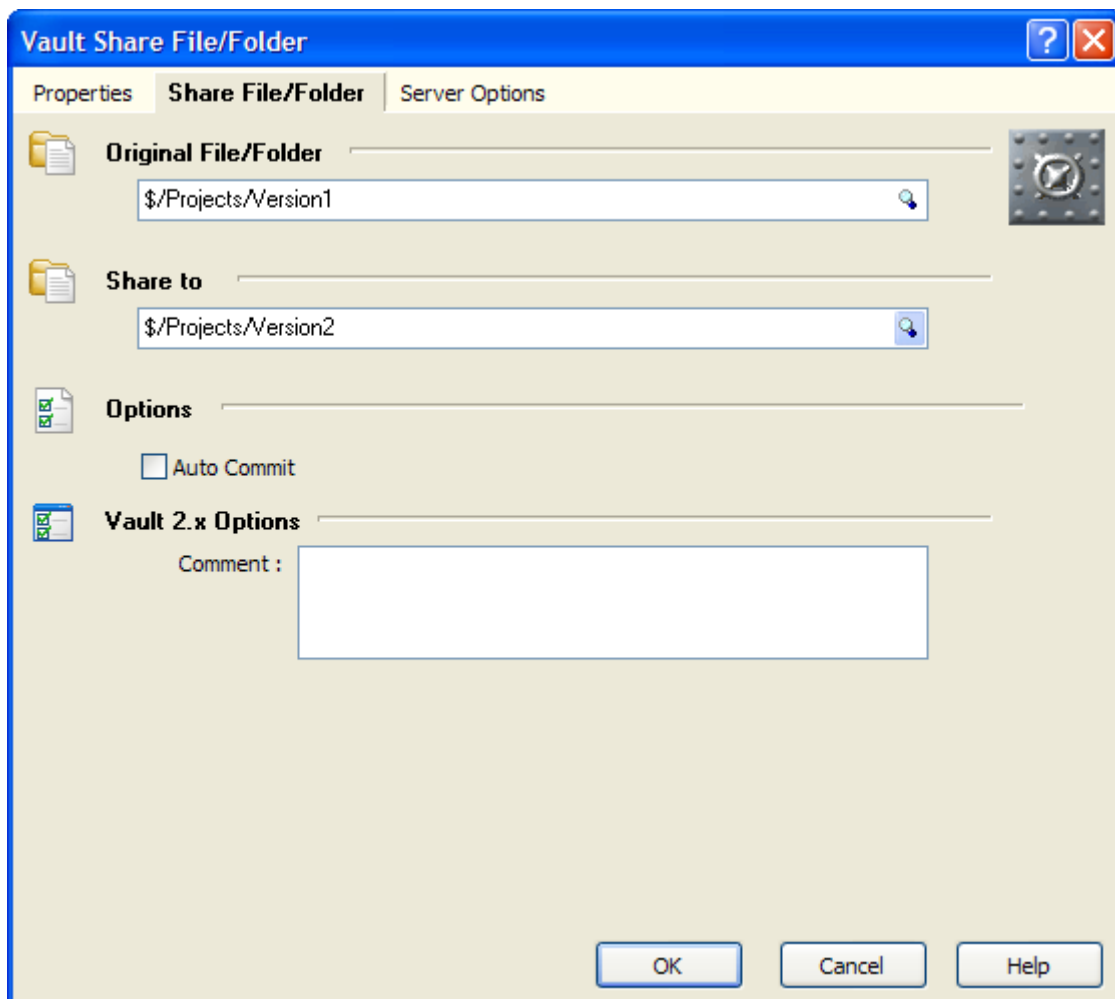
The screenshot shows a Windows-style dialog box titled "Vault Rename File/Folder". It has three tabs: "Properties", "Rename File/Folder" (which is selected), and "Server Options". The dialog is divided into several sections, each with an icon on the left:

- Old File/Folder**: Represented by a folder icon. It contains a text field with the path "\$/Projects/MyProject/AFile.txt" and a search icon on the right.
- New Name**: Represented by a document icon. It contains a text field with the name "RenamedFile.txt".
- Options**: Represented by a document icon with a checkmark. It contains a checkbox labeled "Auto Commit", which is currently unchecked.
- Vault 2.x Options**: Represented by a document icon with a checkmark. It contains a label "Comment :" followed by a large, empty text area.

At the bottom right of the dialog, there are three buttons: "OK", "Cancel", and "Help".

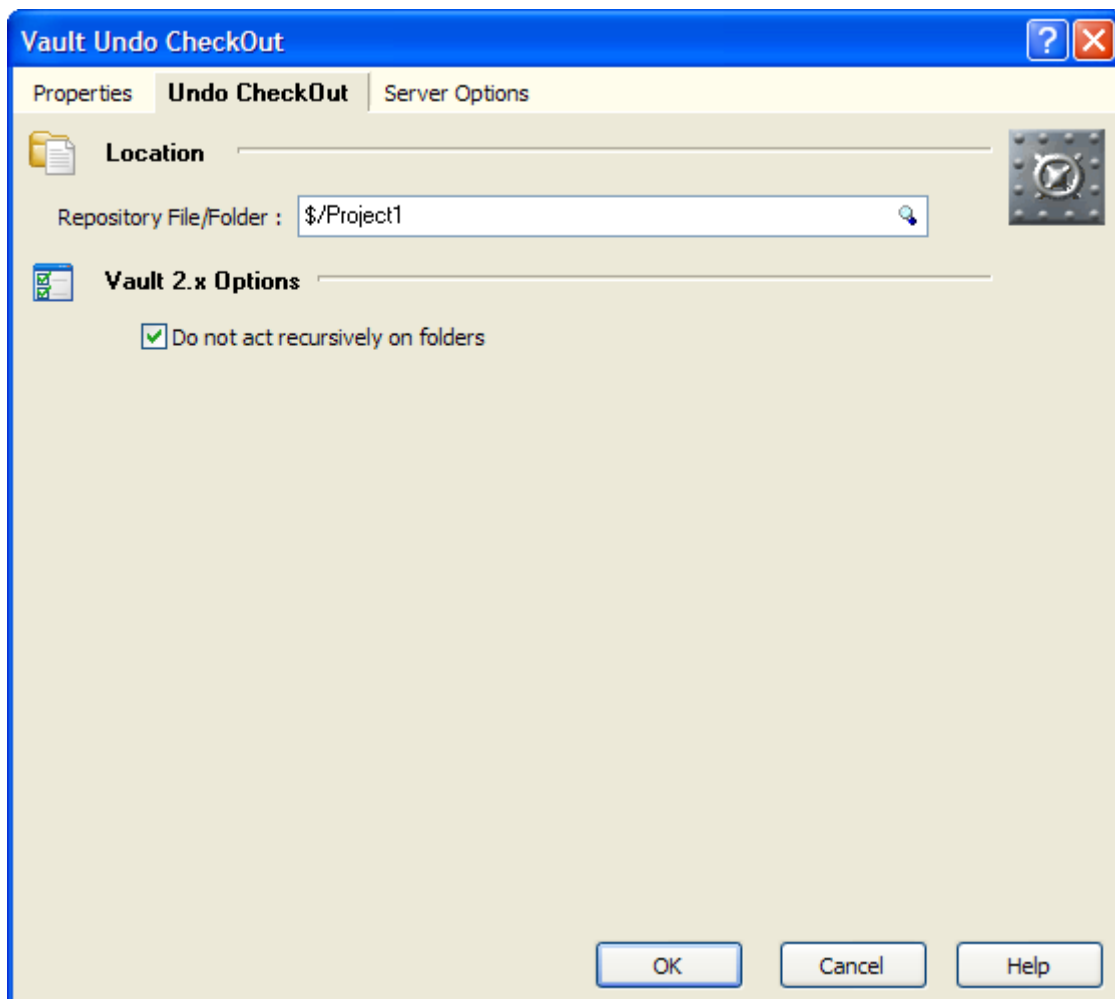
#### 5.2.8.16 Vault Share File/Folder Action

Share a file/folder to a new location in the vault repository.



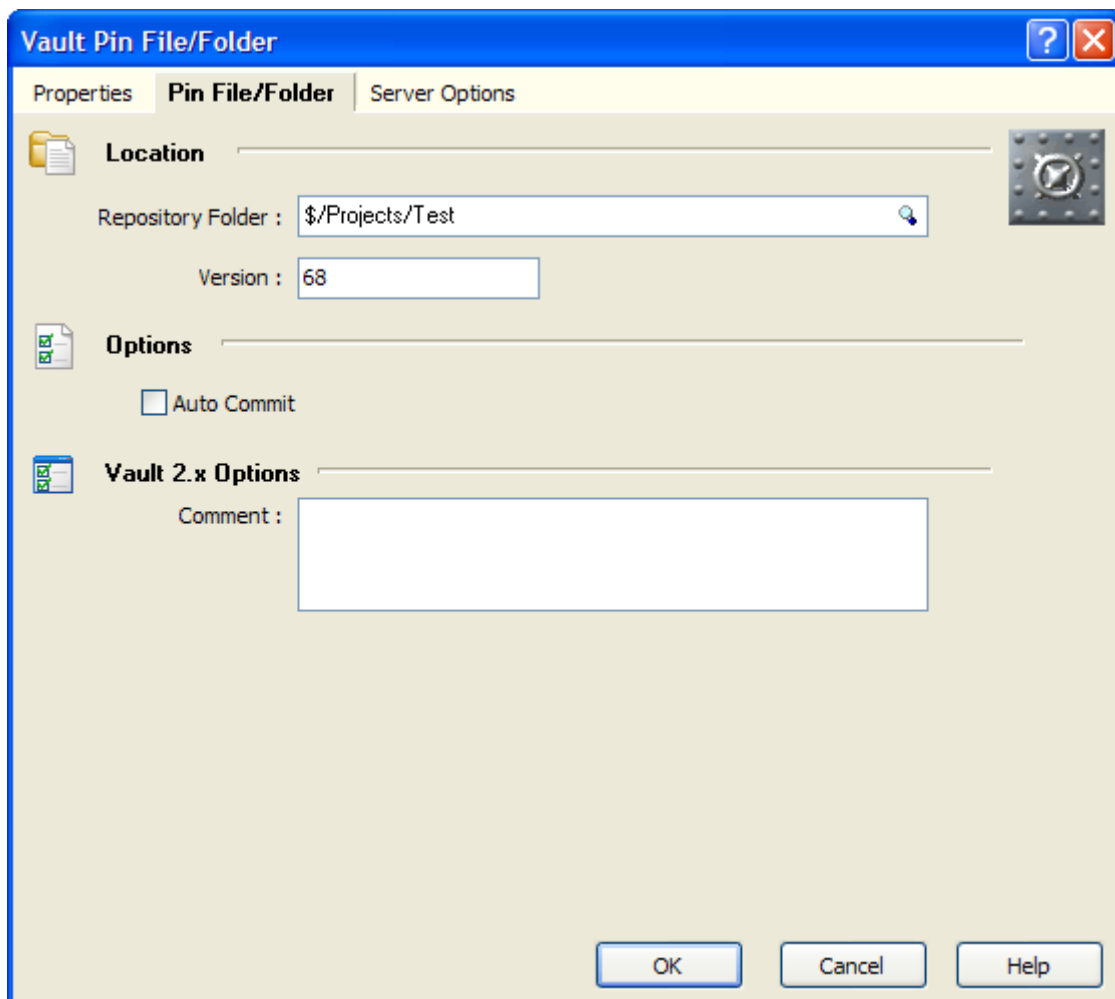
#### 5.2.8.17 Vault Undo Checkout Action

The vault undo checkout action enables you to undo the checkout of any files in the specified repository folder, or will undo the checkout of a single file if a file in the repository is specified.



#### 5.2.8.18 Vault Pin File/Folder Action

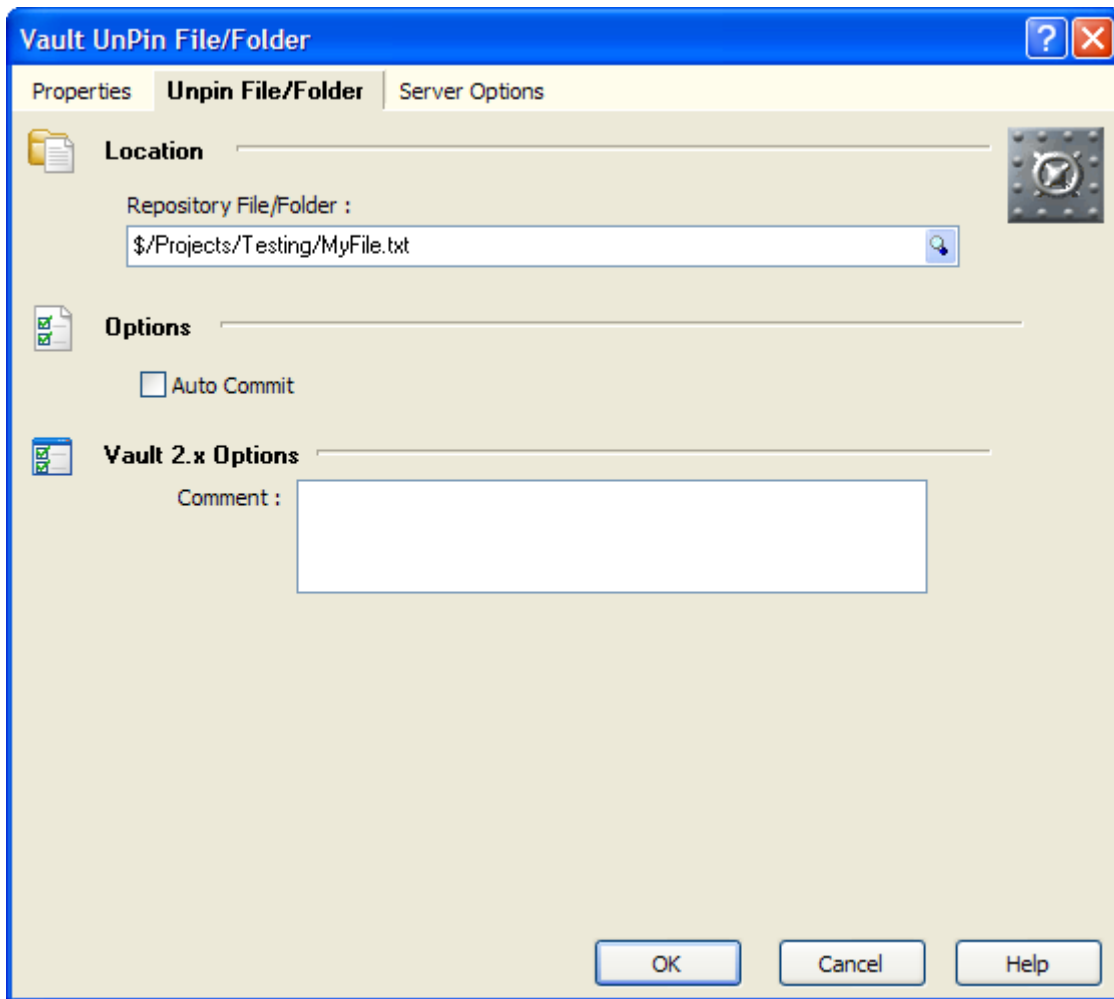
The vault Pin action allows you to Pin the specified file or folder in the repository to a particular version



#### 5.2.8.19 Vault UnPin File/Folder Action

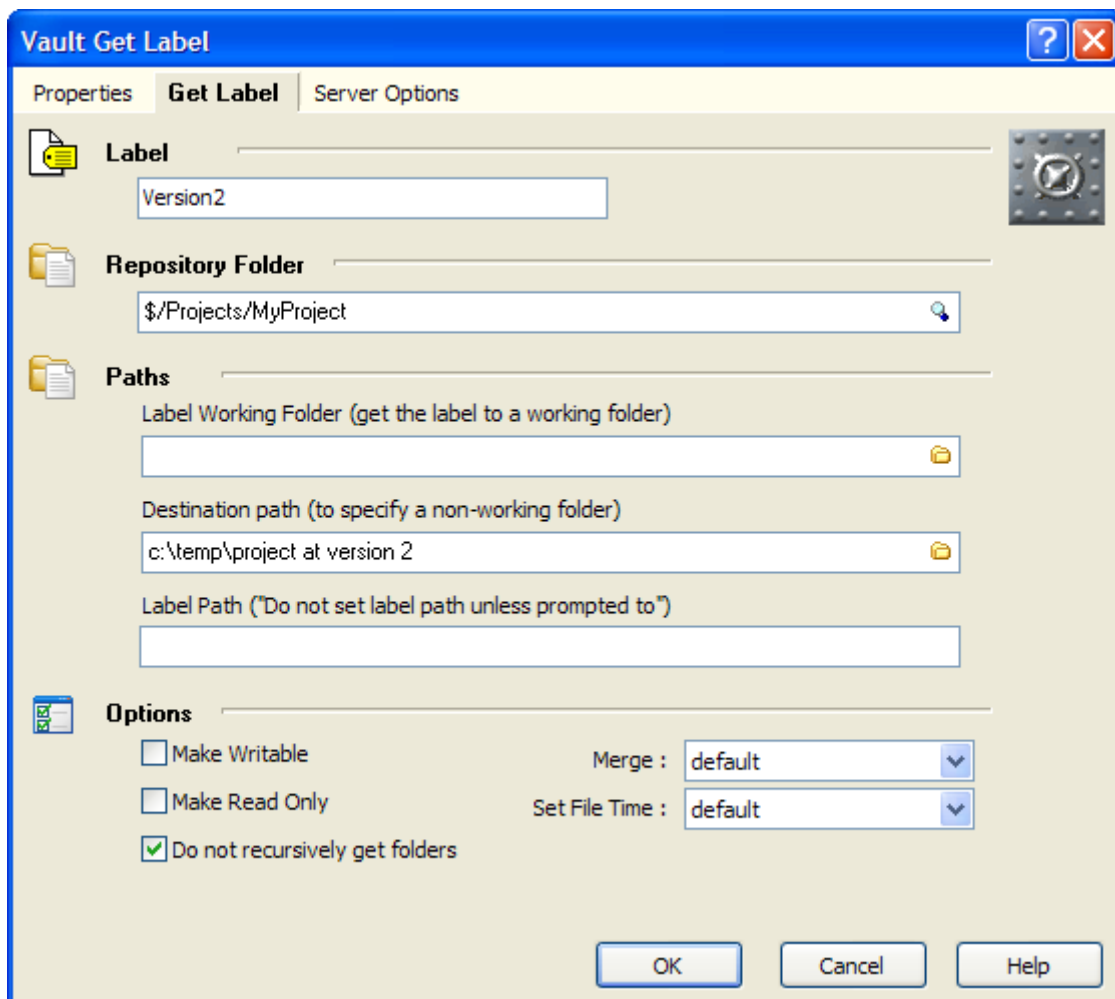
The Vault UnPin action allows you remove any Pins on the specified file or folder in the repository





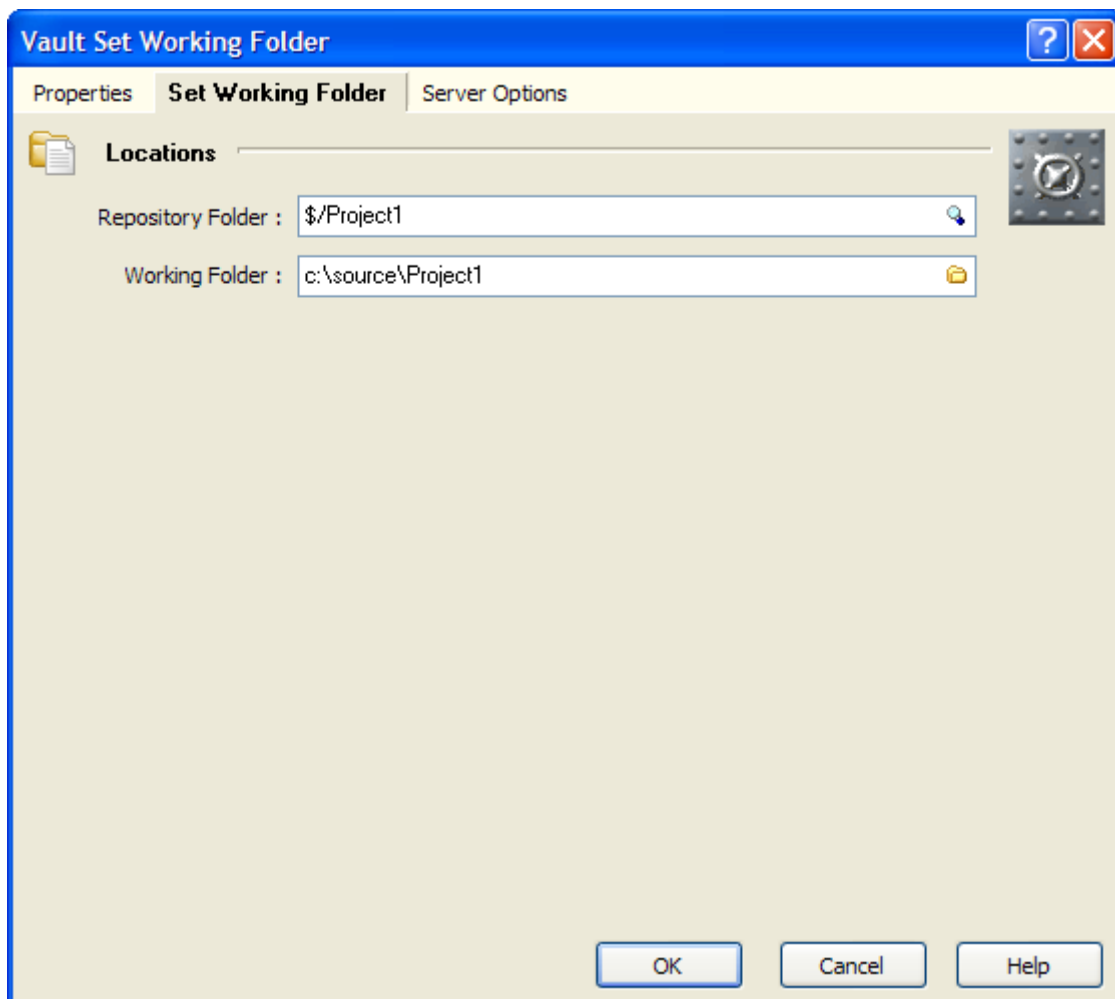
#### 5.2.8.20 Vault Get Label

The Vault Get Label action allows you to get files from the repository at a particular label.



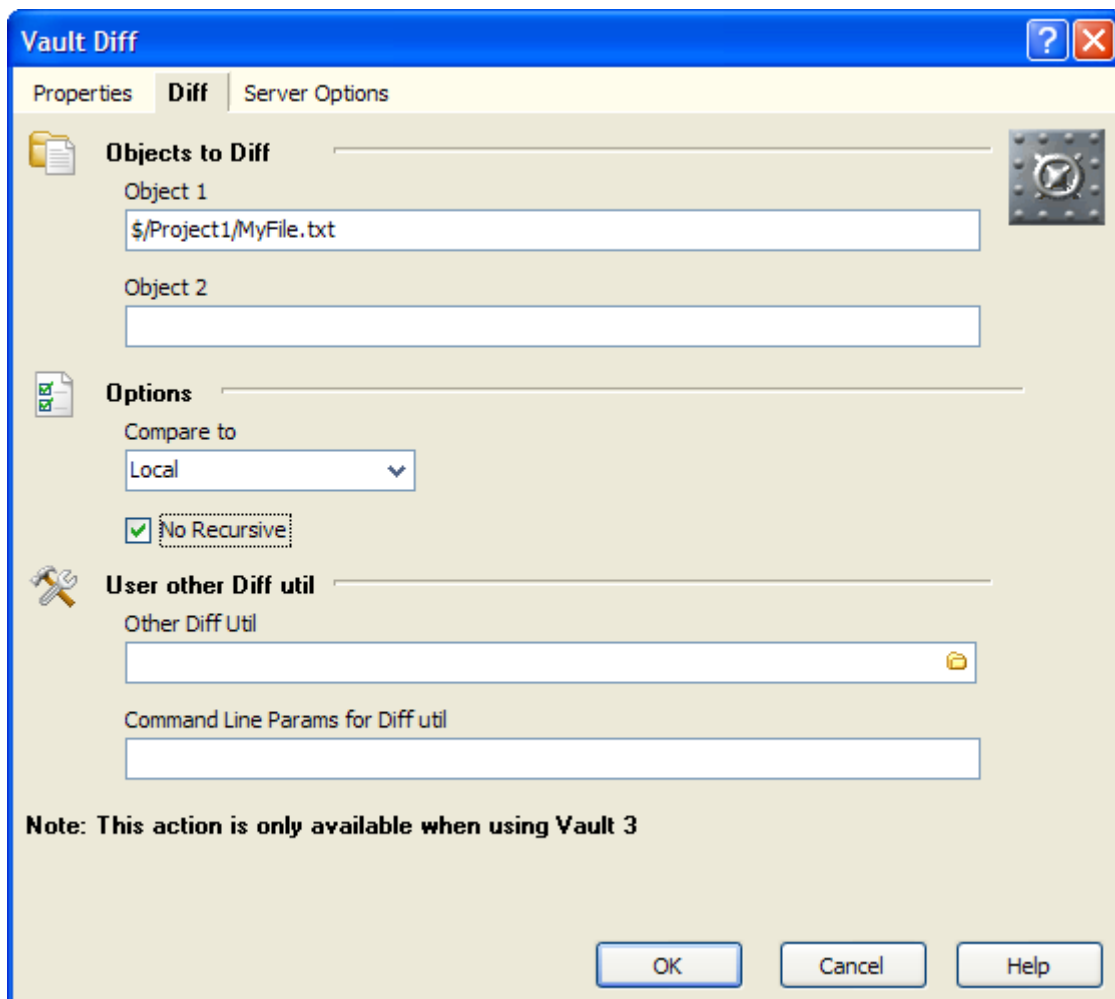
#### 5.2.8.21 Vault Set Working Folder

The vault set working folder action will set the working folder of a specified repository folder to a different location.



#### 5.2.8.22 Vault Diff

The vault Diff action allows you to compare objects using various methods within and external to the vault repository.



#### 5.2.8.23 Vault GetLabelDiffs

Executes the GetLabelDiffs in Vault 3.0

**Vault GetLabelDiffs**

Properties | **Label Diff** | Server Options

**Repository Folder**

\$/Project1

**Labels**

Label 1  
Version1

Label 2 Path (optional)

**Options**

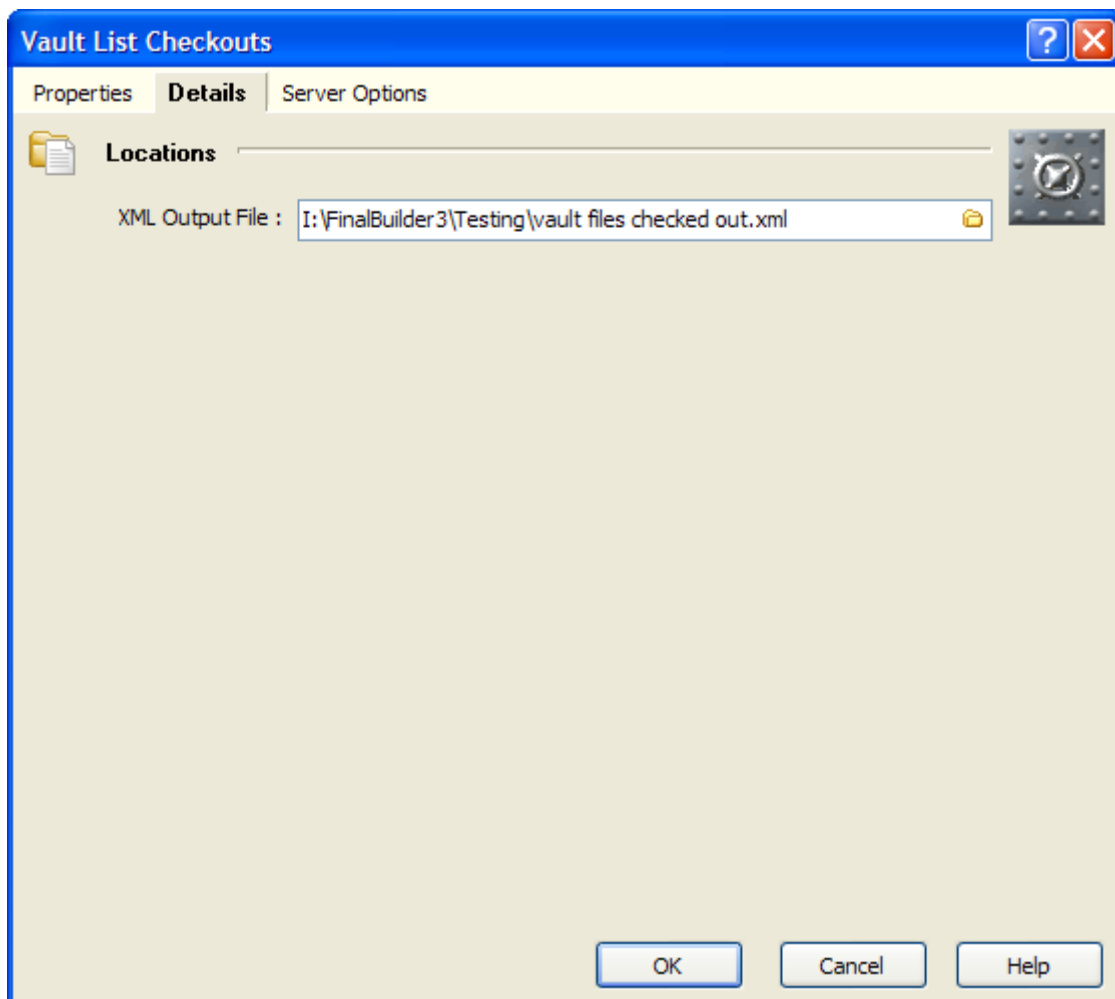
Row Limit (set to 0 for no limit)  
0

**Note: This action is only available when using Vault 3**

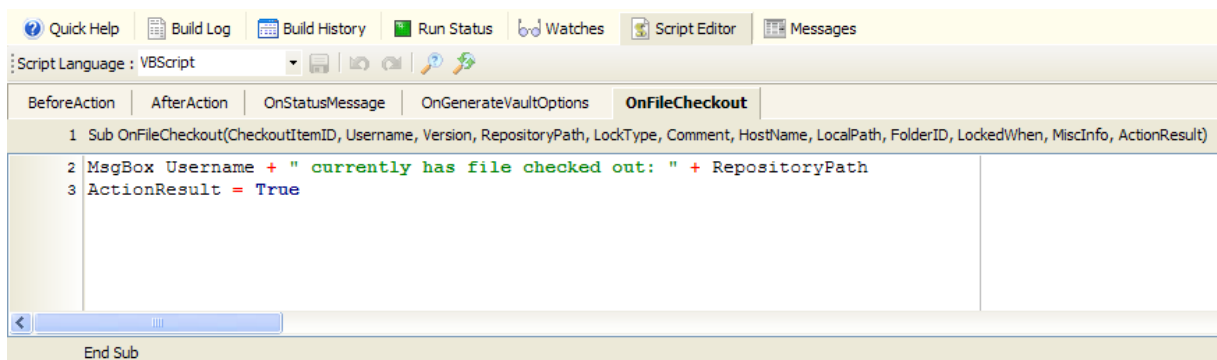
OK Cancel Help

#### 5.2.8.24 Vault ListCheckouts

The Vault ListCheckouts action enables you to query a vault repository for a report on all files currently checked out. The action can output the report as an xml file, or you can use the OnFileCheckout script event to process each file checked out by each user.



And to process each file checked out by each user, you can add code in the OnFileCheckout script event:



Setting the ActionResult to False will cause the action to fail after it has processed all of the files.

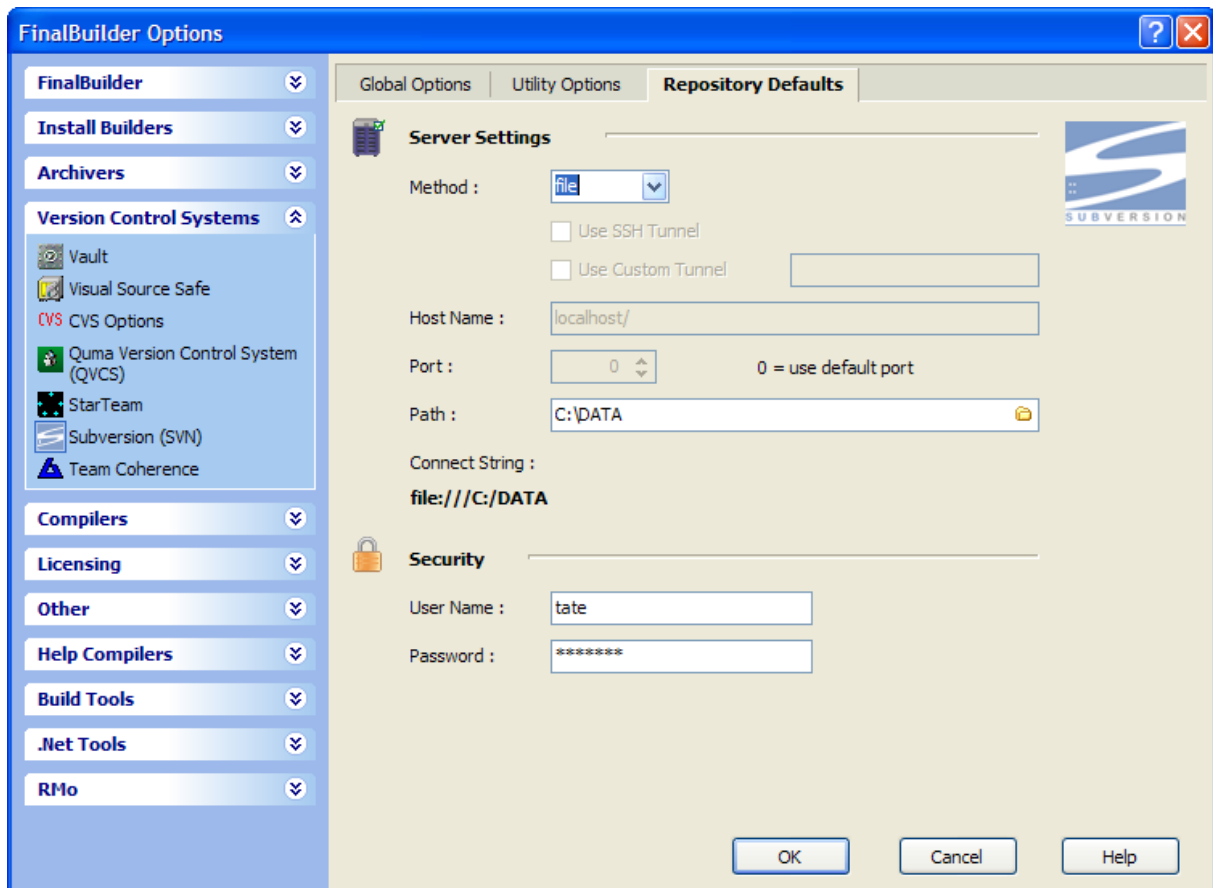
### 5.2.8.25 Vault File Status

Enter topic text here.

## 5.2.9 Subversion Actions

The Subversion actions support the Subversion version control system.

For more information about Subversion, see the homepage at <http://subversion.tigris.org/>



Before you start using the Subversion actions in FinalBuilder, you should set up the global options, utility options and repository defaults in the Tools->Options dialog.

### 5.2.9.1 Subversion Add

The Add command will put files and directories under version control, scheduling them for addition to repository. They will be added in next commit.

### 5.2.9.2 Subversion Checkout

The Checkout command will check out a working copy from a repository.

**5.2.9.3 Subversion Cleanup**

The Cleanup command will recursively clean up the working copy, removing locks, resuming unfinished operations, etc.

**5.2.9.4 Subversion Copy**

The Copy command will duplicate something in working copy or repository, remembering history.

**5.2.9.5 Subversion Commit**

The Cleanup command will send changes from your working copy to the repository.

**5.2.9.6 Subversion Export**

The Export command will create an unversioned copy of a tree.

**5.2.9.7 Subversion Import**

The Import command will commit an unversioned file or tree into the repository.

**5.2.9.8 Subversion Mkdir**

The Mkdir command will create a new directory under version control.

**5.2.9.9 Subversion Update**

The Update command will bring changes from the repository into the working copy.

**5.2.9.10 Subversion Status**

The status command gets the status of working copy files and directories.

**5.2.10 Seapine Surround SCM**

The Surround SCM actions provide FinalBuilder integration for Seapine Surround SCM. The actions were developed with version 3.1.x of Surround SCM.

For more information on Surround SCM, see <http://www.seapine.com>

Before you start using the Surround SCM actions, it is suggested you first set up Global Surround SCM Options

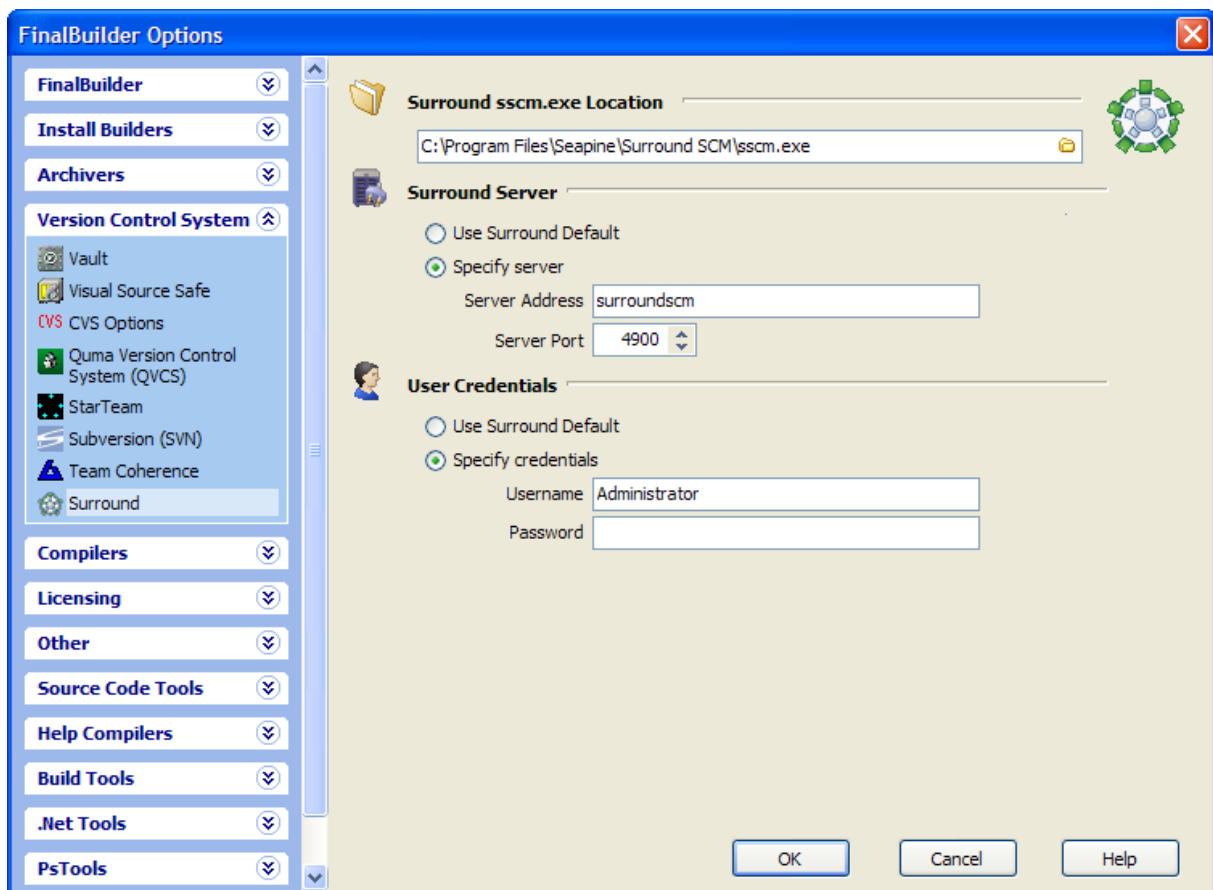
The Surround SCM operations supported are: Get, CheckOut, CheckIn, Label, Create Branch, Freeze Branch, Unfreeze Branch, Checkout Report.

If the Surround SCM operation you want to perform is not supported by one of the FinalBuilder actions, you can use the Surround SCM Generic action.

**5.2.10.1 Surround SCM Global Options**

The Surround SCM Global Options allow you to set up your FinalBuilder integration with default options for any Surround SCM actions used in any build, as well as the path to the Surround SCM command line interface (this is what FinalBuilder uses to perform the Surround SCM actions).





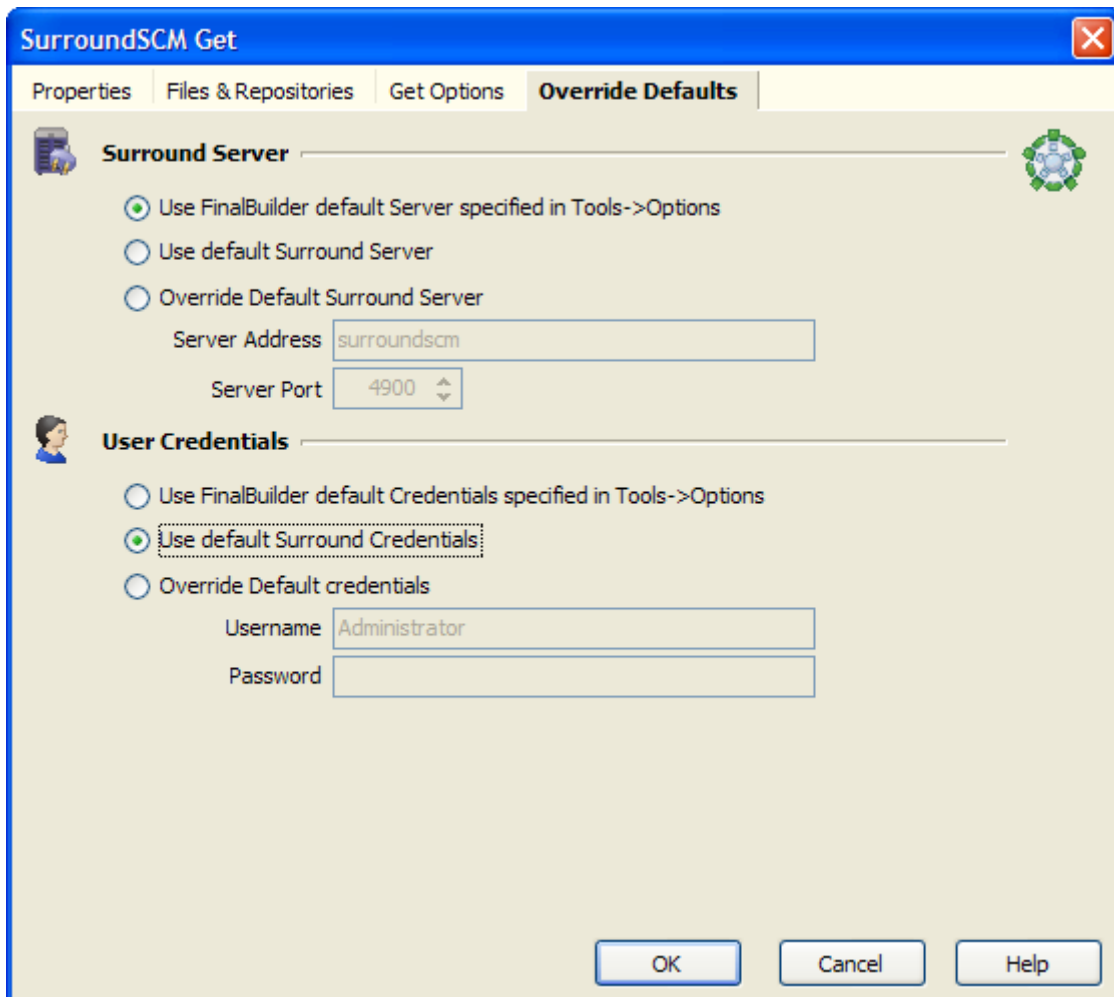
Access the Global Options for Surround via the Tools->Options menu in FinalBuilder. Then select the Version Control System category, and finally the Surround item.

FinalBuilder will attempt to autodetect the sscm.exe location - if the Surround sscm.exe Location is blank, then specify the location of sscm.exe.

You can specify both the default server and user credentials for all the Surround SCM actions. Either specify the specific values, or use the Surround SCM defaults (in this case no values will be specified on the command line). Note that in the any Surround SCM action, you can either use the global options specified here, or override the values.

#### 5.2.10.1.1 Surround SCM Override Global Options

Each Surround SCM action in FinalBuilder allows you to override the Global Surround SCM options.



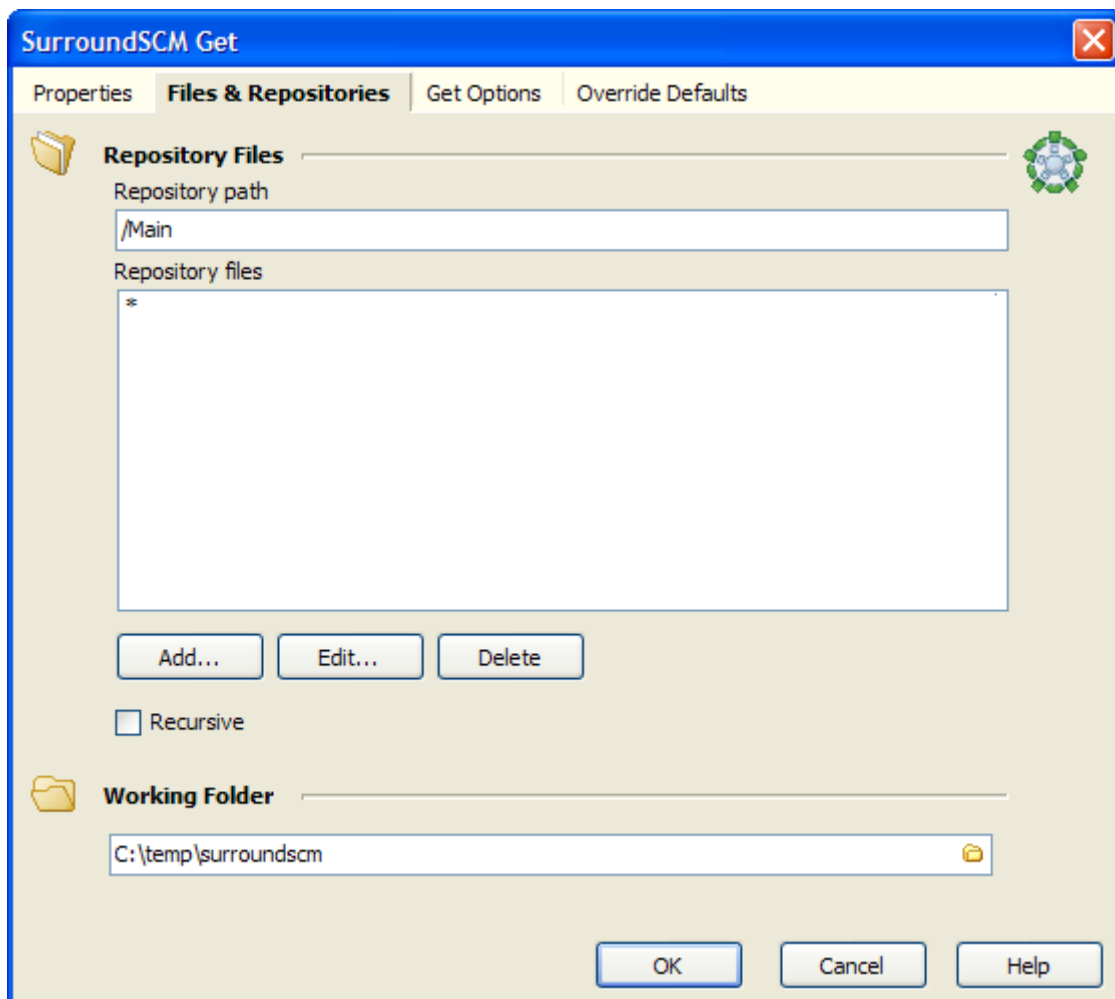
To specify the FinalBuilder global options, see Surround SCM Global Options

#### 5.2.10.2 Surround SCM Get

Get files when you want to view a file but do not need to make any changes. You can get a single file, multiple files, or a repository. A read-only copy of the file is created in the specified directory.

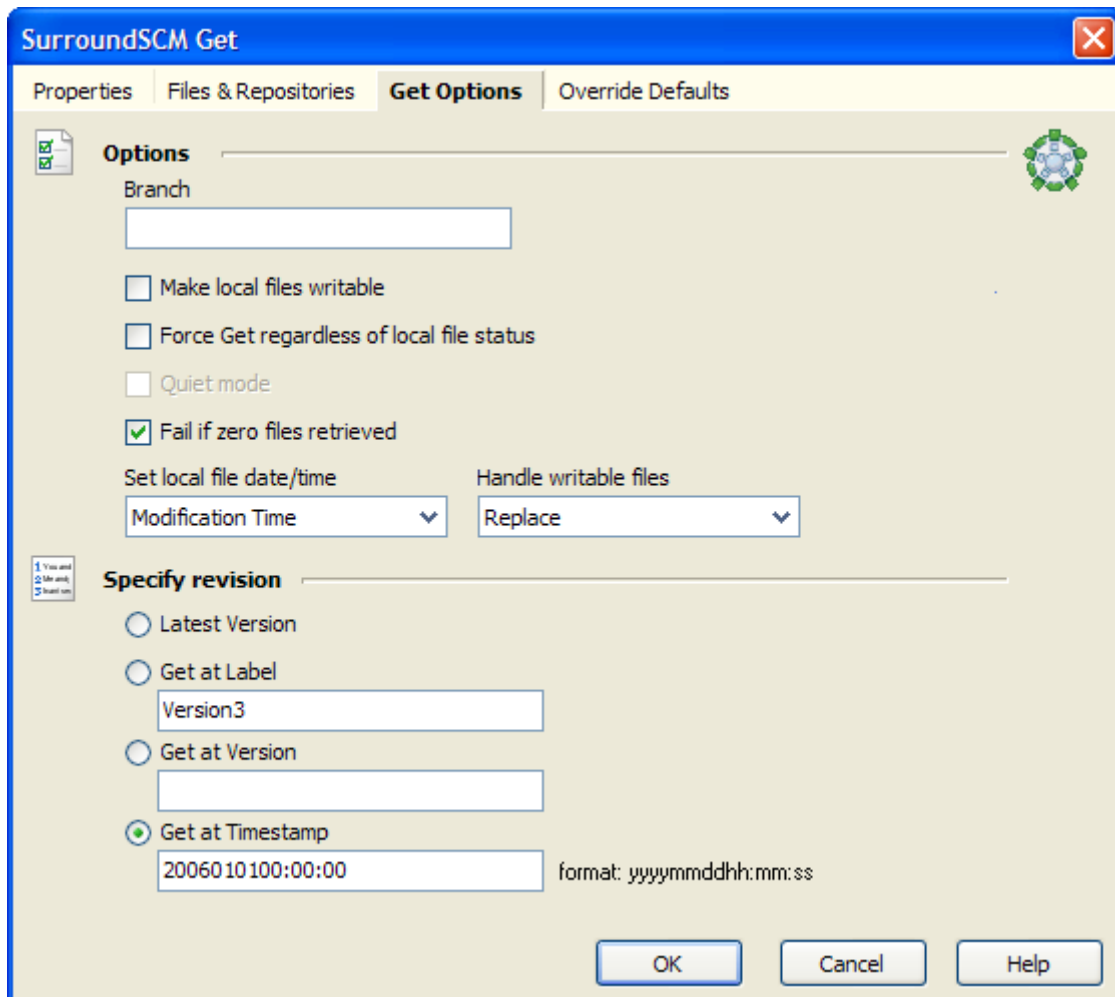
##### Files & Repositories Tab

Specify the repository path and a list of files to retrieve. You can use \* to specify all files. Specify Recursive to Recursively get files and subrepositories.



### Get Options Tab

Specify the Get options - all values are optional.

**Tips:**

If you don't specify a working folder, then Surround will use the current working folder for the repository selected.

**5.2.10.3 Surround SCM CheckOut**

Check out files when you need to make changes. You can check out single files, multiple files, or a repository. Surround SCM creates a read-write copy of the file in the working directory.

The Checkout action uses the same Files & Repositories Tab as the Surround SCM Get action.

**Checkout Options Tab**

Specify the checkout options - all values are optional.

The screenshot shows the 'SurroundSCM CheckOut' dialog box with the 'Checkout Options' tab selected. The dialog has a title bar with a close button. Below the title bar are four tabs: 'Properties', 'Files & Repositories', 'Checkout Options' (active), and 'Override Defaults'. The 'Options' section includes a 'Branch' text field with 'MyBranch' entered, and several checkboxes: 'Make local files writable' (unchecked), 'Exclusive Lock' (unchecked), 'Quiet mode' (unchecked), 'Fail if zero files checked out' (checked), and 'Force Checkout regardless of local file status' (checked). There are two dropdown menus: 'Set local file date/time' set to 'Checkin Time' and 'Handle writable files' set to 'Skip'. The 'Comment' section has a large text area. The 'Specify revision' section has two radio buttons: 'Latest Version' (selected) and 'Checkout at Version' (unchecked), with a text field below it. At the bottom are 'OK', 'Cancel', and 'Help' buttons.

**SurroundSCM CheckOut**

Properties Files & Repositories **Checkout Options** Override Defaults

**Options**

Branch  
MyBranch

☐ Make local files writable ☐ Exclusive Lock

☐ Quiet mode ☒ Fail if zero files checked out

☒ Force Checkout regardless of local file status

Set local file date/time: Checkin Time  
Handle writable files: Skip

**Comment**

**Specify revision**

☒ Latest Version  
☐ Checkout at Version

OK Cancel Help

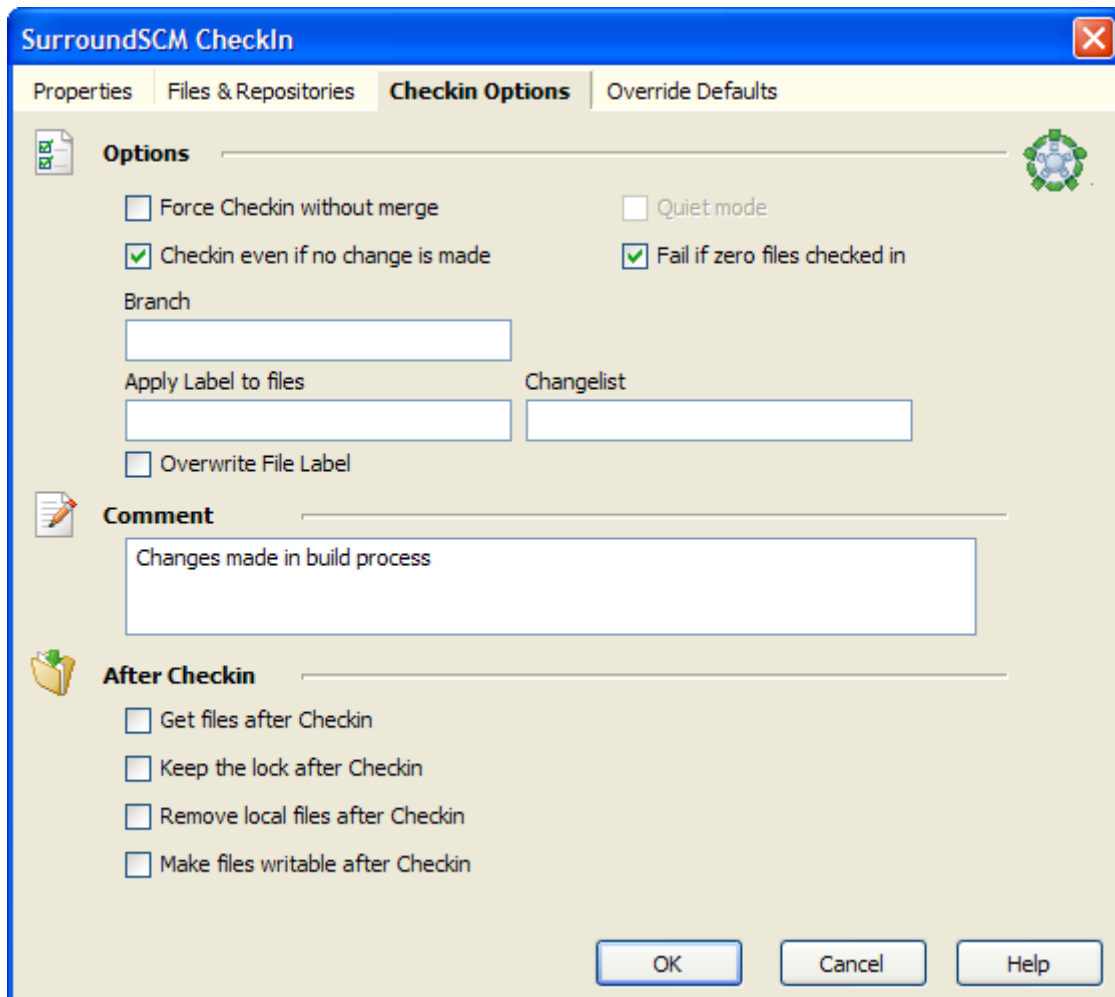
#### 5.2.10.4 Surround SCM CheckIn

Check in updates Surround SCM files with changes, removes the lock on the files, and makes changes available to other users.

The Checkin action uses the same Files & Repositories Tab as the Surround SCM Get action.

#### Checkin Options Tab

Specify the checkin options - all values are optional.



Branch - Enter the branch name to check in the changes to. Default is set in the working directory.

Label - Enter a label for the check in code.

Changelist - Enter a new or existing changelist name to check in the file as part of a changelist.

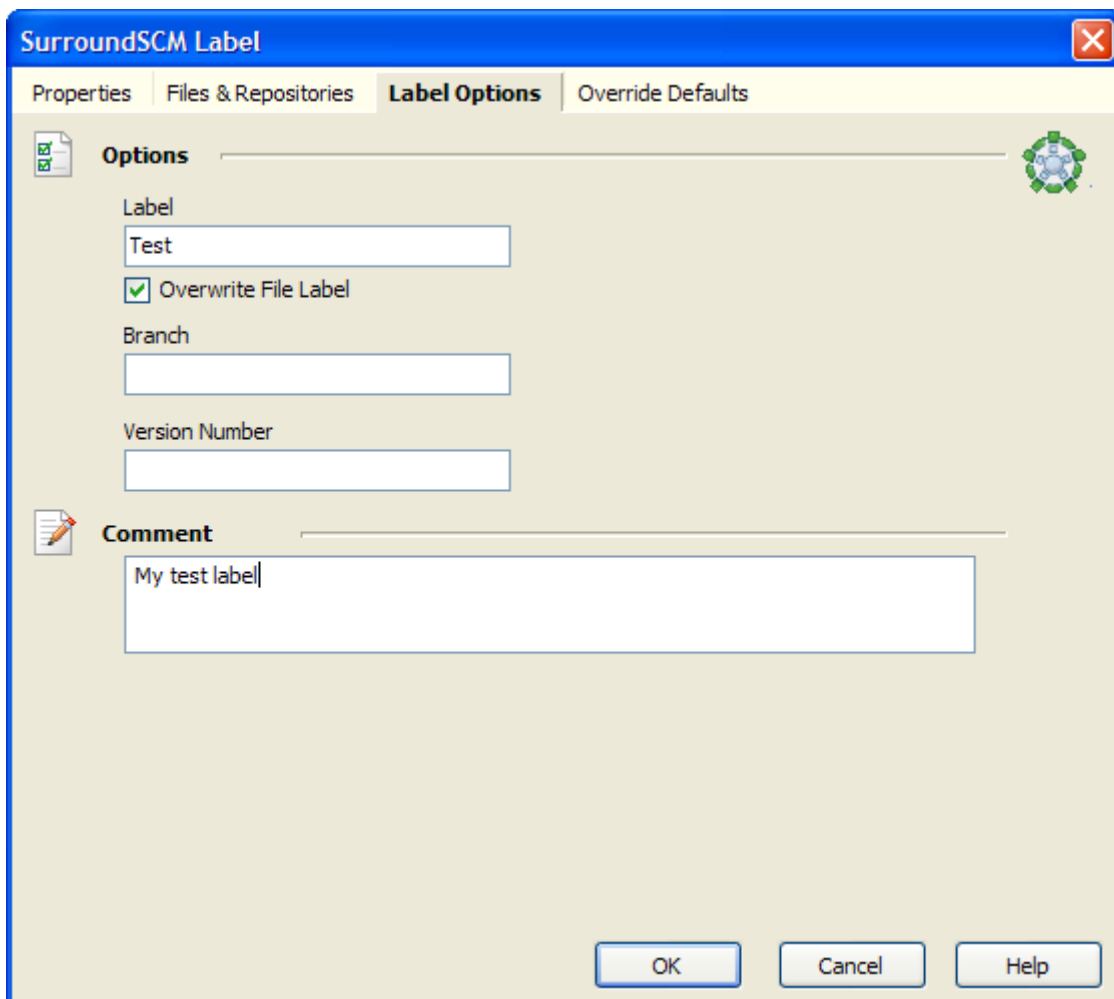
#### 5.2.10.5 Surround SCM Label

Labels provide a way to mark a specific version of a file or repository. When you create a label, a new entry is created in the history. The file and the version number do not change.

The Label action uses the same Files & Repositories Tab as the Surround SCM Get action.

#### Label Options Tab

Specify the new label values - all values are optional except the Label name.



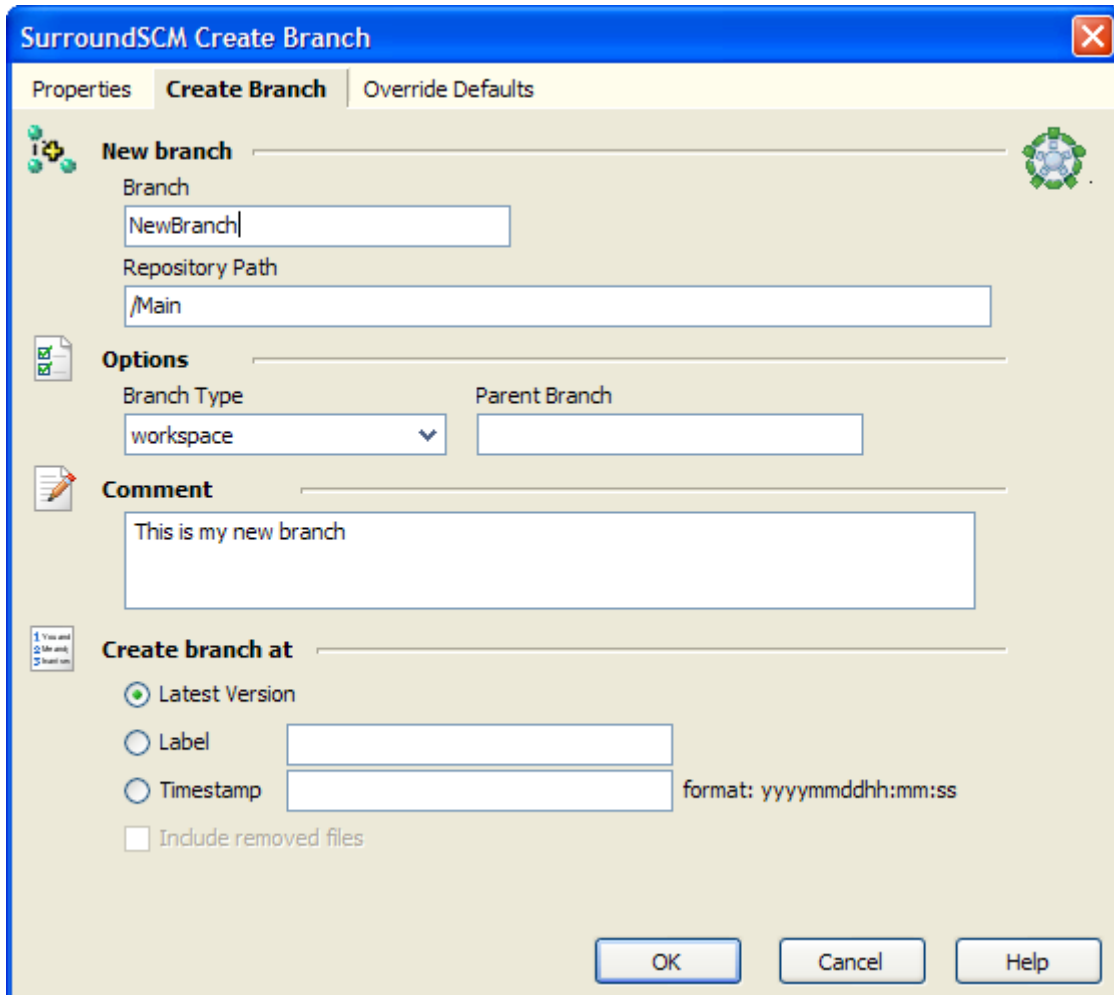
The image shows a Windows-style dialog box titled "SurroundSCM Label". It has a blue title bar with a close button (X) in the top right corner. Below the title bar is a tabbed interface with four tabs: "Properties", "Files & Repositories", "Label Options" (which is currently selected), and "Override Defaults". The "Label Options" tab contains two main sections. The first section, titled "Options" with a document icon, includes a "Label" text field containing the word "Test", a checked checkbox labeled "Overwrite File Label", a "Branch" text field, and a "Version Number" text field. The second section, titled "Comment" with a pencil icon, contains a large text area with the text "My test label". In the bottom right corner of the dialog, there are three buttons: "OK", "Cancel", and "Help".

Tip: To label all files in a repository then use "/" as the repository file spec (without quotes).

#### 5.2.10.6 Surround SCM Create Branch

The Create Branch action allows creation of a new workspace, baseline or snapshot branch in Surround SCM.

#### Create Branch Tab



The image shows the 'SurroundSCM Create Branch' dialog box. It has three tabs: 'Properties', 'Create Branch' (selected), and 'Override Defaults'. The 'Create Branch' tab contains the following fields and options:

- New branch**: A section with a 'Branch' text box containing 'NewBranch' and a 'Repository Path' text box containing '/Main'.
- Options**: A section with a 'Branch Type' dropdown menu set to 'workspace' and a 'Parent Branch' text box.
- Comment**: A text area containing the text 'This is my new branch'.
- Create branch at**: A section with three radio buttons: 'Latest Version' (selected), 'Label' (with an empty text box), and 'Timestamp' (with an empty text box). To the right of the 'Timestamp' text box is the text 'format: yyyyymmddhh:mm:ss'. Below these is a checkbox labeled 'Include removed files'.

At the bottom of the dialog are three buttons: 'OK', 'Cancel', and 'Help'.

Branch - enter the new branch name

Repository Path - the repository path for the new branch

Branch Type - specify workspace, baseline or snapshot

Parent Branch - specify the parent branch for the new branch

Create Branch At:

Latest Version - the most recent version of the files are used

Label - This option specifies which parent branch file versions are copied into the child branch, specified by the label.

Timestamp - Enter the timestamp (local time). This option specifies which parent branch file versions

are copied into the new child branch. Date/time format: yyyyymmddhh:mm:ss

#### 5.2.10.7 Surround SCM Freeze Branch

Freezing a branch prevents any code changes being made to files in the branch. When a branch is frozen, it is locked and no changes can be made to it.



#### 5.2.10.8 Surround SCM Unfreeze Branch

Unlock a frozen branch.

#### 5.2.10.9 Surround SCM Checkout Report

The Checkout report action lists the files checked out in the repository. The action can also automatically fail if any files are checked out.

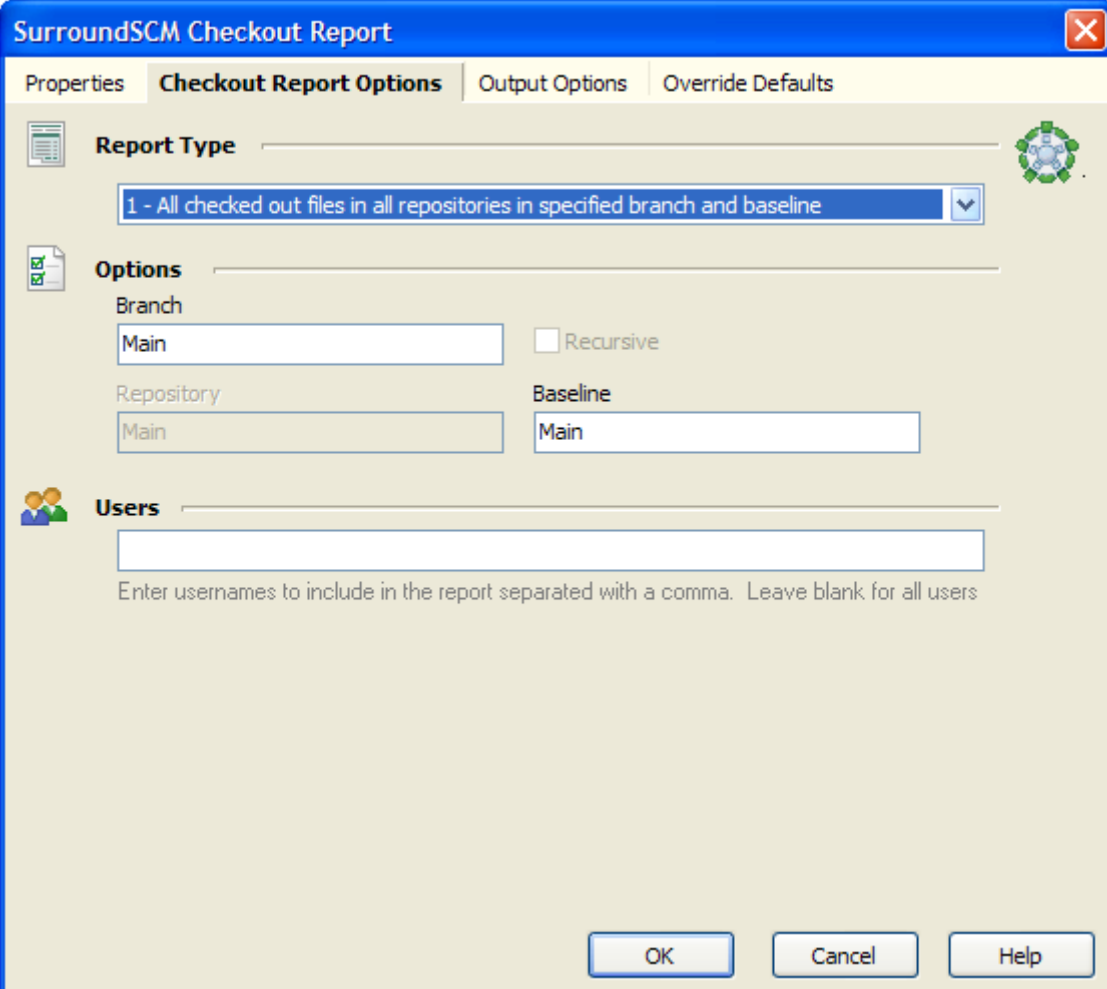
##### Checkout Report Options

There are five report types:

1. All checked out files in all repositories in the specified branch and baseline
2. All checked out files in the specified repository and branch
3. All checked out files in all repositories and branches in the specified baseline
4. All checked out files in the specified repository across all branches
5. All checked out files

Depending on which report type is chosen, the Branch, Repository, Baseline and Recursive options will be enabled/disabled to allow you to fill in the correct fields.

To further filter the results by user, enter one or more user names separated by a comma.



The screenshot shows the 'SurroundSCM Checkout Report' dialog box with the 'Checkout Report Options' tab selected. The dialog has a blue title bar and a yellow tab bar. The 'Report Type' section has a dropdown menu set to '1 - All checked out files in all repositories in specified branch and baseline'. The 'Options' section includes text boxes for 'Branch' (Main), 'Repository' (Main), and 'Baseline' (Main), along with an unchecked 'Recursive' checkbox. The 'Users' section has a large text box for entering usernames, with a note below it: 'Enter usernames to include in the report separated with a comma. Leave blank for all users'. The bottom of the dialog features 'OK', 'Cancel', and 'Help' buttons.

**SurroundSCM Checkout Report**

Properties **Checkout Report Options** Output Options Override Defaults

**Report Type**

1 - All checked out files in all repositories in specified branch and baseline

**Options**

Branch: Main

Repository: Main

Baseline: Main

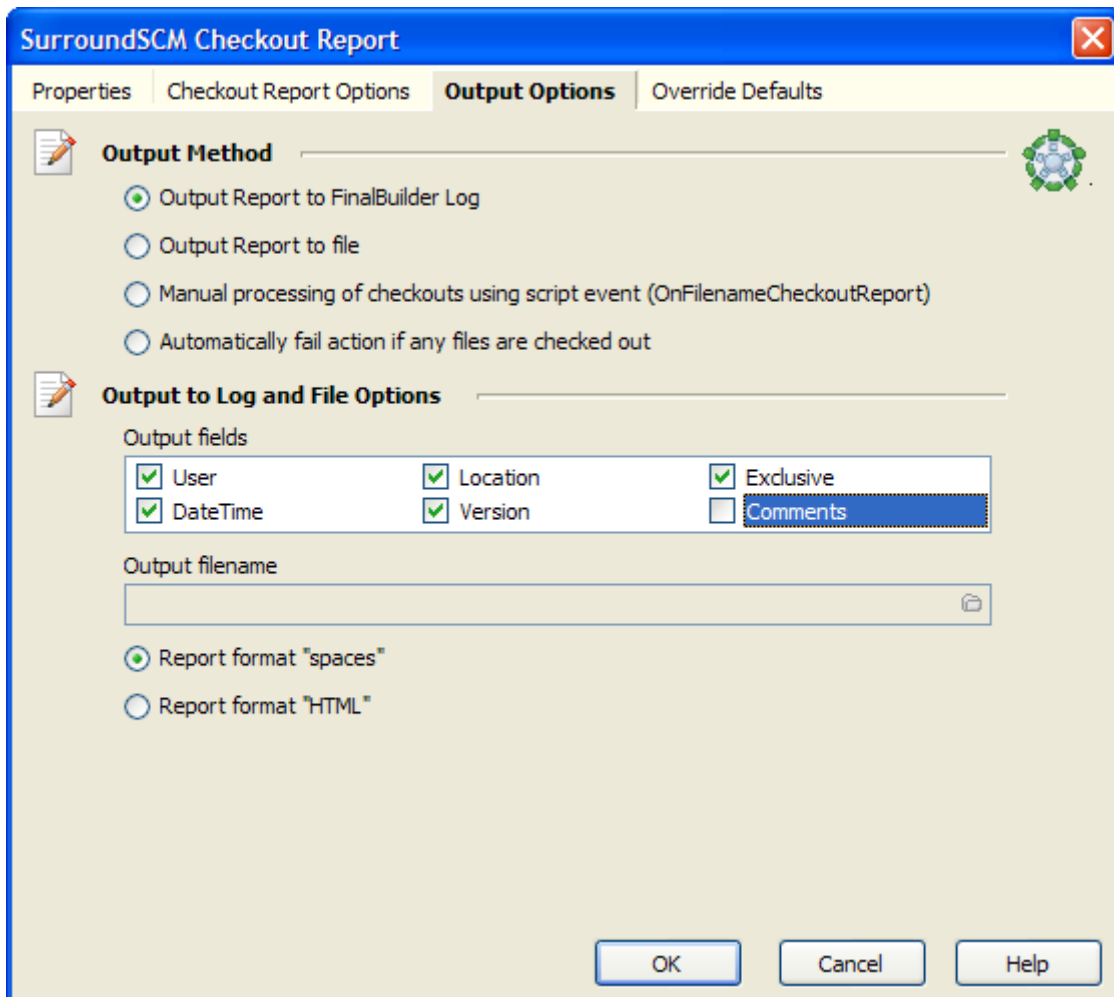
☐ Recursive

**Users**

Enter usernames to include in the report separated with a comma. Leave blank for all users

OK Cancel Help

### Output Options



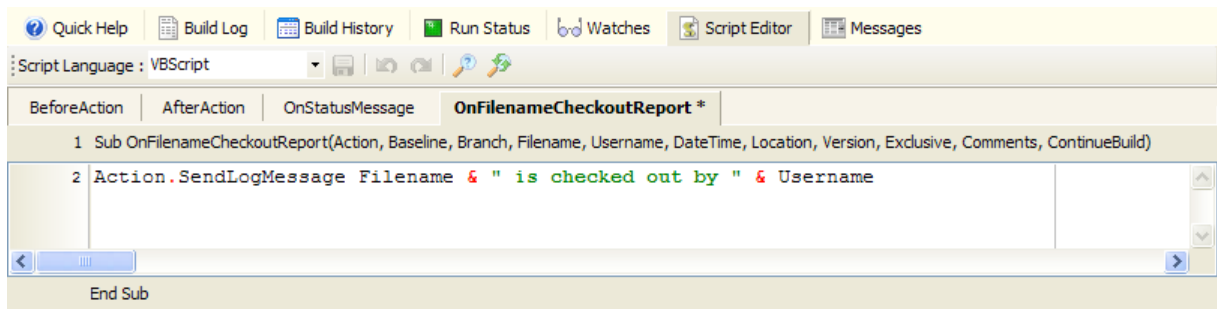
There are four possible modes when this action runs:

1. Output Report to FinalBuilder log
2. Output Report to file

Both these options allow you to specify which fields to output, eg. User, DateTime, Version etc. and also if the report should be formatted using spaces or in HTML.

3. Manual processing of checkouts using script event (OnFilenameCheckoutReport)
4. Automatically fail action if any files are checked out.

Option 3 allows the user to write script (VBScript or JScript) to process each checkout. The processing occurs in the OnFilenameCheckoutReport script event.



The above screen pic shows a very simple script which is executed for each file checked out. The parameters are as follows:

Action - the action object.

Baseline - the name of the baseline where the checked out file is located

Branch - the name of the branch where the checked out file is located

Filename - the filename which is checked out

Username - the user who has the file checked out

DateTime - the date/time when the file was checked out

Location - the location of the checked out file

Version - the version of the file

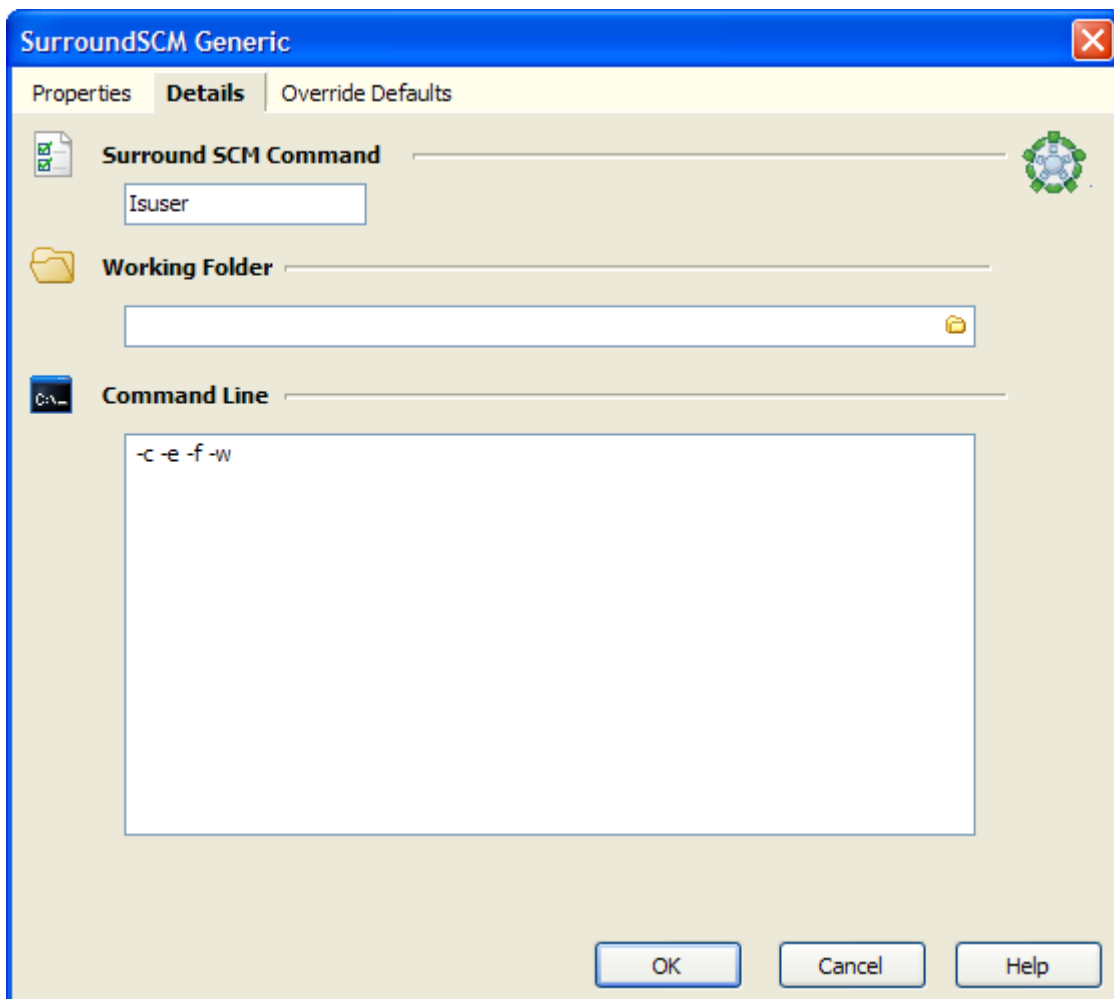
Exclusive - specifies if the file is exclusively checked out

Comments - checked out comments of the file

ContinueBuild - out parameter, set this to False if you want to fail the action.

#### 5.2.10.10 Surround SCM Generic

If the Surround SCM operation you want to perform is not covered by the built in FinalBuilder Surround SCM actions, then you can use the Surround SCM Generic action. The action requires you specify the SCM Command, the working folder to run the command from, and the command line to send to sscm.exe.

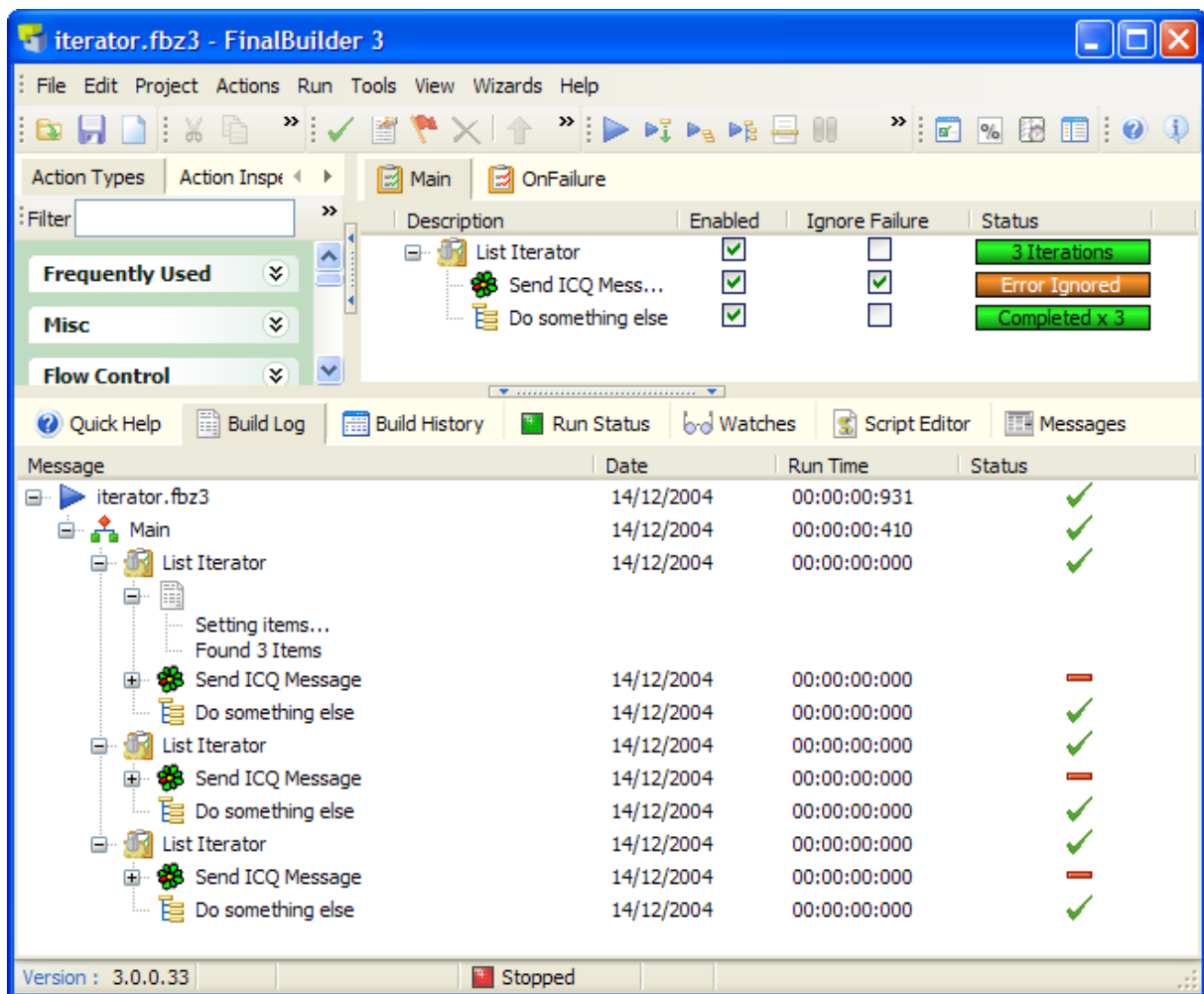


See the "Surround SCM CLI Guide.PDF" file for more information on the Surround SCM commands and command line options.

## 5.3 Iterators

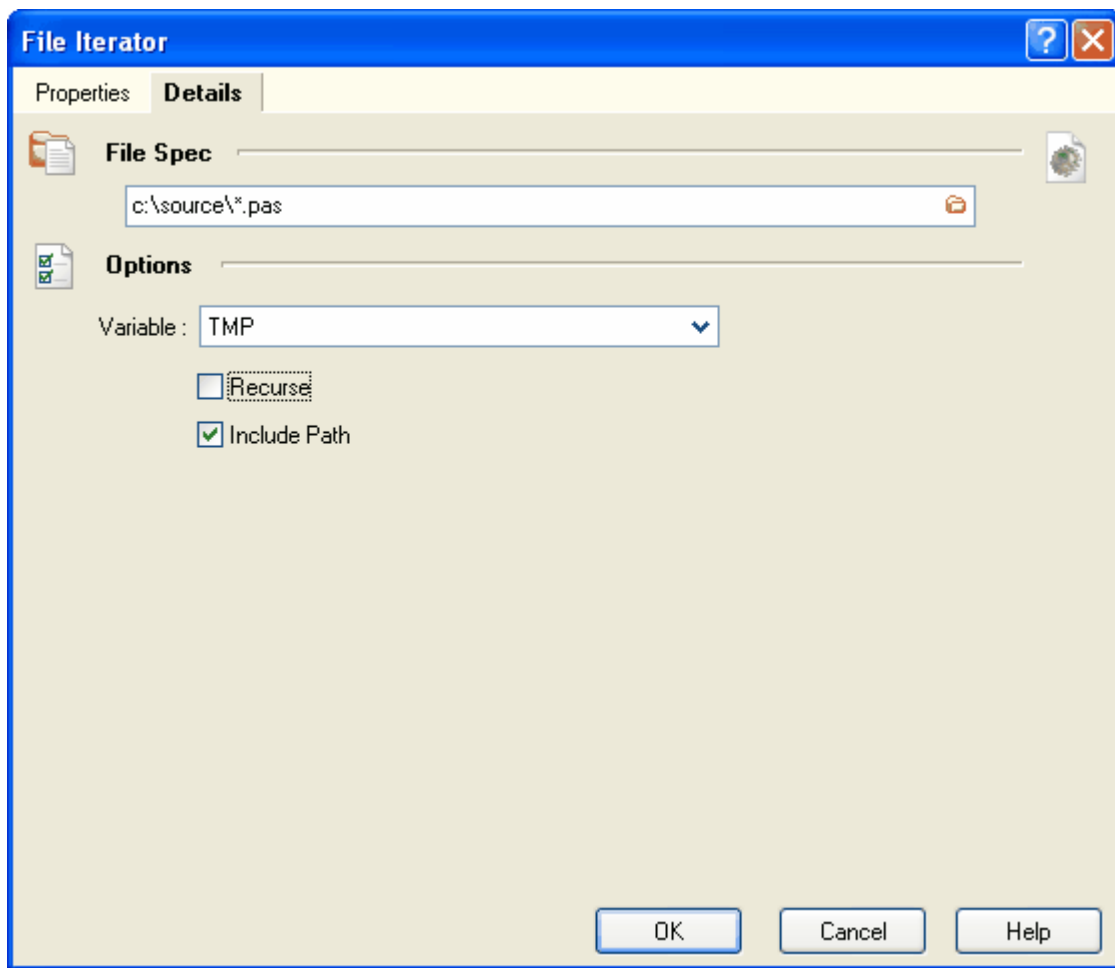
Iterator actions work by executing their child actions for each item that the iterator finds.

For example, a list iterator might have 3 items: "A, B, and C". When the action executes the first time, it first figures out how many items there are. Typically the action will then set a variable of your choice to the first item and then run the child actions. So, for example, the variable "MyLetter" would be set to "A" and then the child actions are executed. Then it will set the variable to the next item in the list and run the children again - this continues for as many items in the list there are. Below is a screen shot of an iterator after the build has finished.



### 5.3.1 File Iterator

The File Iterator Action allows you to perform actions based on the files found from the file spec you provide. For example, if you wanted to process all the .obj files in a directory, you would set the file spec to `c:\mydir\*.obj`. The action will then execute its child actions for each .obj file it finds in `c:\mydir`, placing the file name of the file in the FinalBuilder variable you specify. You can then access this filename using the variable.



**File Spec :** The File specification for the files you wish to find.

**Variable :** The name of the FinalBuilder Project/User Variable to place the found filename in.

### Scripting Info

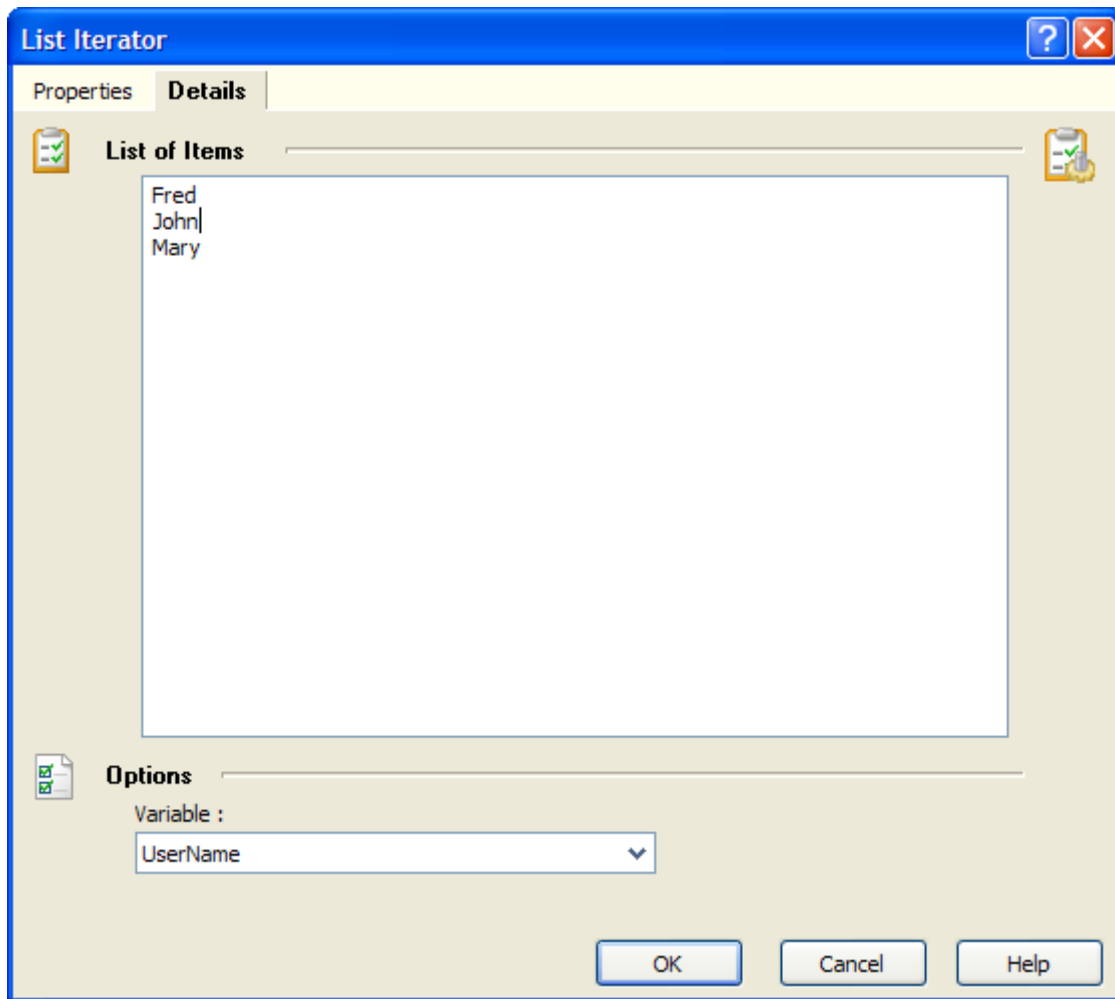
The Action properties available are :

**property** VariableName : WideString;

**property** FileSpec : WideString;

## 5.3.2 List Iterator

The List Iterator Action allows you to perform actions based on items in a list.



**List of Items :** This is the list of items to perform the action on. Each new line represents the new value.

**Variable :** The name of the FinalBuilder Project/User Variable to place the item in.

### Scripting Info

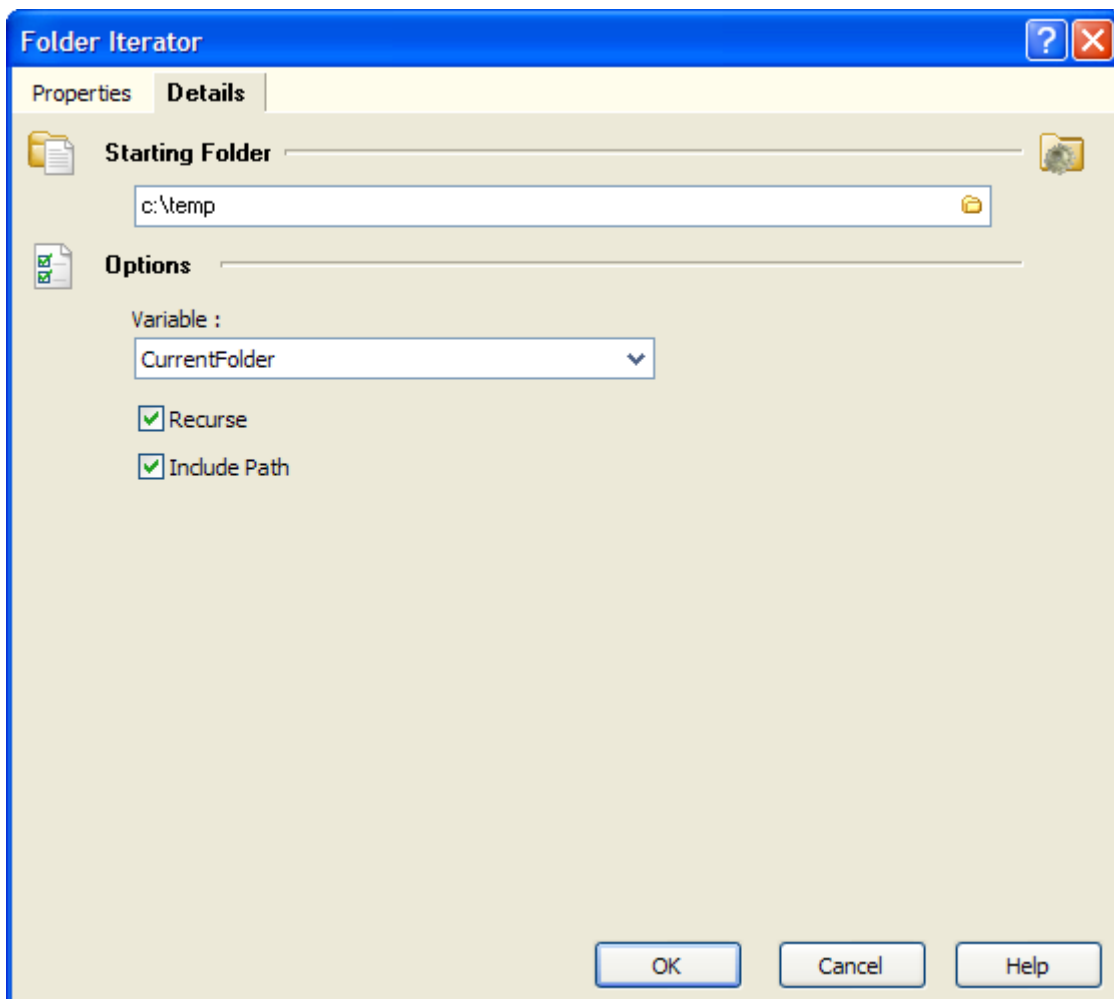
There is a scripting event "**OnFirstRun**" which allows you to set the list of items in script. Example script text:

```
Action.ListOfItems.clear  
Action.ListOfItems.Add("Item 1")  
Action.ListOfItems.Add("Item 2")
```

### 5.3.3 Folder Iterator

The Folder Iterator action enables you to repeat a set of build steps for one or more folders. Specify a starting folder (eg. c:\temp\) and then when the action executes any child actions will be executed for each folder found inside the starting folder.





**Starting Folder** - Enter the root folder to begin the search from

**Variable** - Specify the variable for the iterator to put in the current folder

**Recurse** - Specify recurse to continue finding folders within the folders inside the starting folder

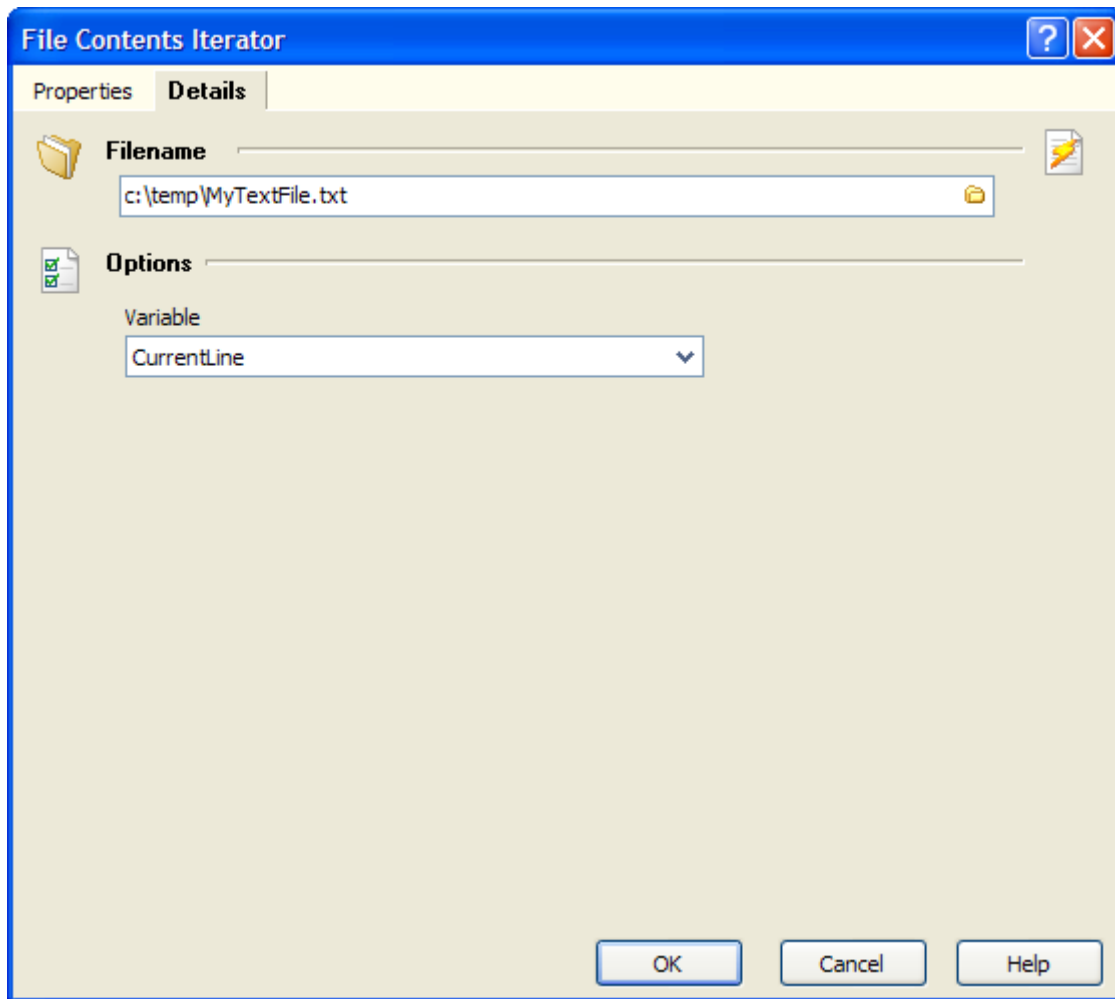
**Include Path** - If include path is specified, then the variable is set to the complete path to the folder. Eg. "c:\temp\myfolder", not specifying this option would give "myfolder"

### 5.3.4 File Contents Iterator

The File Contents Iterator action enables you to perform a group of actions for each line of text read from a file

An example might be to read a file containing a list of project files; for each project file execute the compiler action.

The actions performed for each line of the file must be child actions of the File Contents Iterator action.



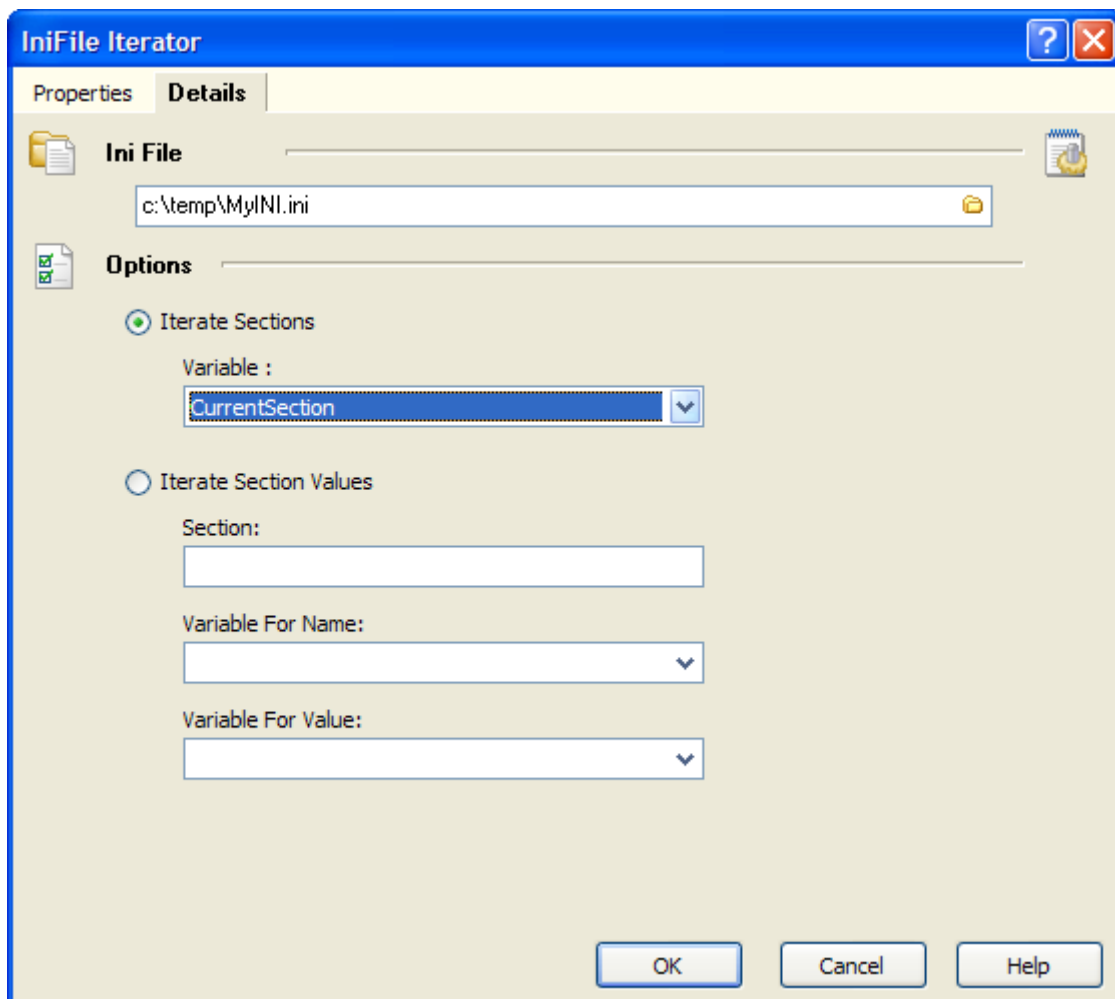
**Filename** - the name of the text file to read

**Variable** - the FinalBuilder variable to place the current line of the file in for each iteration

### 5.3.5 INI File Iterator

The IniFile Iterator action enables you to repeat a set of steps for each section or each value in a section of an INI file as part of your build process.

The action can work in two modes: Section Iterator, or Section Values Iterator. In Section Iterator mode, the child actions will be called for each section that exists in the specified INI file. In the Section Values mode, the child actions will be called for each Name=Value within the specified section of the INI file.



**Ini File** - Specify the INI file to read

**Iterate Sections** - Iterates for each section found in the INI file.

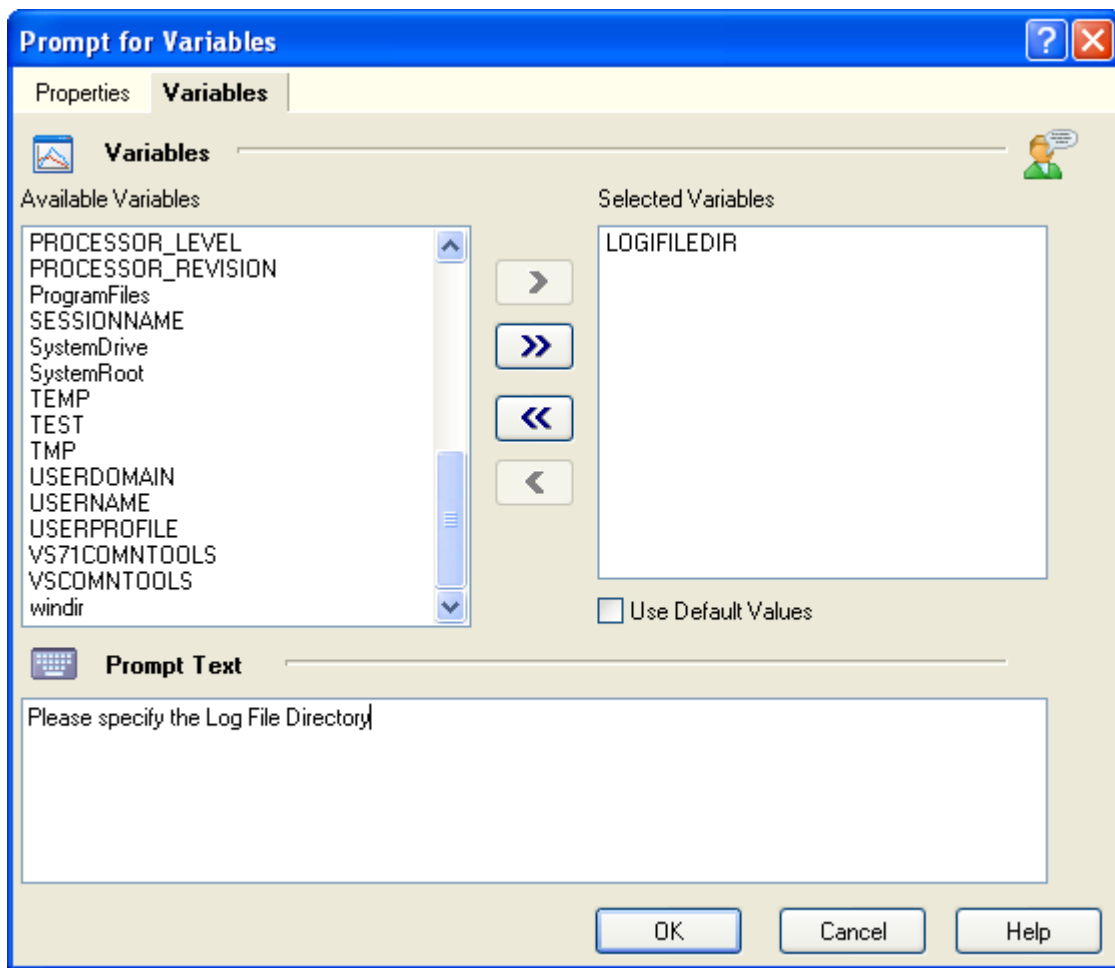
**Iterate Section Values** - You need to specify which section in the INI file, and then the action will iterate for each Name=Value pair found in the specified section

## 5.4 Interactive

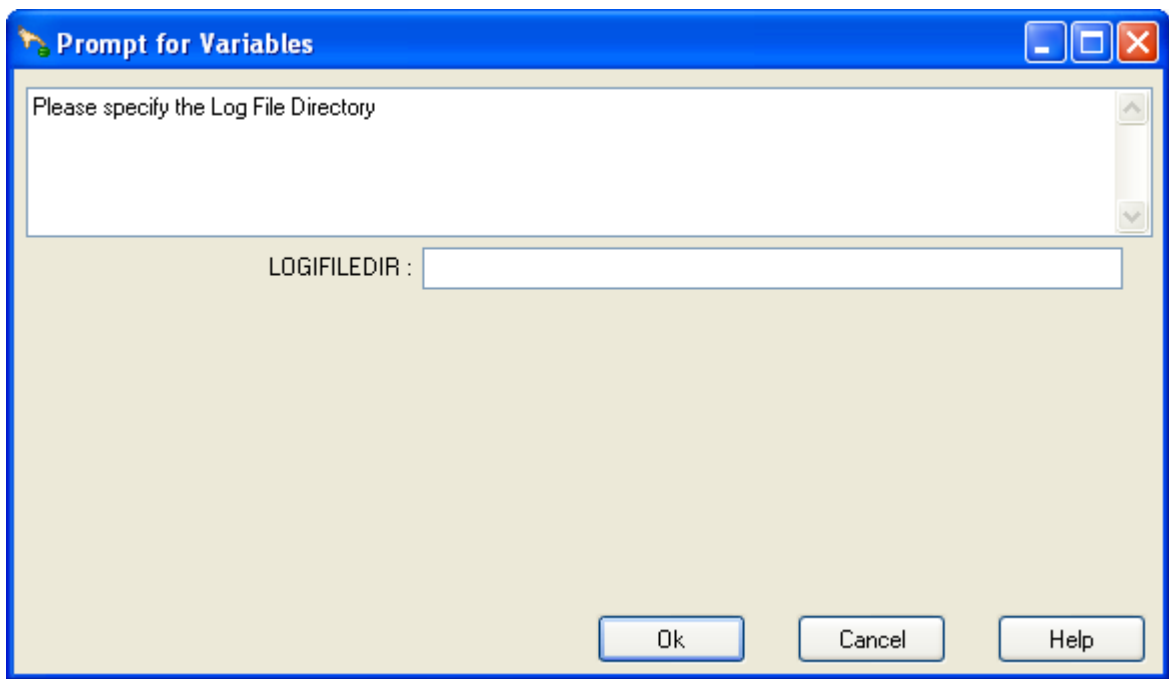
Interactive actions can only be run in interactive mode, ie. when the build is being run manually and not from the command line or a schedule.

### 5.4.1 Prompt for Variables Action

This action allows you to prompt the user for variable values at run time. Note that this action should not be used in scheduled builds as it will cause them to hang waiting for user input.

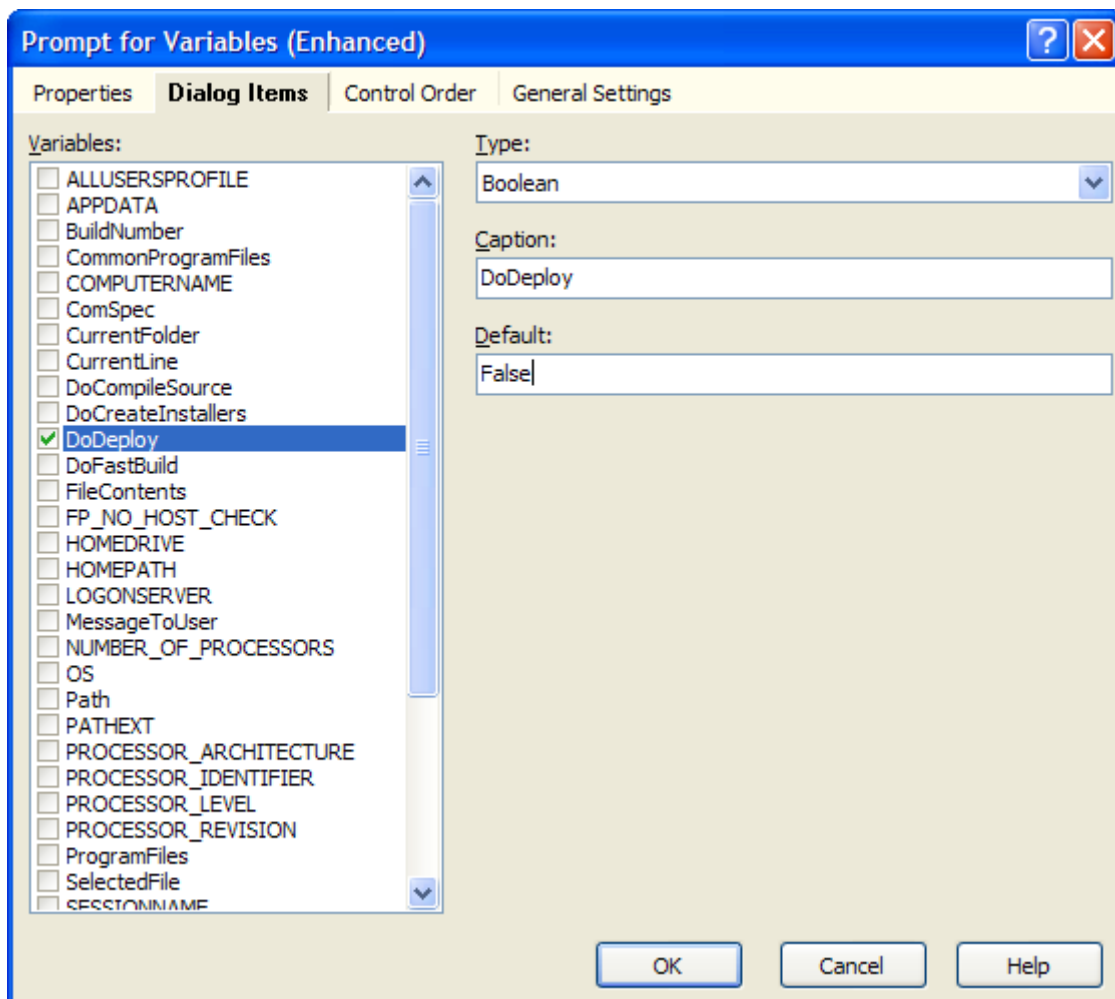


At runtime the above configuration would cause this form to be displayed when the action executes.

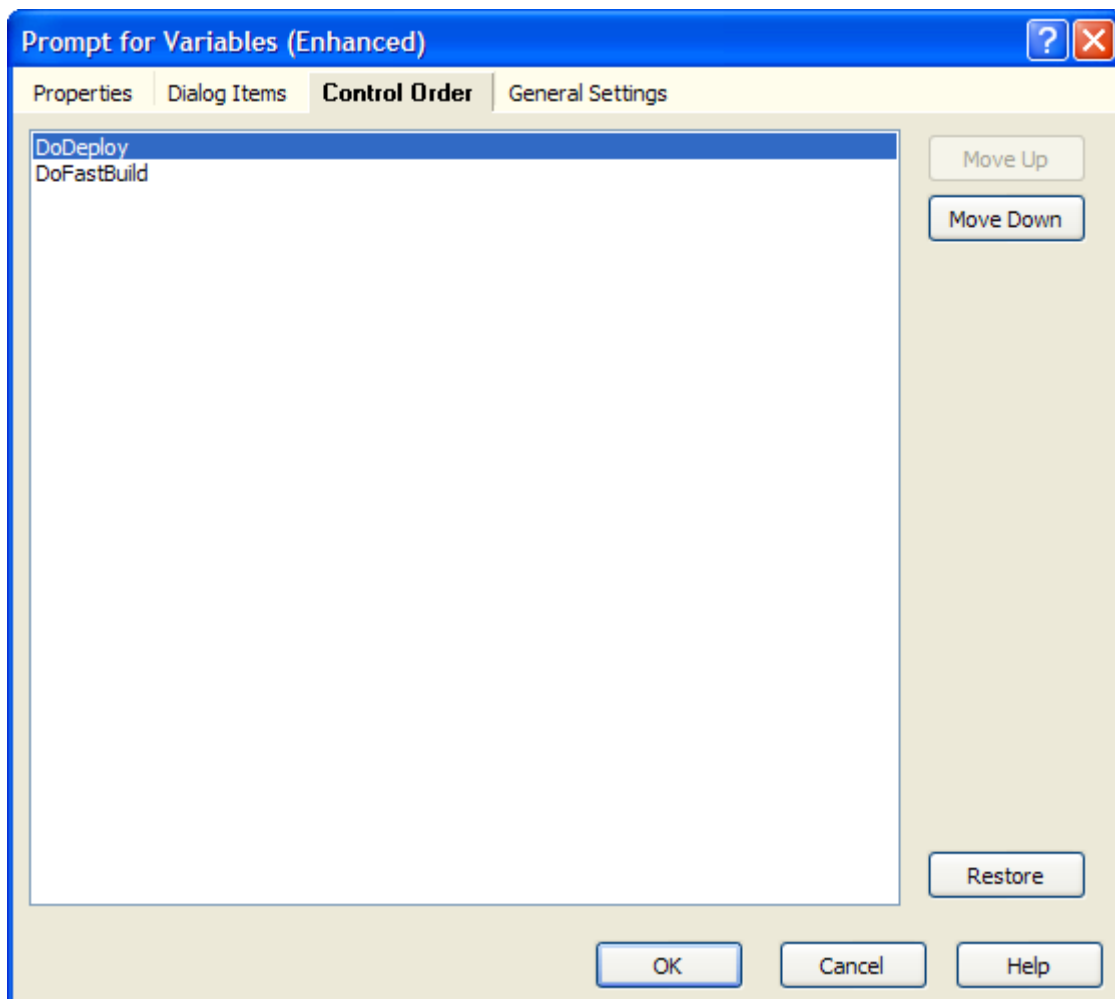


#### 5.4.2 Prompt for Variables Action (Enhanced)

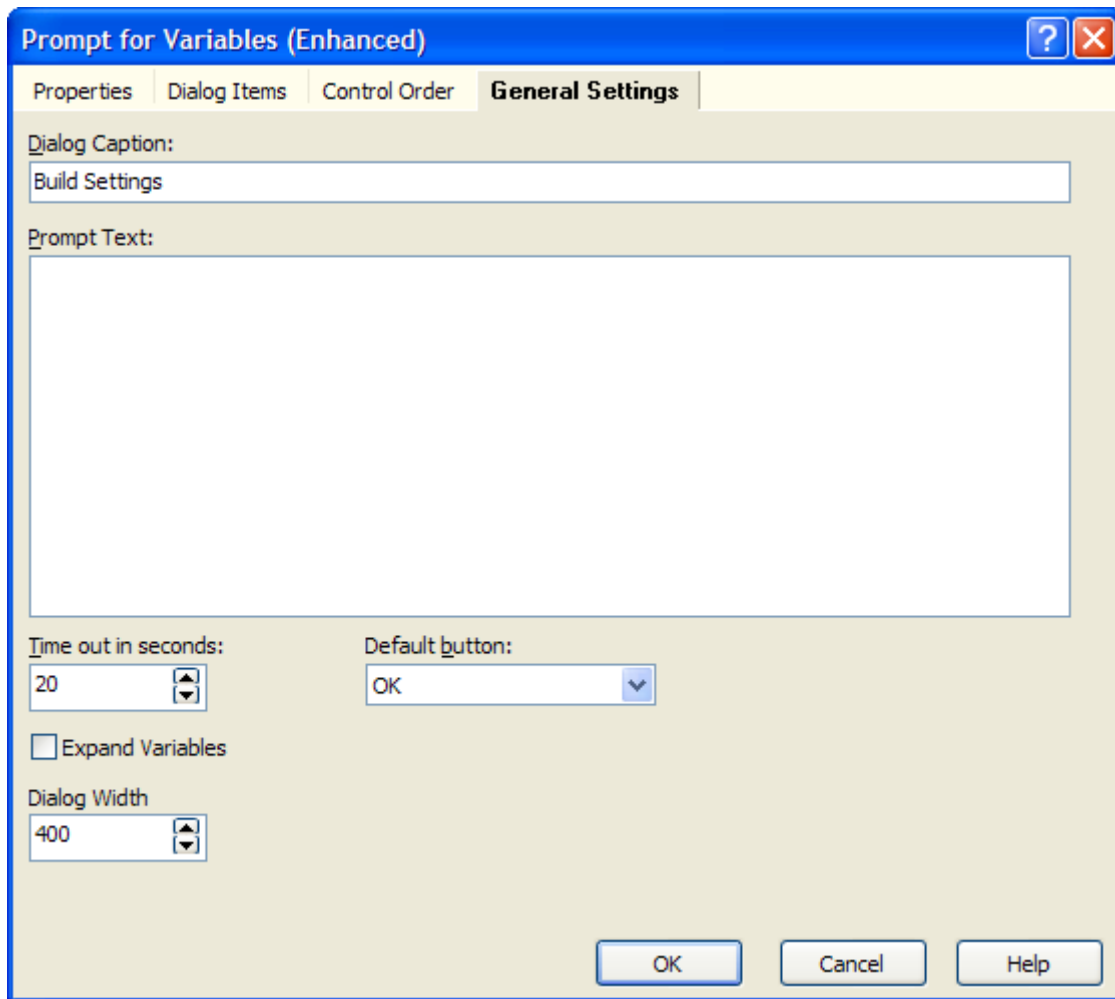
This action (kindly donated by Peter Thörnqvist) is an enhanced version of the Prompt for Variables action that allows you to specify the control types to be used at runtime in the prompt form.



The action allows you to set the order of the various values:



The general settings lets you further customise the dialog:

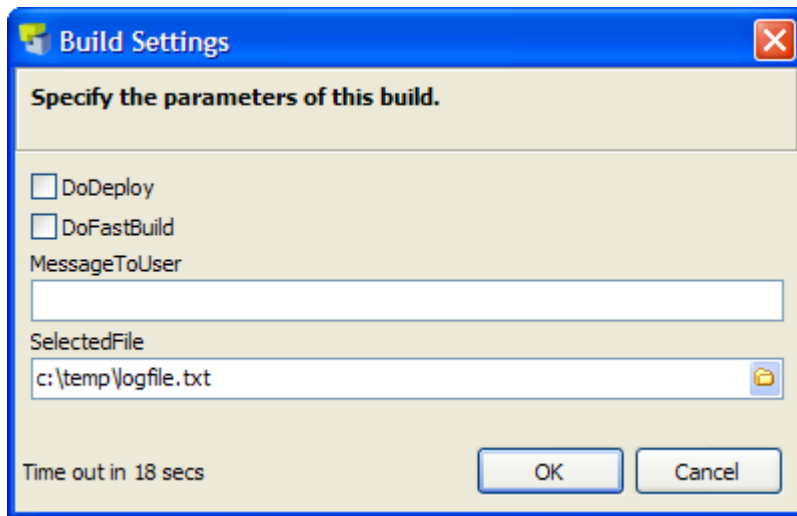


**Time Out In Seconds** - The dialog will automatically close if the user hasn't interacted with the dialog for the specified time

**Default Button** - This used for the timeout to choose a suitable action when the timeout is reached

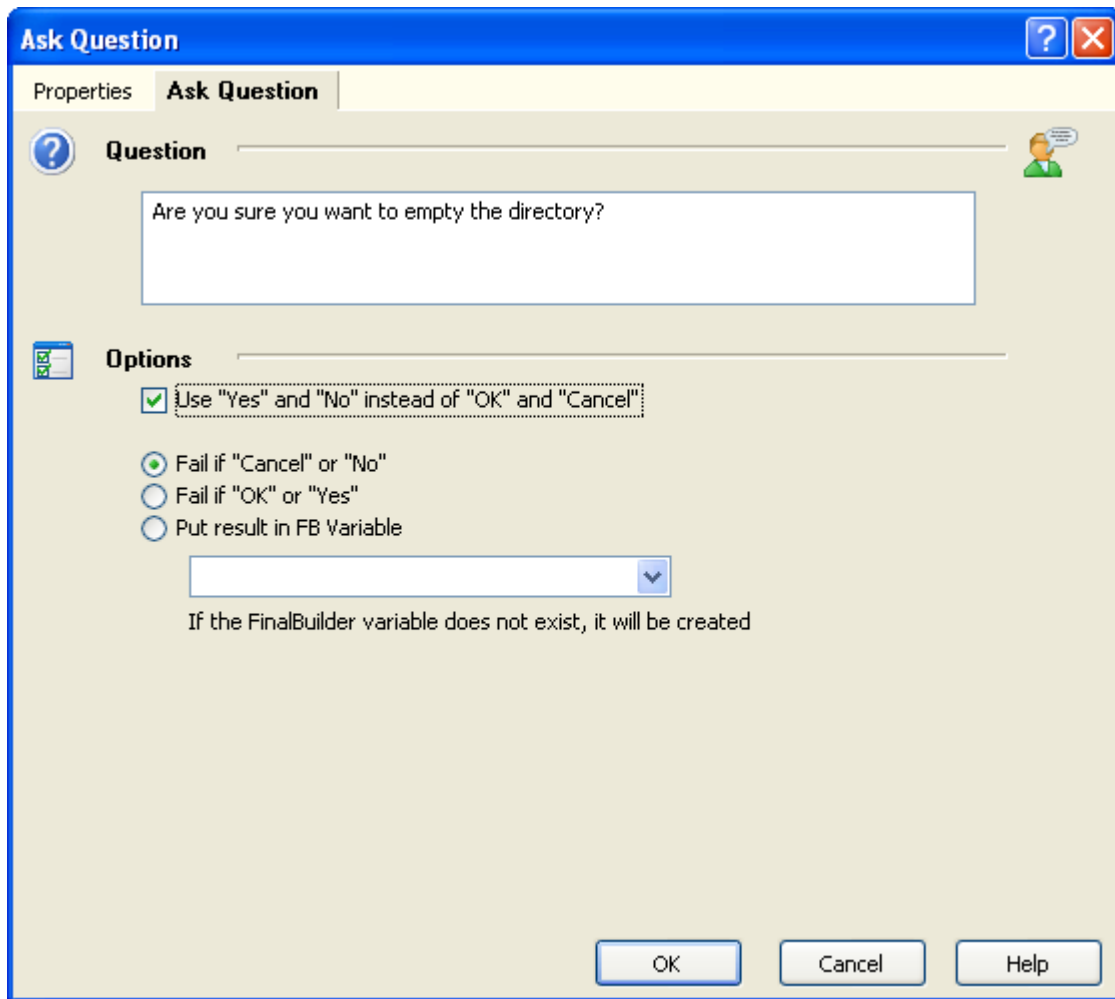
This is the prompt form at Runtime, showing 4 variables. Note that clicking on the Cancel button would cause the action to fail and stop the build.





### 5.4.3 Ask Question Action

This action allows you to prompt the person running the build process. You can optionally store the result in a FinalBuilder variable.



If you choose to store the result in a FinalBuilder variable, then OK / Yes will set the variable to True, and Cancel / No will set the variable to False

#### 5.4.4 Multi Question

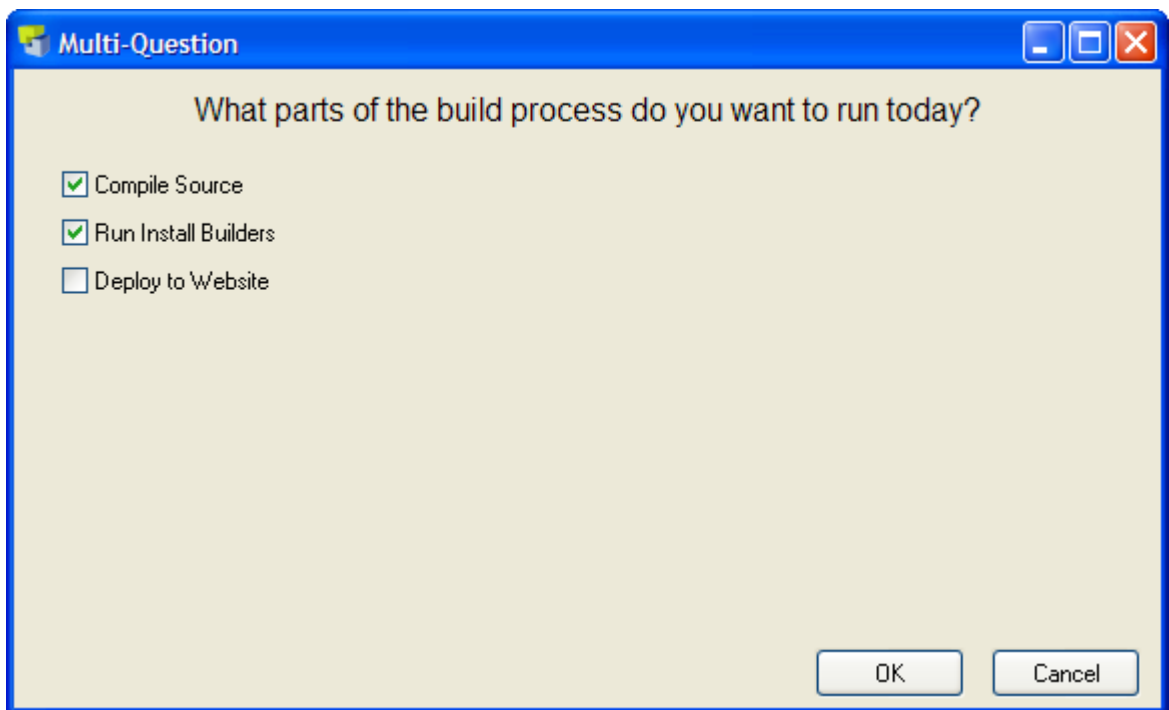
The MultiQuestion action enables you to interact with the user during your build process. Each answer can be either set to True or False which is then saved in FinalBuilder variables which can then control the flow of your build process.

| Answer Text          | Set Boolean Variable | Default                             |
|----------------------|----------------------|-------------------------------------|
| Compile Source       | DoCompileSource      | <input checked="" type="checkbox"/> |
| Run Install Builders | DoCreateInstallers   | <input checked="" type="checkbox"/> |
| Deploy to Website    | DoDeploy             | <input type="checkbox"/>            |
|                      |                      | <input type="checkbox"/>            |
|                      |                      | <input type="checkbox"/>            |
|                      |                      | <input type="checkbox"/>            |
|                      |                      | <input type="checkbox"/>            |
|                      |                      | <input type="checkbox"/>            |
|                      |                      | <input type="checkbox"/>            |
|                      |                      | <input type="checkbox"/>            |

For Example, the main question could be "What parts of the build process do you want to run today". Answer 1: "Compile Source", Answer 2: "Run Install Builders", Answer 3: "Deploy to Website". The boolean answers can then be used in the Condition property of other actions to control whether they are executed or not.

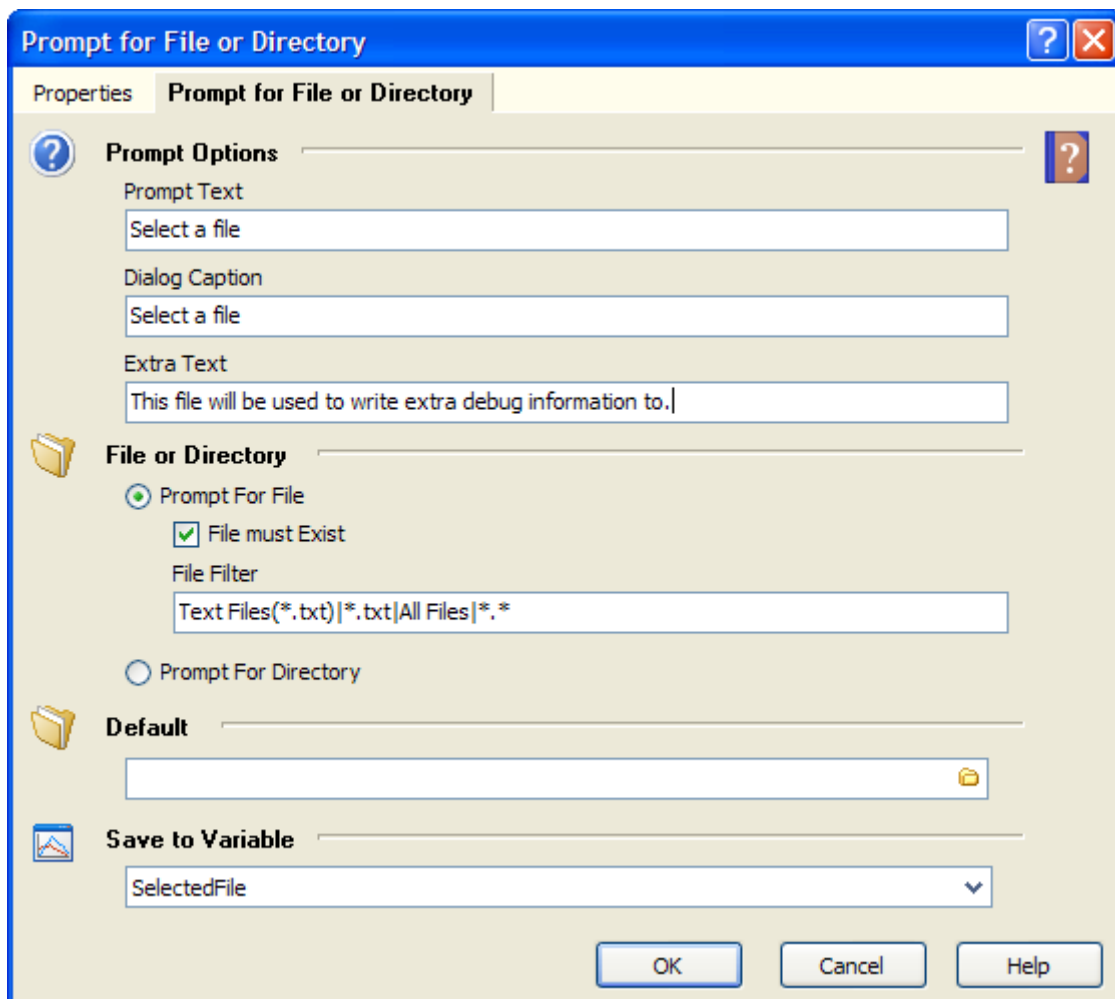
If you specify **Remember Last Used Settings** then the default value when the action runs will be set to the previously selected values when the action was last run.

This is what the action looks like at run time:

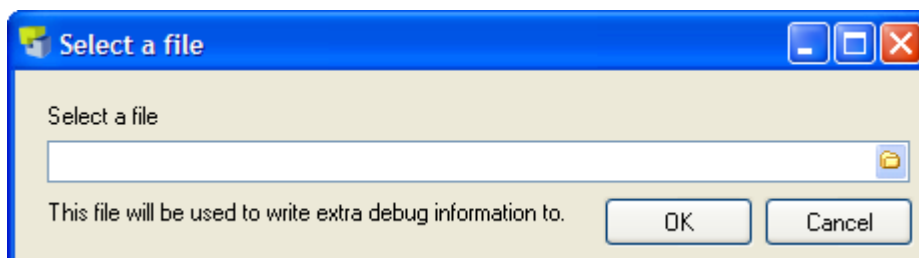


#### 5.4.5 Prompt for File or Directory

The Prompt for File or Directory action enables you to ask the user to specify a file or directory during your build process. The file or directory specified is saved in a variable so that it can be used in subsequent actions.



When the action is executed it looks something like this:



## 5.5 Misc Actions

### 5.5.1 Action Group

The Action Group action type has no specific functionality. It is provided as a handy action to use when grouping actions. This action's BeforeAction and AfterAction script events will execute as they do on any other action type.

### 5.5.2 Set Variable Action

The Set Variable Action provides a means to set the value of a Project Variable or a User Variable without resorting to writing script. The value can include other variables. To append to the existing variable, simply prefix the new value with %VARIABLENAME% where variablename is the name of the variable whose value you are setting.

The Modifier and Apply to Existing Value options are new in FinalBuilder 2. A modifier is a function that can be applied to the new value After any variables have been replaced. The available modifiers are :

None - the default.

Trim

TrimLeft

TrimRight

IncludeTrailingBackSlash

ExcludeTrailingBackSlash

ExtractFileName,

ExtractFilePath

ExtractFileDrive

ExtractFileExt

ShortFileName

AddQuotes

AddDoubleQuotes

StripQuotes

Increment

Decrement

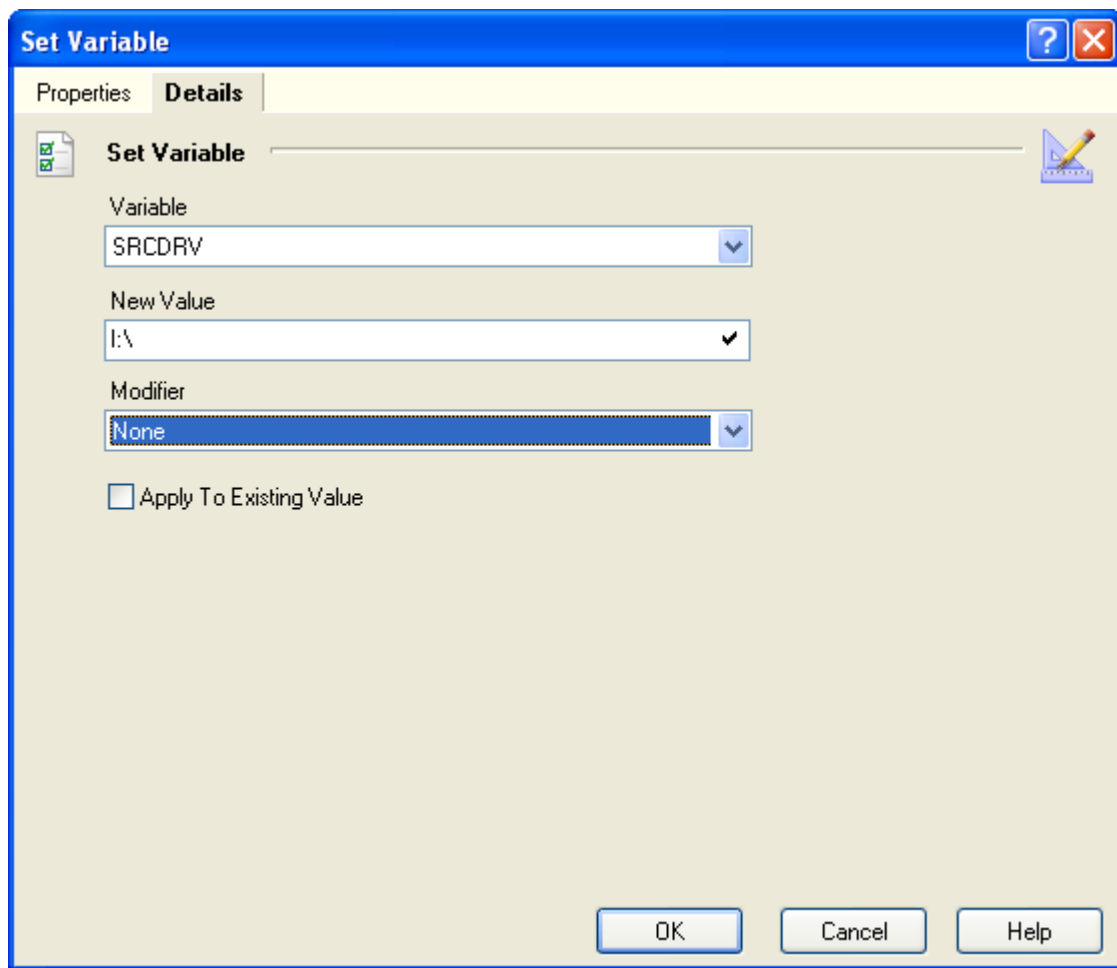
LowerCase

UpperCase

Encrypt - Encrypts using blowfish with a hard wired key

Decrypt - Decrypts values previously encrypted with the Encrypt modifier.

If Apply to Existing Value is checked then the New Value field is ignored and the modifier is applied to the existing variable value.



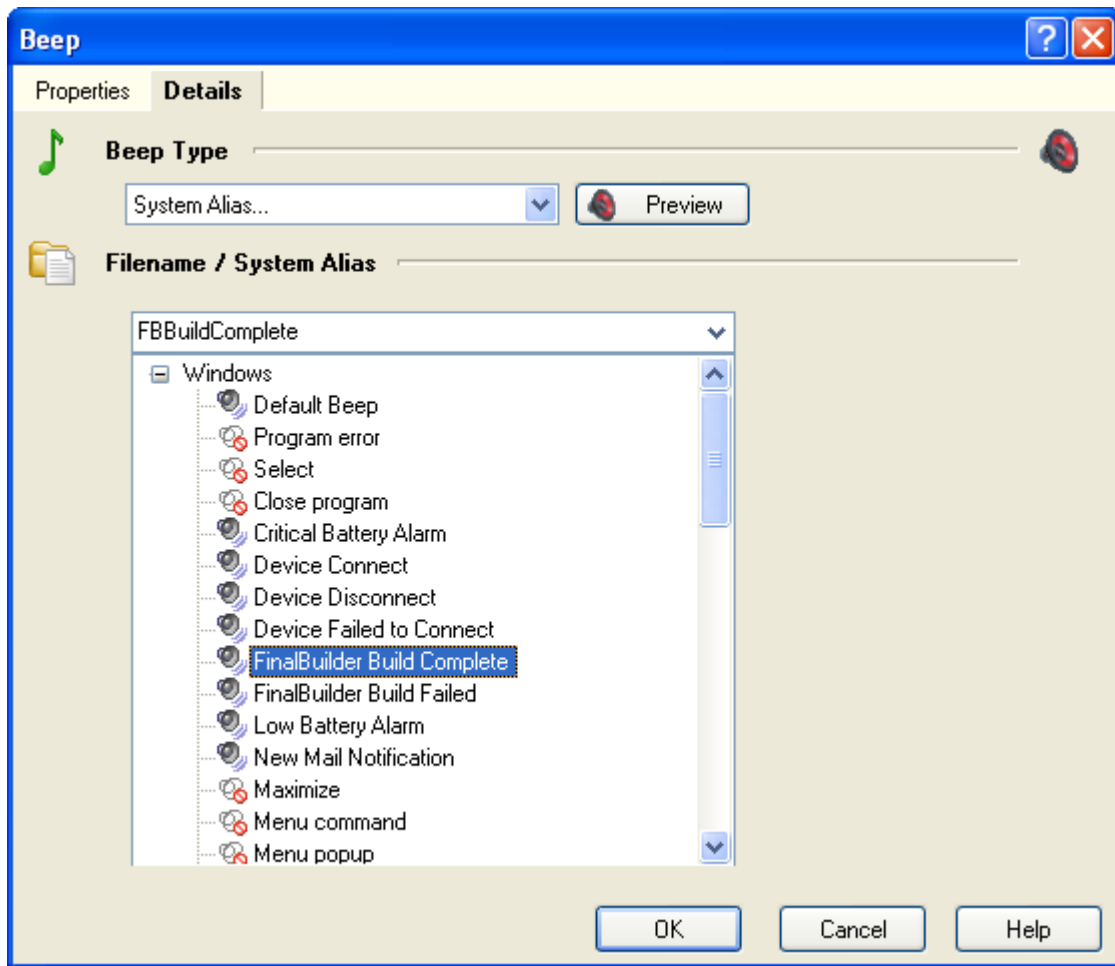
### Scripting Info

The Action properties available are :

- property** NewValue : WideString;
- property** VariableName : WideString;
- property** Modifier : TFBSetVariableModifier;
- property** ApplyToExisting : boolean;

### 5.5.3 Beep Action

The Beep Action will play a sound, either one of the pre defined system sounds, or a Wave File, or a System Sound Alias. FinalBuilder creates two System Aliases, Build Complete and Build Error. You can change the sounds assigned to these aliases in the windows control panel.

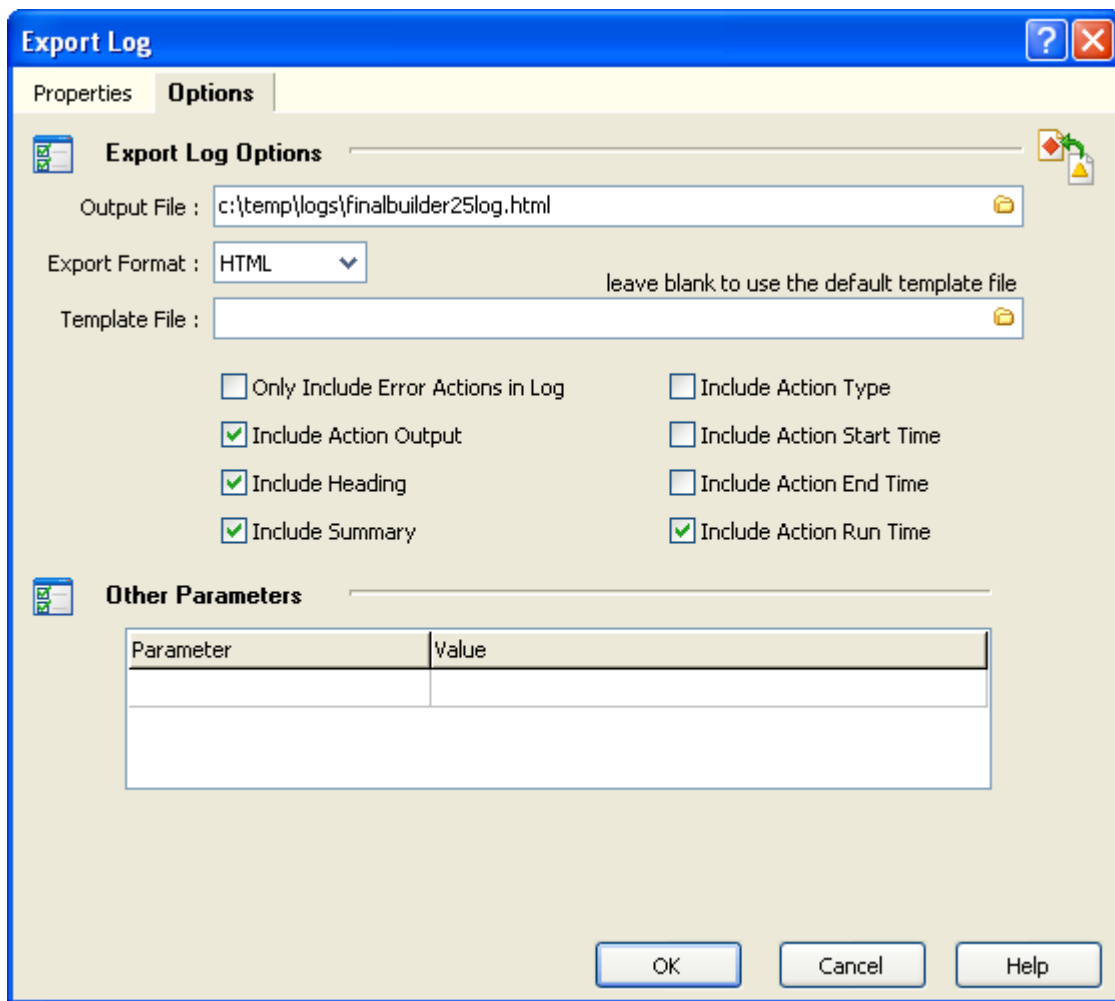


#### 5.5.4 Export Log Action

This action allows you to export the FinalBuilder log file as part of your build process. This enables the log to be sent as an attachment via email etc.

You can choose to export the current build, in Text, HTML or XML formats.



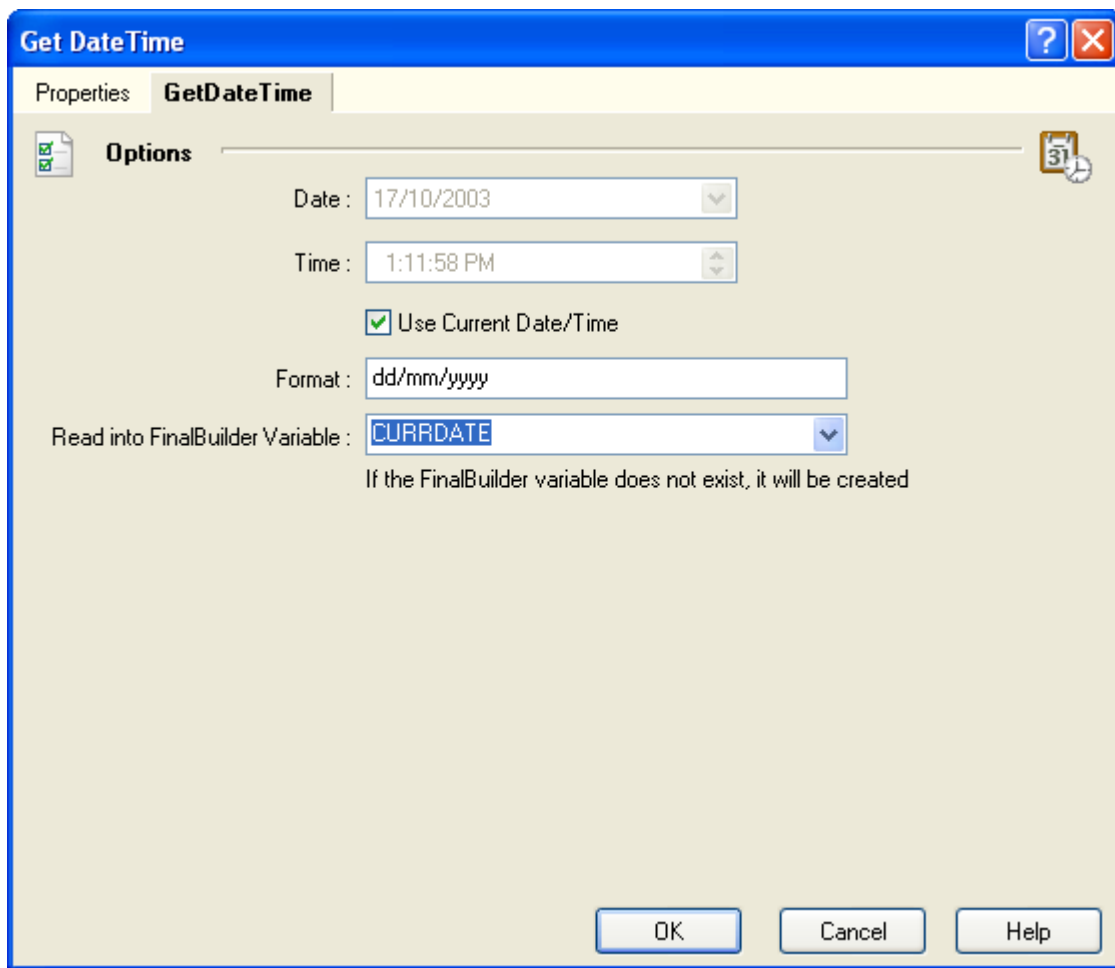


You can also specify a different XSL Stylesheet template file to alter the output format of the log file. The "Other Parameters" allow you to pass variables to your XSL Stylesheet.

Not that the default options for the log can be set in the FinalBuilder options dialog.

### 5.5.5 Get DateTime Action

This action gets a Date or Date & Time into a FinalBuilder Variable.



Valid date format specifiers for the Format property :

#### **Specifier      Displays**

- c      Displays the date using the format given by the ShortDateFormat global variable, followed by the time using the format given by the LongTimeFormat global variable. The time is not displayed if the date-time value indicates midnight precisely.
- d      Displays the day as a number without a leading zero (1-31).
- dd     Displays the day as a number with a leading zero (01-31).
- ddd    Displays the day as an abbreviation (Sun-Sat) using the strings given by the ShortDayNames global variable.
- dddd   Displays the day as a full name (Sunday-Saturday) using the strings given by the LongDayNames global variable.
- dddddd Displays the date using the format given by the ShortDateFormat global variable.
- dddddd      Displays the date using the format given by the LongDateFormat global variable.
- e      (Windows only) Displays the year in the current period/era as a number without a leading zero (Japanese, Korean and Taiwanese locales only).
- ee     (Windows only) Displays the year in the current period/era as a number with a leading zero (Japanese, Korean and Taiwanese locales only).
- g      (Windows only) Displays the period/era as an abbreviation (Japanese and

Taiwanese locales only).

gg (Windows only) Displays the period/era as a full name. (Japanese and Taiwanese locales only).

m Displays the month as a number without a leading zero (1-12). If the m specifier immediately follows an h or hh specifier, the minute rather than the month is displayed.

mm Displays the month as a number with a leading zero (01-12). If the mm specifier immediately follows an h or hh specifier, the minute rather than the month is displayed.

mmm Displays the month as an abbreviation (Jan-Dec) using the strings given by the ShortMonthNames global variable.

mmmm Displays the month as a full name (January-December) using the strings given by the LongMonthNames global variable.

yy Displays the year as a two-digit number (00-99).

yyyy Displays the year as a four-digit number (0000-9999).

h Displays the hour without a leading zero (0-23).

hh Displays the hour with a leading zero (00-23).

n Displays the minute without a leading zero (0-59).

nn Displays the minute with a leading zero (00-59).

s Displays the second without a leading zero (0-59).

ss Displays the second with a leading zero (00-59).

z Displays the millisecond without a leading zero (0-999).

zzz Displays the millisecond with a leading zero (000-999).

t Displays the time using the format given by the ShortTimeFormat global variable.

tt Displays the time using the format given by the LongTimeFormat global variable.

am/pm Uses the 12-hour clock for the preceding h or hh specifier, and displays 'am' for any hour before noon, and 'pm' for any hour after noon. The am/pm specifier can use lower, upper, or mixed case, and the result is displayed accordingly.

a/p Uses the 12-hour clock for the preceding h or hh specifier, and displays 'a' for any hour before noon, and 'p' for any hour after noon. The a/p specifier can use lower, upper, or mixed case, and the result is displayed accordingly.

ampm Uses the 12-hour clock for the preceding h or hh specifier, and displays the contents of the TimeAMString global variable for any hour before noon, and the contents of the TimePMString global variable for any hour after noon.

/ Displays the date separator character given by the DateSeparator global variable.

: Displays the time separator character given by the TimeSeparator global variable.

'xx'/"xx" Characters enclosed in single or double quotes are displayed as-is, and do not affect formatting.

### 5.5.6 Comment Action

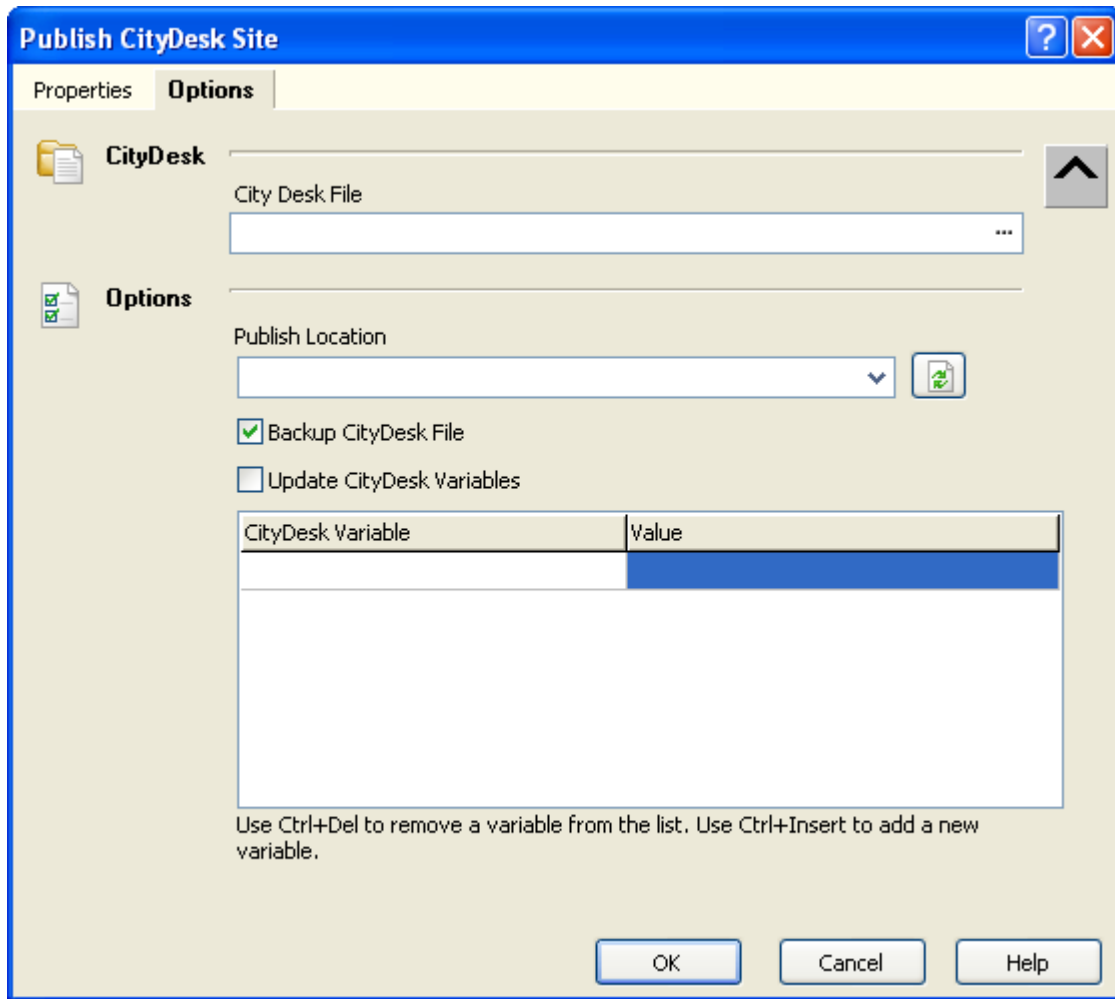
Comment Actions are just a way to place text inside the action list, they do not get executed at run time and do not have script events.

### 5.5.7 Run Script Action

The Run Script Action is a simple action that has an extra script event, OnExecute where you can place active script code.

### 5.5.8 CityDesk Action

This Action will publish a [CityDesk 2](#) Site to a predefined location. Before using this action for the first time you need to set the CityDesk location option in the FinalBuilder Options dialog.

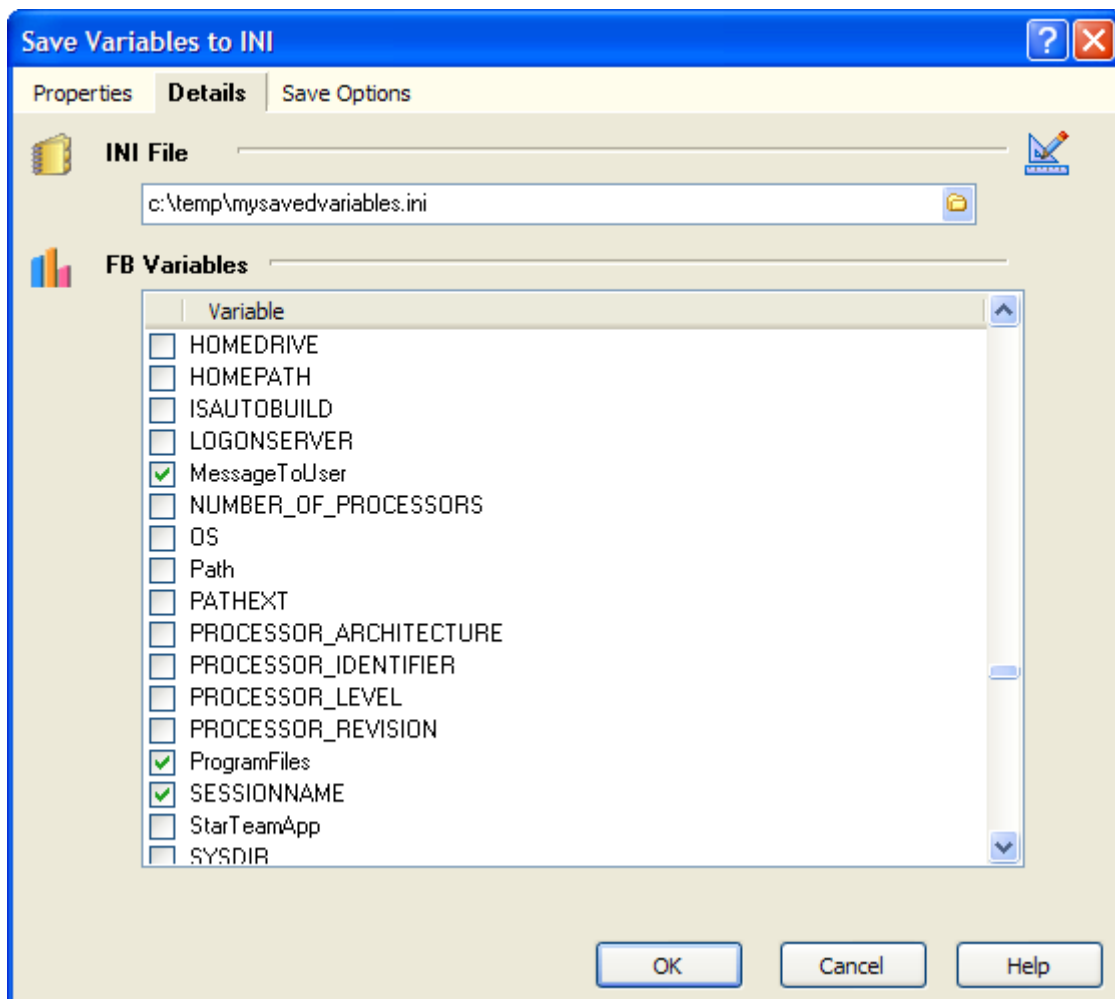


When you select a CityDesk file, FinalBuilder will read the defined publish locations into the combo box. You can however type in this combo. Note however that when the action runs this field must have a valid location, otherwise CityDesk will display a message box with an error (which would cause unattended build to hang!).

**Note :** CityDesk 2 Files are access databases, and this action uses the Microsoft Jet 4.x OLEDB provider. This provided is not installed in windows by default. If you have Office XP/2003 or Access XP/2003 you should already have it, otherwise you can download this from the Microsoft web site.

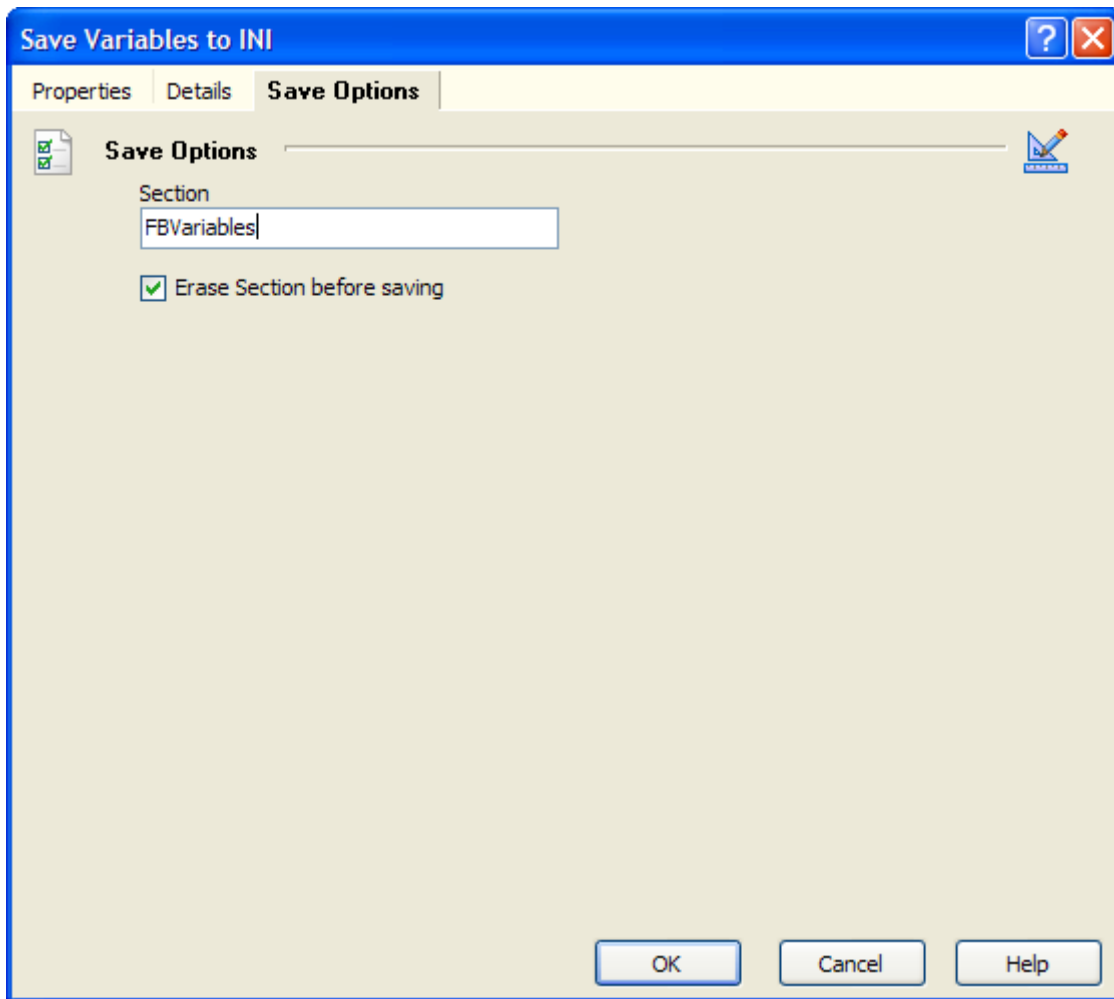
### 5.5.9 SaveVariablesToIni

The Save Variables from INI file action enables you to save a set of FinalBuilder variables to an INI file.



**INI File** - Specify which INI file the variables will be written to

**FB Variables** - Decide which variables will be written to the file. The variables will be written in the form: <variable name>=<variable value>

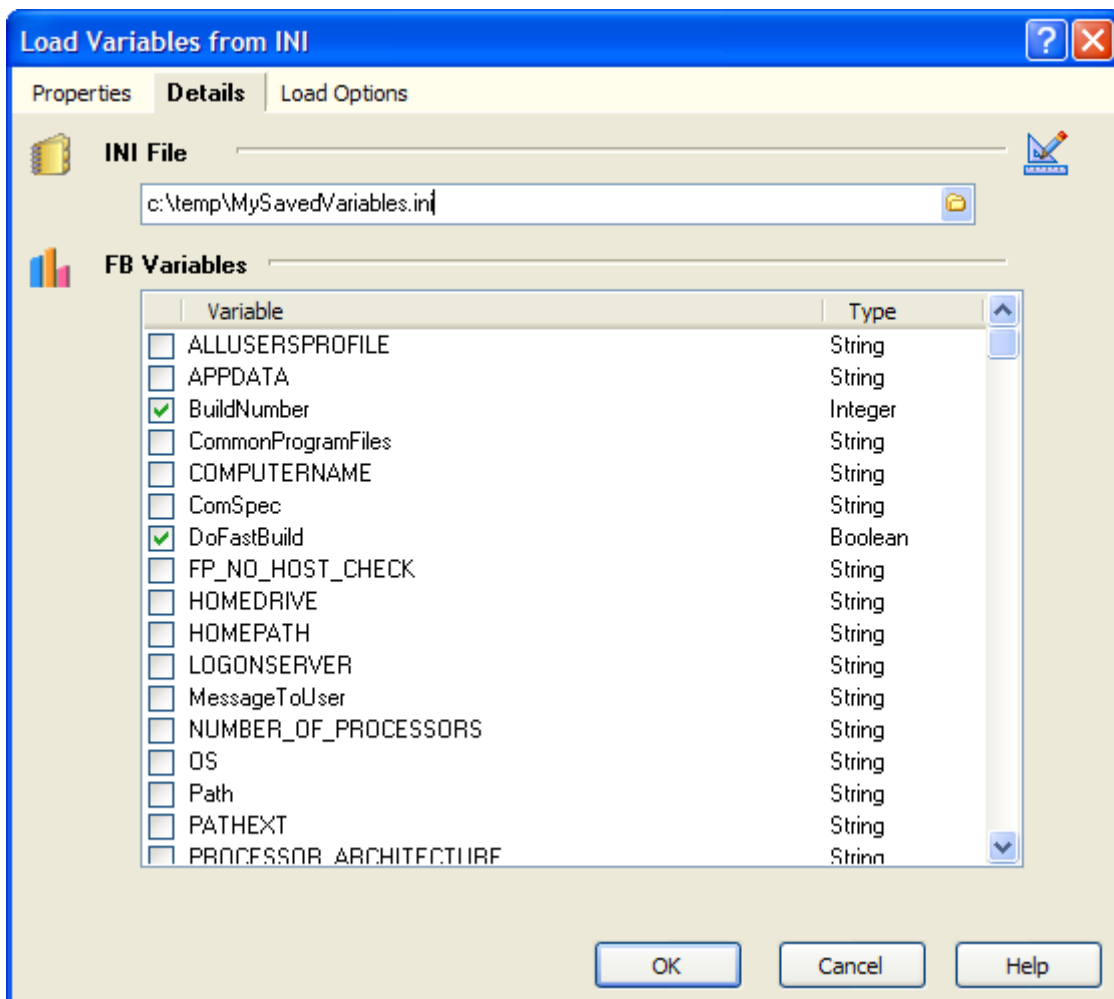


**Section** - Specify the section in the INI file to write the variables to

**Erase Section before saving** - this will erase only the Section part of the INI file. It will clear out any other variables which had been previously written to the INI

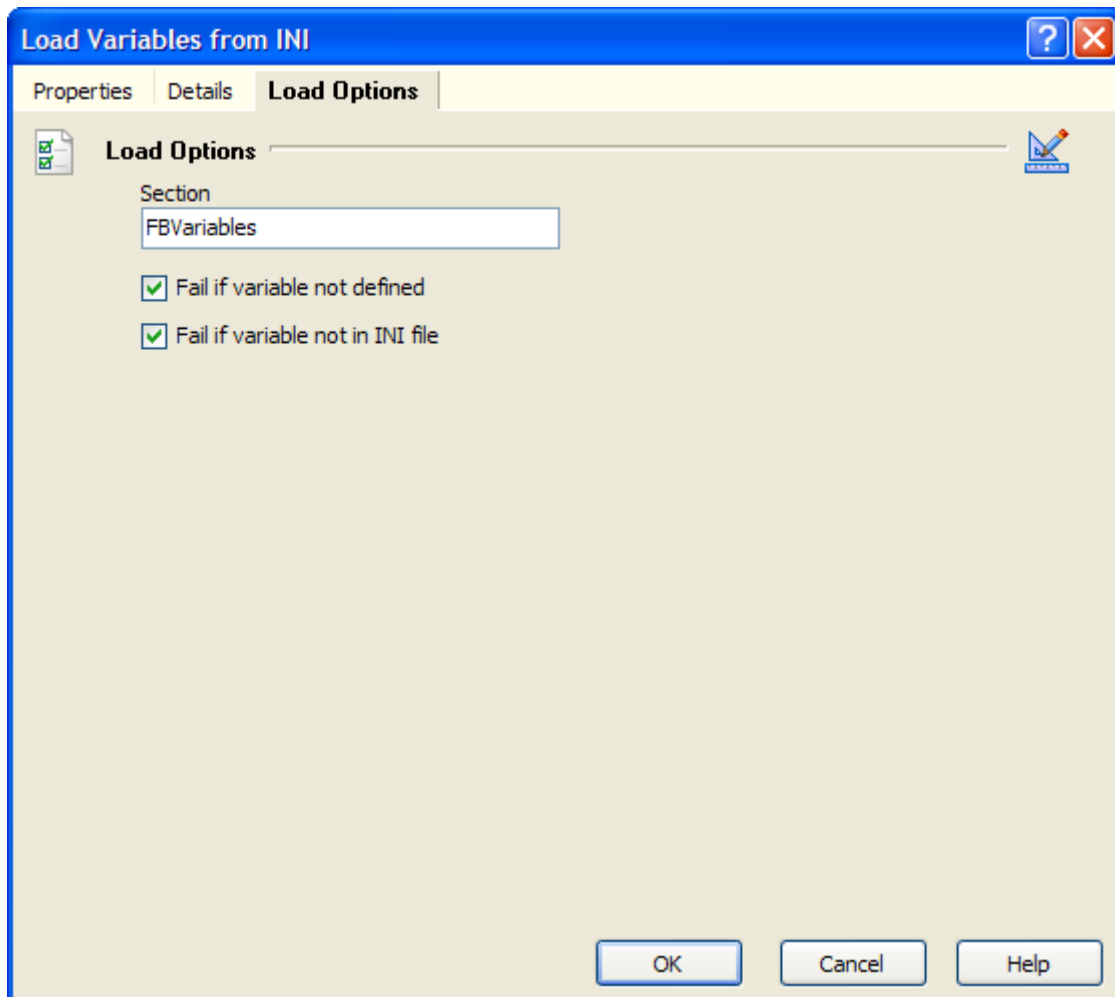
#### 5.5.10 LoadVariablesFromIni

The Load Variables from INI file action enables you to set FinalBuilder variables to the values specified in an INI file.



**INI File** - Specify the INI file which contains the variables

**FB Variables** - Specify which variables should be attempted to be set by values in the INI file. You can force the type of the variable to either String, Boolean, or Integer.



**Section** - specify the section in the INI where the variables are located.

Variables must be specified in the following way:  
<variableName>=<Variable value>

eg.  
[FBVariables]  
BuildNumber=10

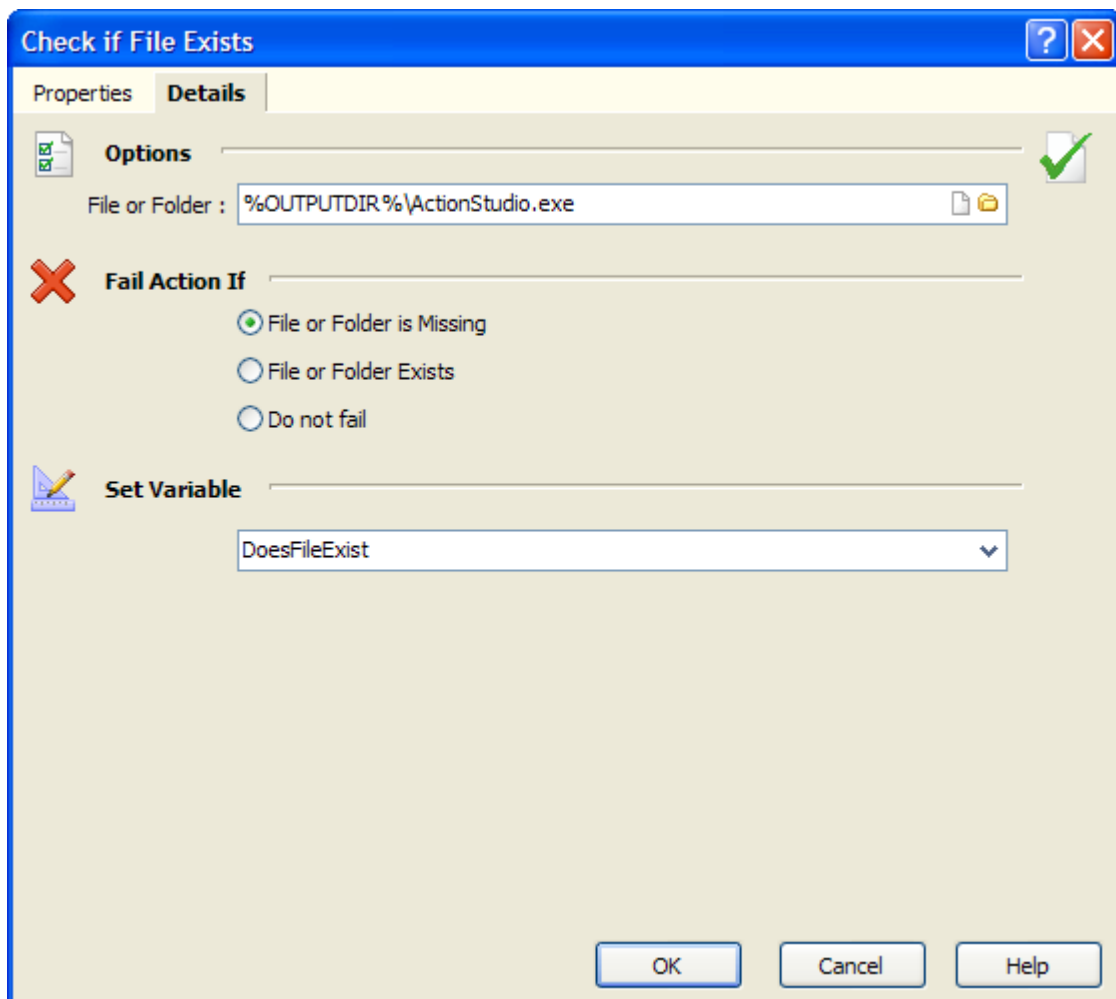
If you want to use boolean values, then False is 0, True is any other integer value.

## 5.6 Files & Directories

### 5.6.1 Check File Exists Actoin

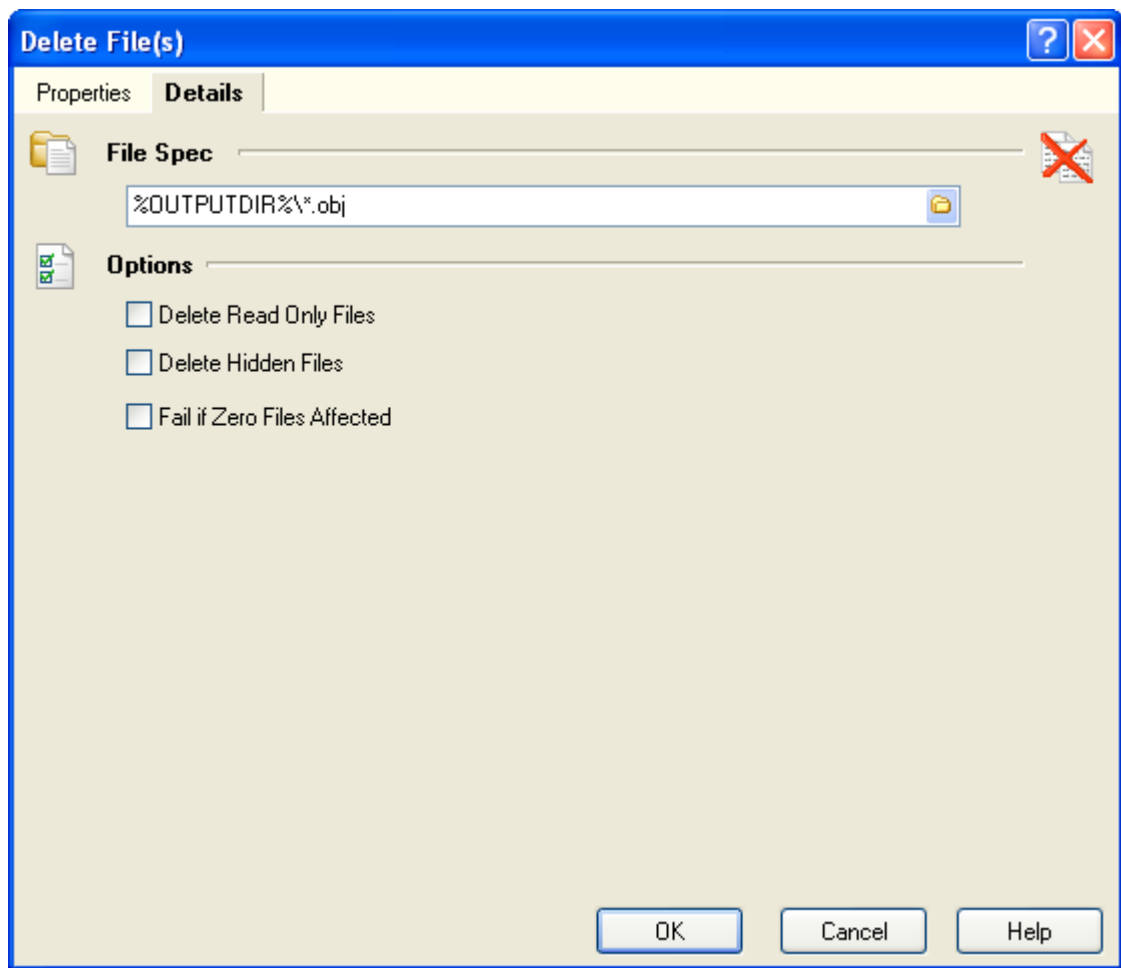
This action allows you to check for the existence of a particular file or folder. You can choose to fail the action if the file or folder exists, or if the file or folder is missing. You can also choose not to fail the action at all; in this case you would normally set a variable so that you can use the condition in a subsequent action (for example an If...Then action).





### 5.6.2 Delete File(s) Action

The Delete files Action does just that, delete files. You can use wildcards (? or \*) in the file name. Note that this action does not provide an option to recurse through directories. This is intentional, to avoid accidents. If you need this functionality, use the Run DOS Command Action.



### Scripting Info

The Action properties available are :

**property** FileSpec : WideString; // the file specification for the files to delete. You may use Wildcards.

**property** FilesAffected : integer ; // read only, the number of files deleted

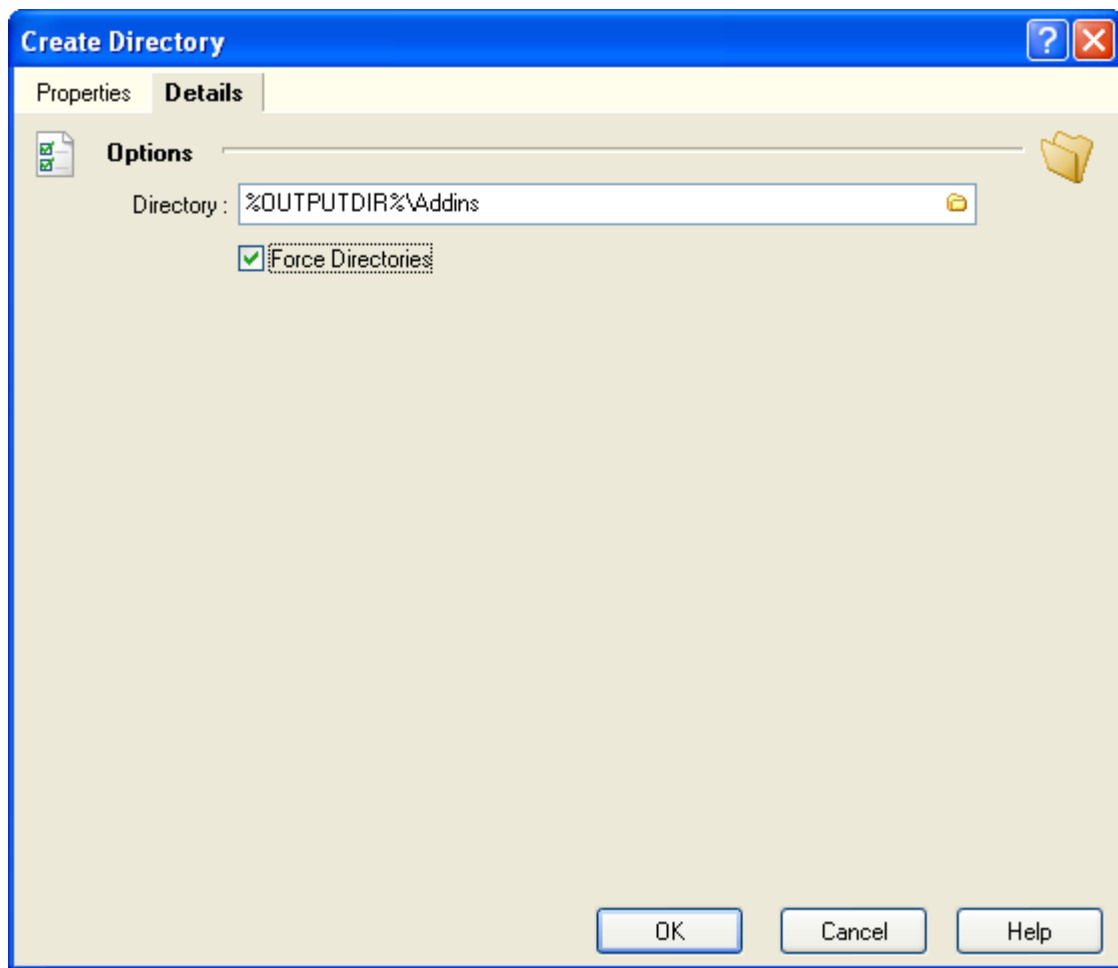
**property** FailIfNoFile : WordBool; // fail if no files affected.

**property** DeleteReadOnly : WordBool; // delete read only files.

**property** DeleteHidden : WordBool; //delete hidden files.

### 5.6.3 Create Directory Action

This action will create the specified Directory. DOS and Windows only allow directories to be created one at a time. For example, to create the C:\APPS\SALES\LOCAL directory, the APPS and SALES directories must exist before the LOCAL directory can be created. Use the Force Directories to create a directory and all parent directories that do not already exist.



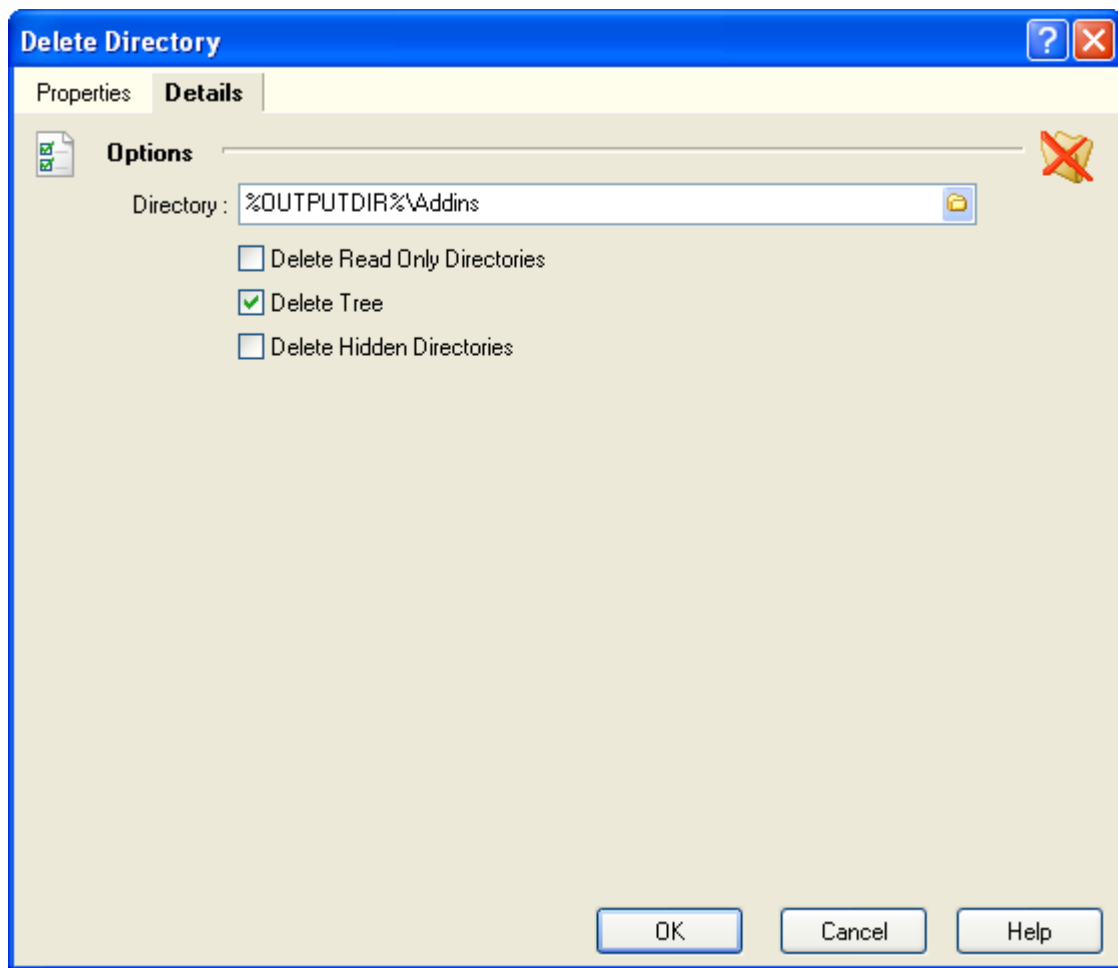
### Scripting Info

The Action properties available are :

**property** FileOrDirectory : WideString; // The Directory to Create  
**property** Force : WordBool; // Force the creation of directories.

#### 5.6.4 Delete Directory Action

This action will delete the specified directory. If Delete Tree is set then the action will delete any files in the target directory, if not then the action will fail if the directory is not empty.



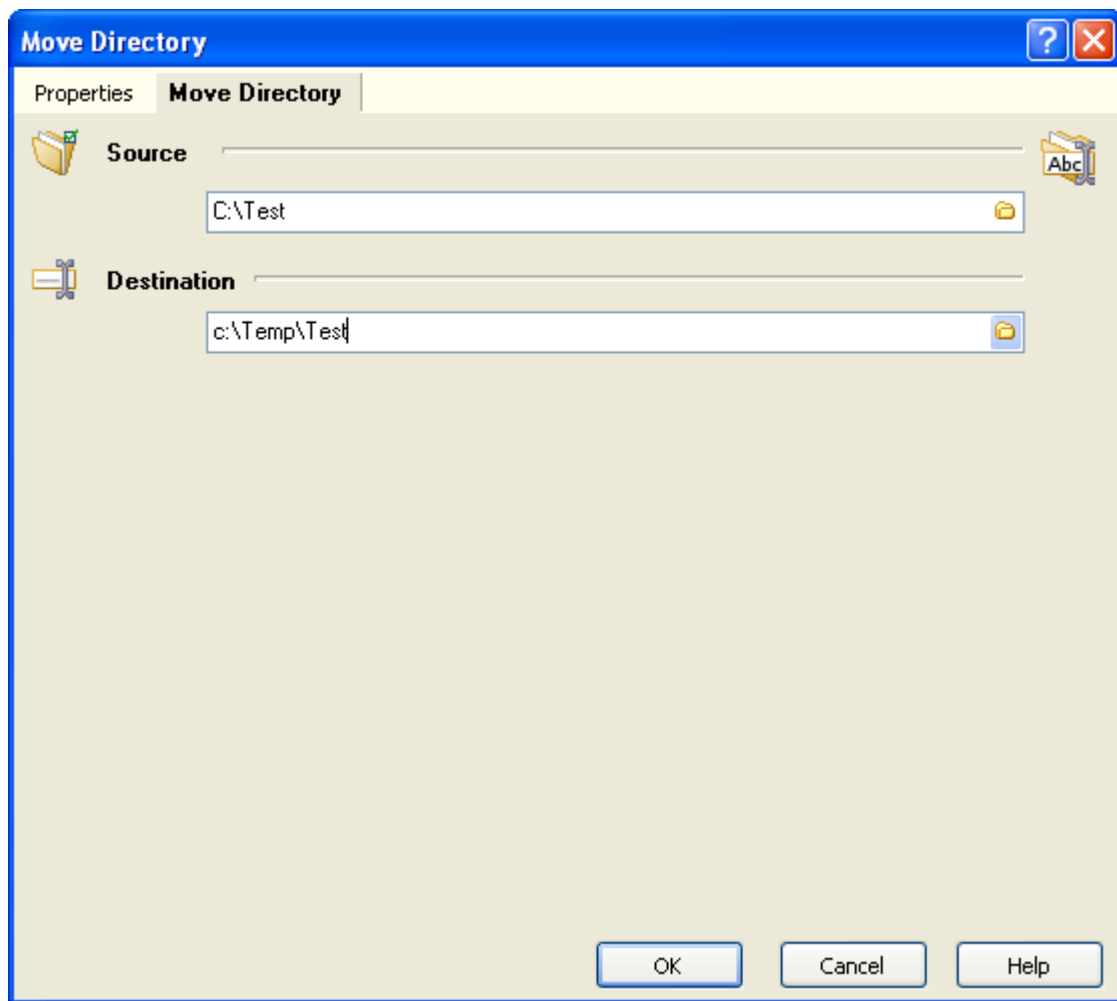
### Scripting Info

The Action properties available are :

**property** FileOrDirectory : WideString;// The Directory to Delete  
**property** DeleteReadOnly : WordBool;  
**property** DeleteHidden : WordBool;  
**property** DeleteTree : WordBool;

### 5.6.5 Move Directory Action

This action will move the specified Directory to the Destination Directory. Note that the Destination directory is the new name for the folder, so for example if you want to move c:\test to c:\temp\test then you should specify c:\temp\test as the destination directory.



### Scripting Info

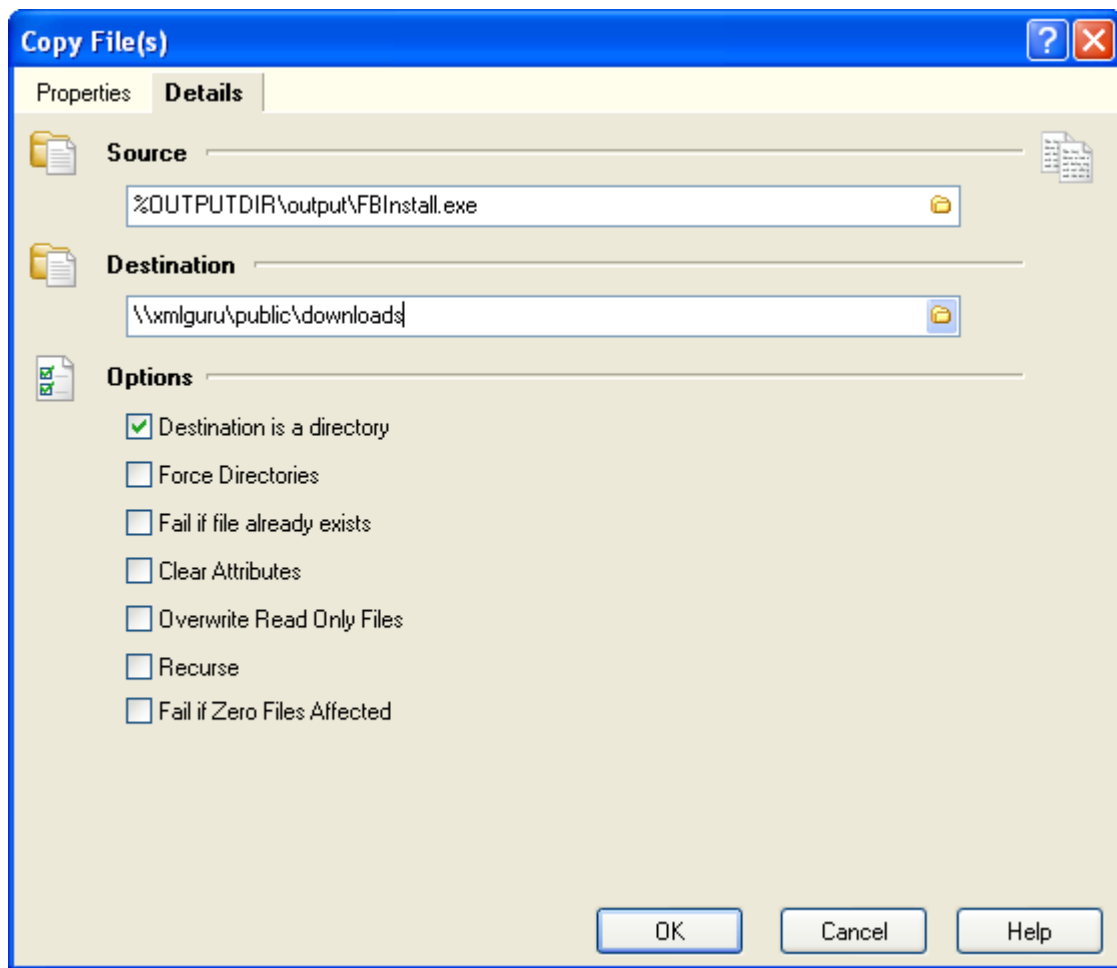
The Action properties available are :

**property** Directory :String; // the directory you wish to move.

**property** DestinationDirectory : String ; // the new path for the directory.

### 5.6.6 Copy File(s) Action

This action will copy a File or Files to the destination file or directory. You can use wildcards in the source setting to copy more than one file, in the Destination is a directory will be set automatically. To copy files from subdirectories as well, check the Recurse property. If the target directories do not already exist, check the Force Directories property to make FinalBuilder create the directories as needed.



### Scripting Info

The Action properties available are :

**property** FileSpec :WideString; // the file specification for the files to copy. You may use Wildcards.

**property** FilesAffected : integer ; // read only, the number of files Copied

**property** Target :WideString; // The target file or directory

**property** Force : WordBool; //Use Force Directories to create a directory and all parent directories that do not already exist.

**property** TargetIsDir : WordBool; // Specifies that the Target Is a Directory

**property** FailIfExists : WordBool; // Fail if the target file Exists

**property** OverwriteReadOnly : WordBool; //Overwrite Read Only Files

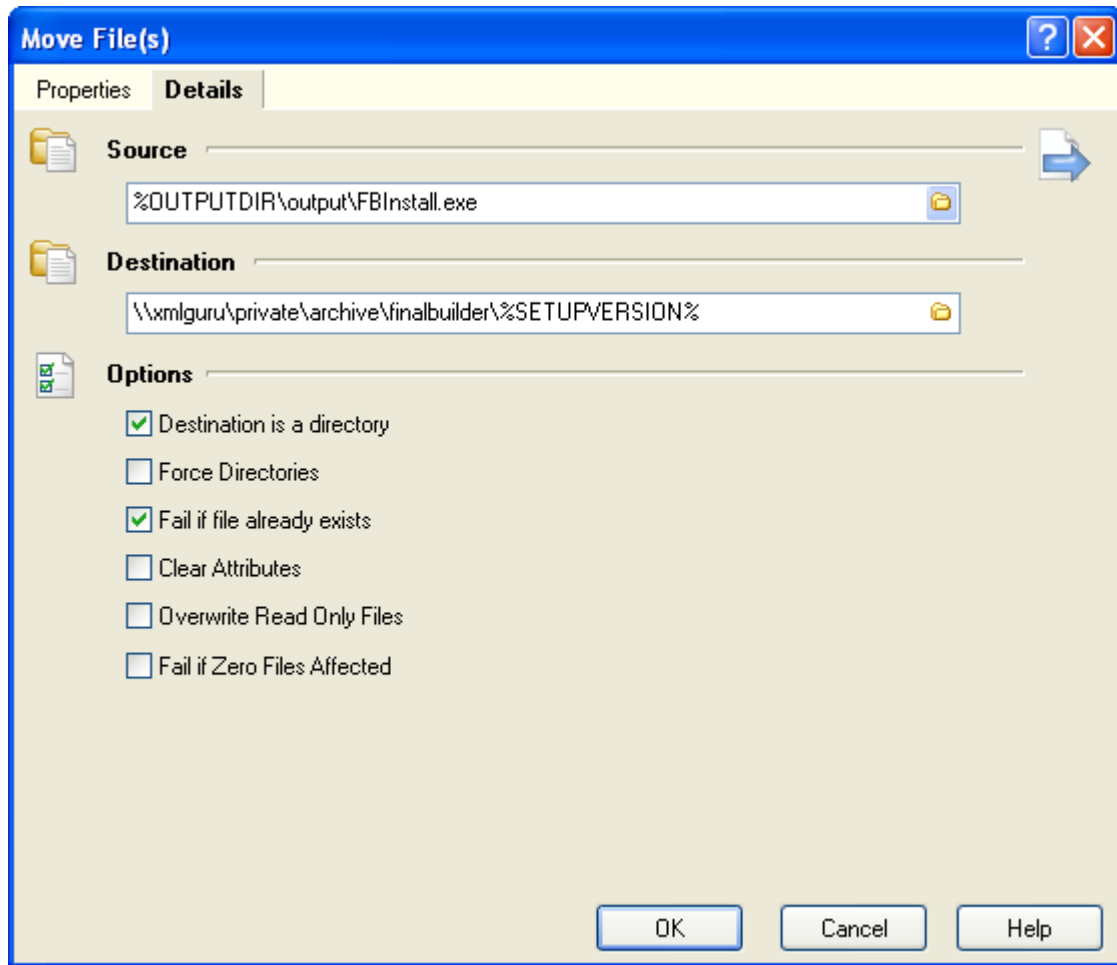
**property** Recurse : WordBool; // Recurse subdirectories, only valid if the target is a directory

**property** ClearAttributes : WordBool // clear the moved file's attributes.

**property** FailIfZeroFiles : WordBool;

### 5.6.7 Move File(s) Action

This action will Move a File or Files to the destination file or directory. You can use wildcards in the source setting to Move more than one file, in the Destination is a directory will be set automatically. If the target directories do not already exist, check the Force Directories property to make FinalBuilder create the directories as needed.



#### Scripting Info

The Action properties available are :

**property** FileSpec :WideString; // the file specification for the files to Move. You may use Wildcards.

**property** FilesAffected : integer ; // read only, the number of files Moved

**property** Target :WideString; // The target file or directory

**property** Force : WordBool; //Use Force Directories to create a directory and all parent directories that do not already exist.

**property** TargetIsDir : WordBool; // Specifies that the Target Is a Directory

**property** FailIfExists : WordBool; // Fail if the target file Exists

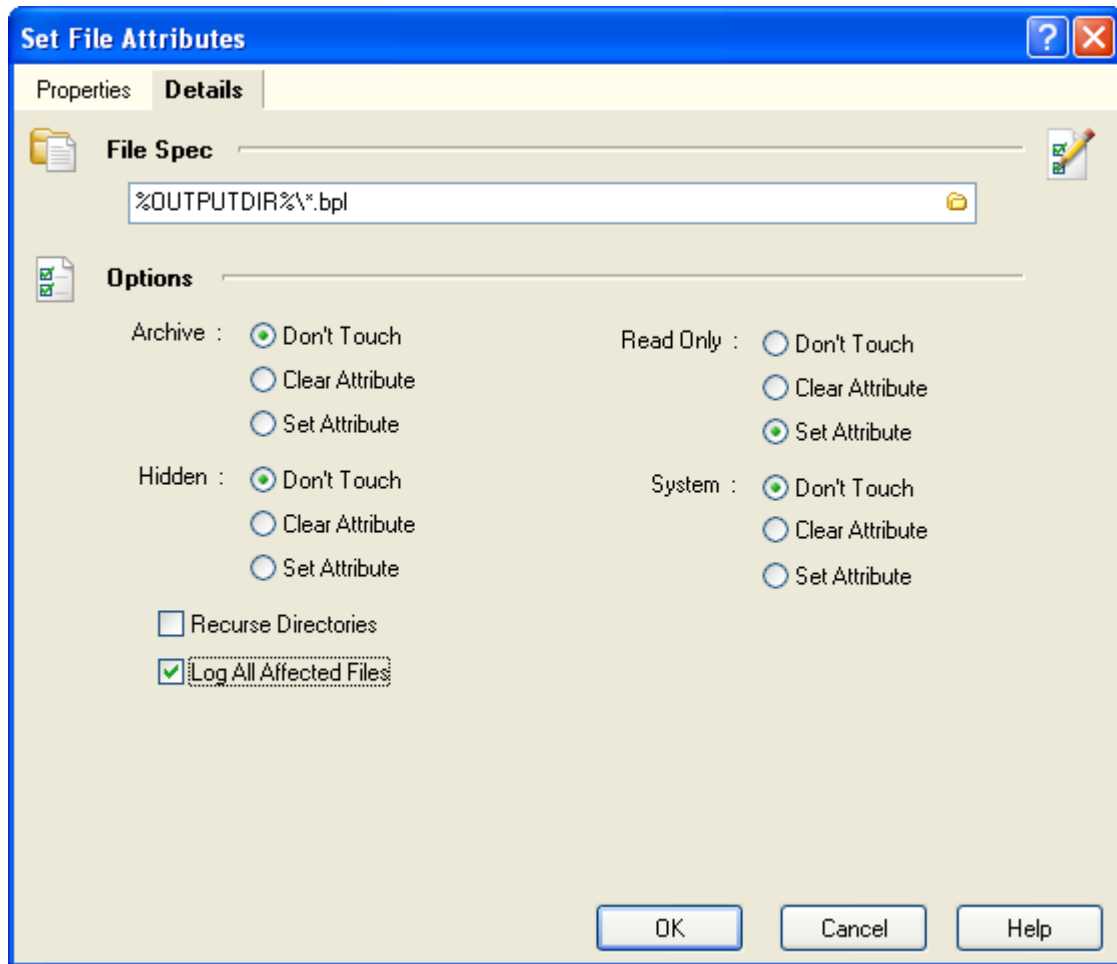
**property** OverwriteReadOnly : WordBool; //Overwrite Read Only Files

**property** Recurse : WordBool; // Recurse subdirectories, only valid if the target is a directory

**property** ClearAttributes : WordBool // clear the moved file's attributes.  
**property** FailIfZeroFiles : WordBool;

### 5.6.8 Set File Attributes Action

This action enables you to set a files attributes.



#### Scripting Info

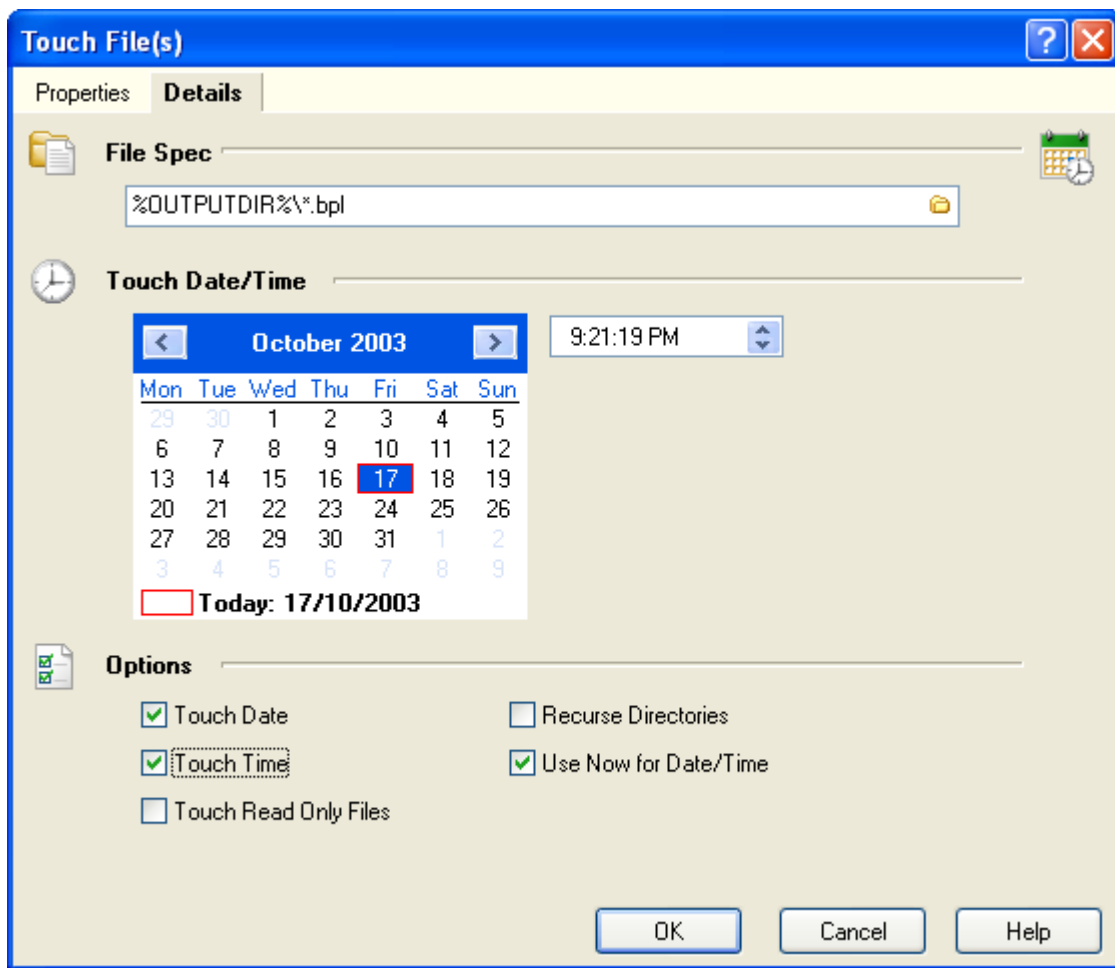
The Action properties available are :

**property** FileSpec : WideString ;// The file specification of the files to process. Wildcards (\*, ?) may be used  
**property** Recurse : WordBool; // Recurse Sub Directories  
**property** LogAllFiles : WordBool; // Log All Files processed to the FinalBuilder Output Tree.

### 5.6.9 Touch File(s)

This Action type provides the same functionality as the Touch command line utility provided with many C++ tools. It allows you to set the File Date and or Time for a file or Files.





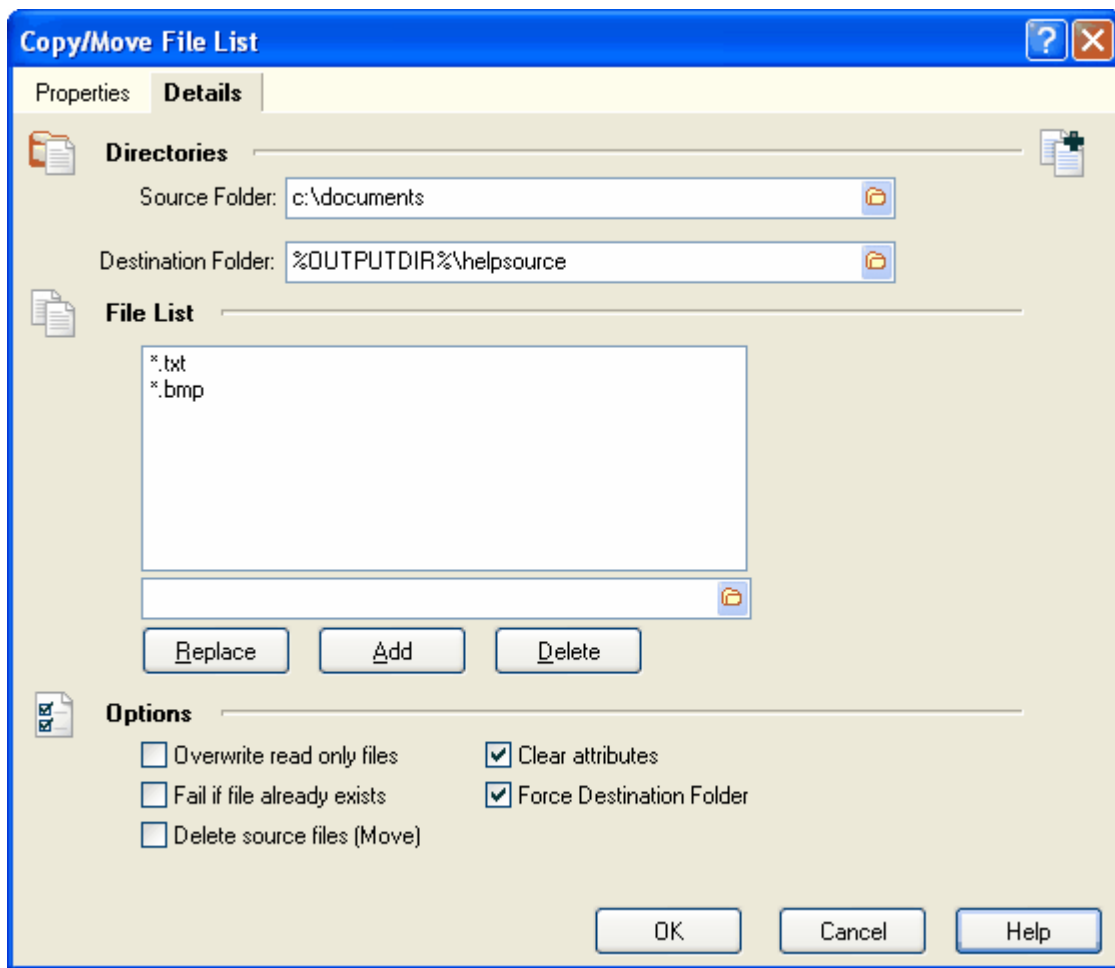
### Scripting Info

The Action properties available are :

**property** DateTime: TDateTime; // Delphi DateTime  
**property** FileSpec: WideString; // files to touch, can include wildcards  
**property** TouchDate: WordBool; // set the date  
**property** TouchTime: WordBool; // set the time  
**property** TouchReadOnly: WordBool; // by default, read only files will not be touched.  
**property** Recurse: WordBool; // recurse directories

### 5.6.10 Copy/Move File List

This action was written by Jim Gunkel from Nevrona Designs. It makes it possible to provide several different file specs for the files that should be moved/copied. Jim has kindly made this action available to all FinalBuilder users. The source for this action is installed as an example of creating custom FinalBuilder Actions. This action is supported by VSoft Technologies as a regular part of FinalBuilder!



**Source Folder :** The folder where the files will be copied/moved from

**Destination Folder :** The Folder where the files will be copied/moved to.

**Overwrite Read Only Files :** Overwrite existing read only Files.

**Clear Attributes :** Clear the file attributes during the copy/move.

**Fail if file already exists :** Fail if a file to be copied/moved already exists in the destination directory

**Delete Source Files (Move) :** Move the files (default is copy)

**Force Destination Folder :** When set, the Destination Folder will be created if it does not exist

**File List :** The list of files to be moved/copied from the Source Folder. This can include wildcards.

### Scripting Info

The Action properties available are :

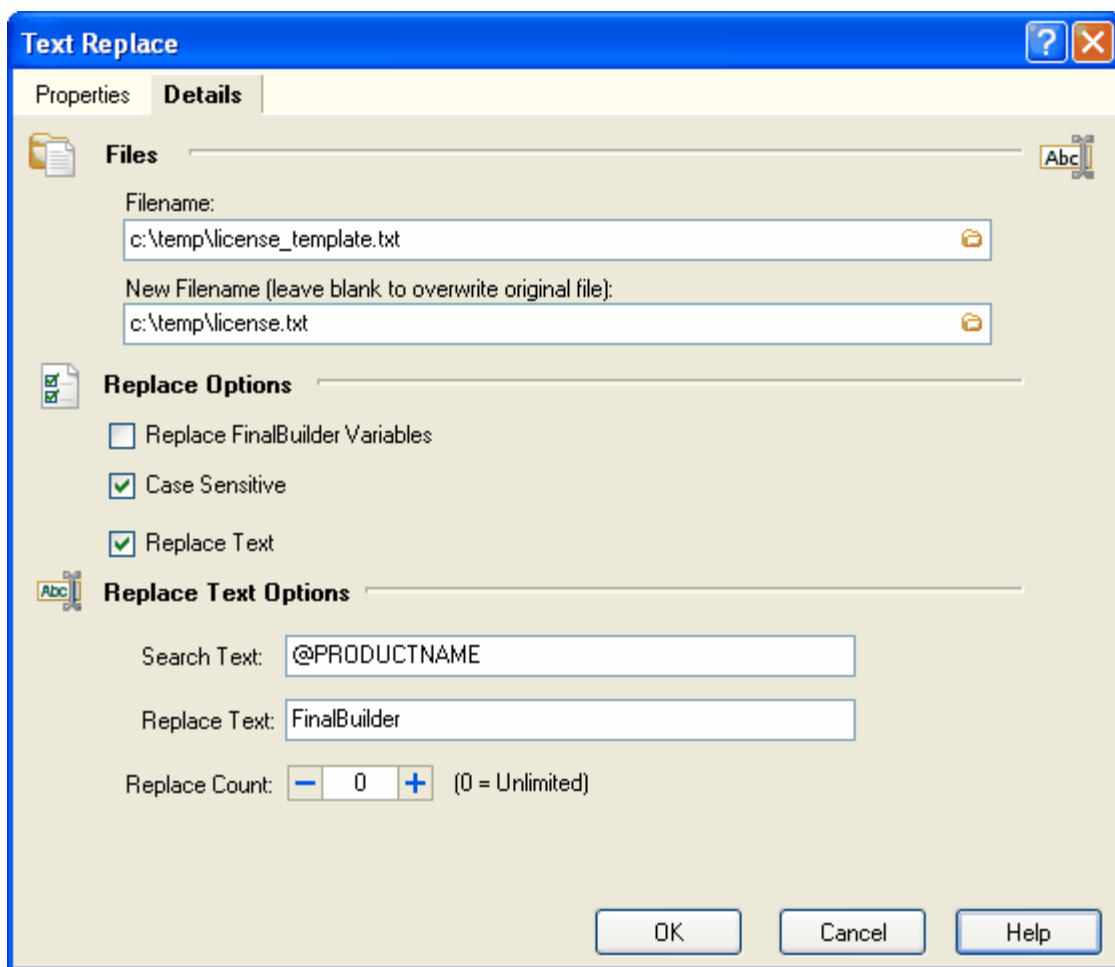
**property** SourceFolder: string;

**property** DestFolder: string;

**property** FileList: string;  
**property** DeleteSource: boolean;  
**property** ClearAttributes: boolean;  
**property** FailExists: boolean;  
**property** OverwriteReadOnly: boolean;

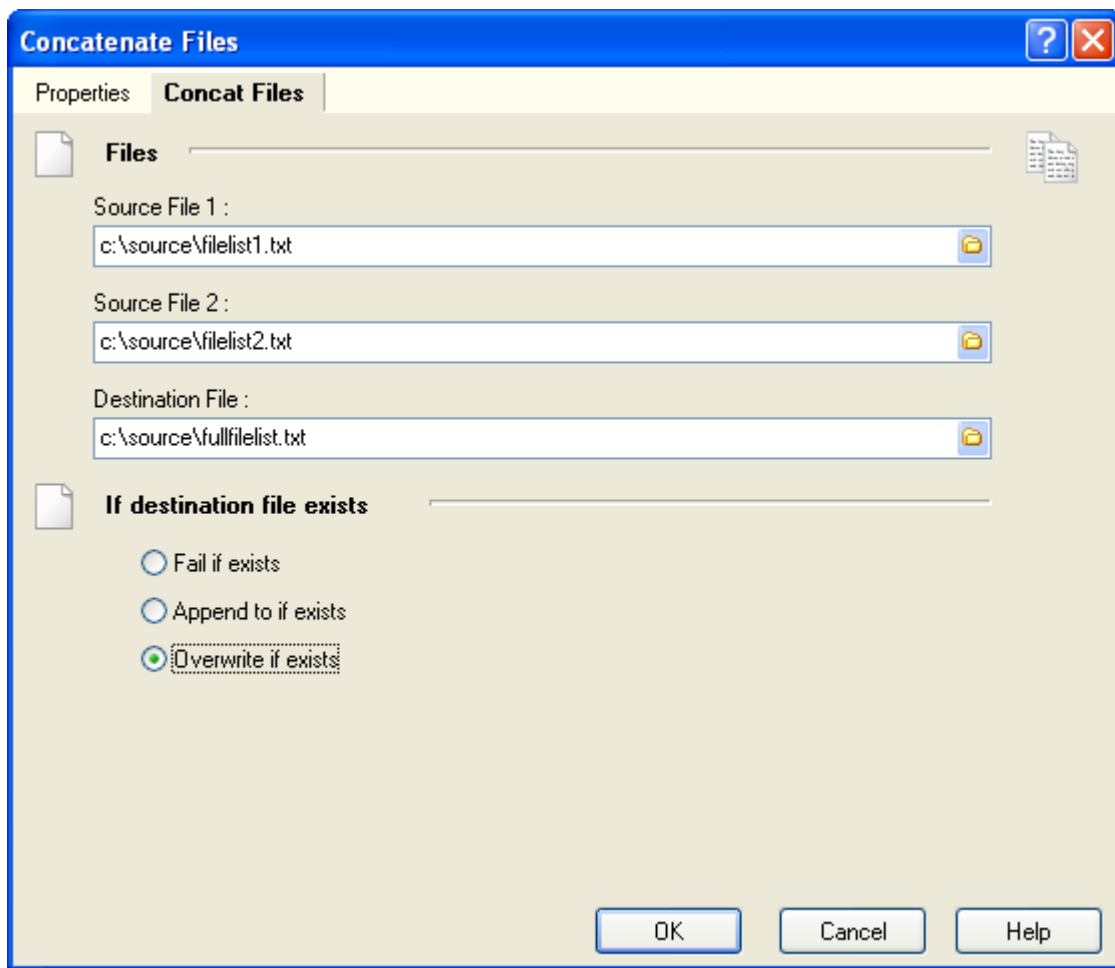
### 5.6.11 Text Replace Action

The text replace action allows you to replace strings and FinalBuilder variables in the file with specified values. Note that this will only work on Text files, do not attempt to use it with binary files.



### 5.6.12 Concatenate Files Action

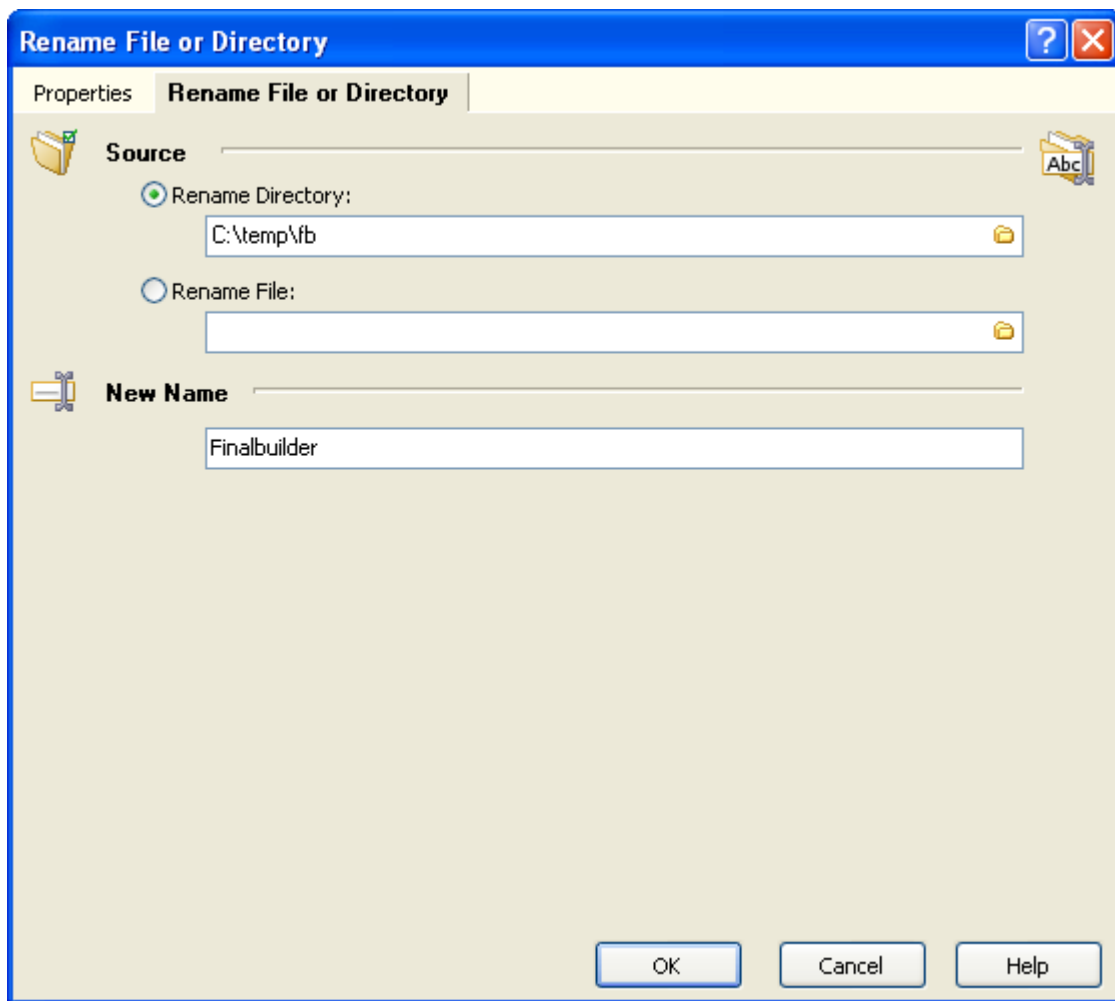
This action concatenates two files into a single new file.



Note that Source Files 1 & 2 cannot be the same file, and they cannot be the same as the destination file. Source 2 may be blank if appending to an existing file.

### 5.6.13 Rename File or Directory Action

This action allows you to Rename a File or a Directory.



The NewName property is the new name of the file or directory (without the path!).

#### 5.6.14 Authenticode

The Authenticode File action enables you to automate Authenticode signing of your executable files during your build process.

This action requires your Authenticode Signing Certificate to be in a PFX file. Most CA's are provided your certificate in two files: an SPC and a PVK.

To make the PFX, you need to use a Microsoft tool called PVKIMPRT [more info: <http://msdn.microsoft.com/vba/technical/pvk.asp>] or [download: <http://download.microsoft.com/download/vba50/Utility/1.0/NT5/EN-US/pvkimprt.exe>].

Then, open a console window (dos box), switch to the directory that contains your certificates and type:

```
pvkimprt -PFX mycert.spc mykey.pvk  
(Replace the mycert and mykey file names where required).
```

You will be prompted to enter your private key password (if set), and then you get the certificate export wizard. Make sure you select 'Yes, export the private key', and 'Include all certificates in the certification path if possible' options. You are then prompted to enter a password: this is the password you will use for code signing with the created PFX -- suggestion: pick a GOOD password. Finally, you get prompted as to where you would like the PFX created.

This page on the VeriSign website provides some useful information also:  
<http://www.verisign.com/resources/gd/authenticode/index.html>

---

This action includes source code from StreamSec. It's license is reproduced here.

#### StreamSec LICENSE

Copyright (c) 2004, Henrick Wibell Hellström, StreamSec

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- \* Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

- \* Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

- \* Neither the name of StreamSec nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS

OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY

AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR

CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL

DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,

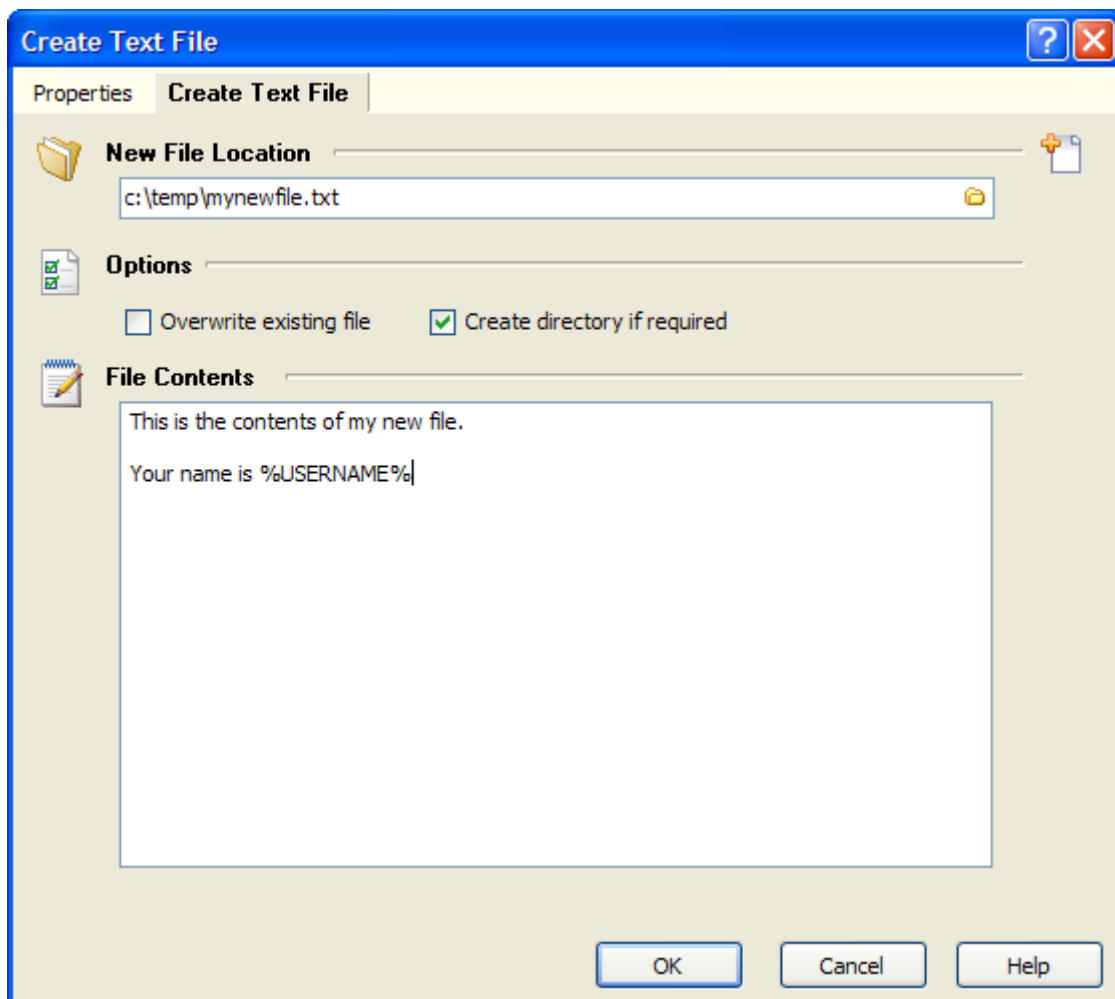
DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER

IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT

OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

### 5.6.15 Create Text File

The Create Text File action enables you to automate the creation of text files during your build process. You can use FinalBuilder variables in the File Contents field to customise the file based on the build.



**New File Location** - path to new file to create

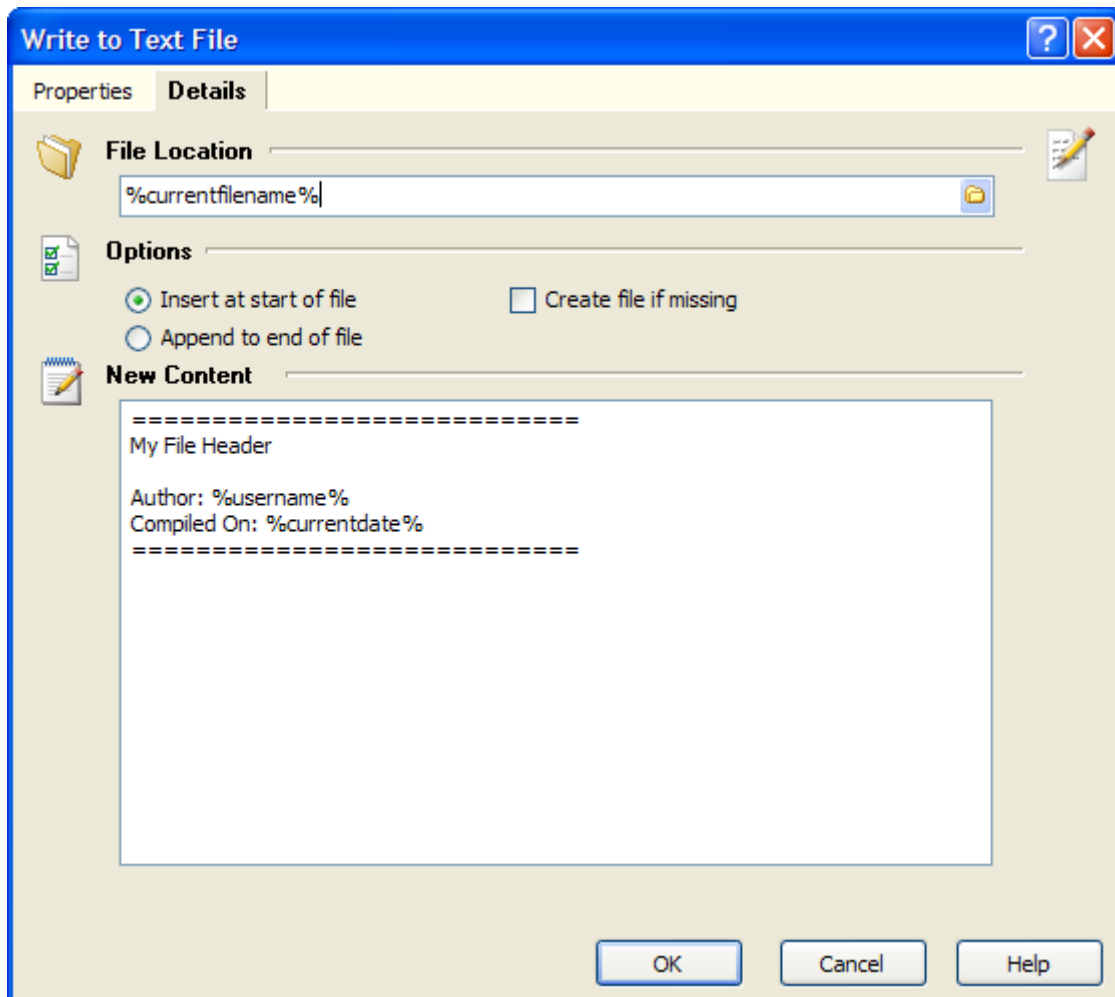
**Overwrite Existing File** - if this is not set, and the file already exists, the action will fail

**Create Directory if required** - if any directory doesn't exist in the new file location, then it will be automatically created

**File Contents** - specify the text to be written to the file. You may use FinalBuilder variables

### 5.6.16 Write Text File

The Write to Text File action enables you to automate appending to text files during your build process. You can use FinalBuilder variables in the File Contents field to customise the content based on the build. You can either append to the end of the file, or insert text at the start of the file.



**File Location** - specify the filename to write to (the example shows a variable which will contain a filename during execution - provided by a file iterator action)

**Insert location** - Specify to append to end of file, or insert at beginning

**Create file if missing** - the action will normally fail if the file doesn't already exist.

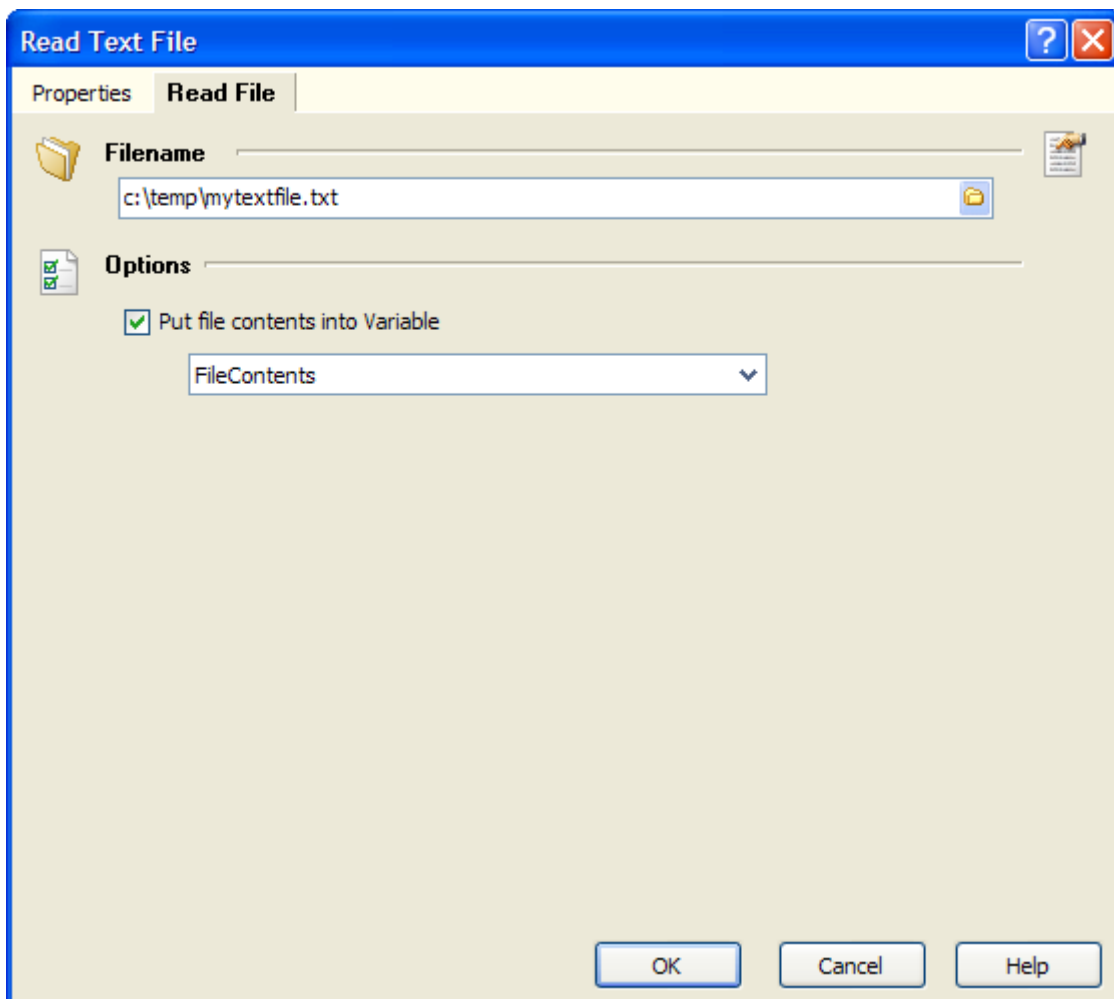
**New Content** - specify the content to write to the text file

### 5.6.17 Read Text File

The Read Text File action enables you to automate the reading of a text file as part of your build process

You can read the text file into a FinalBuilder variable, and also process each line in the OnReadLine event.





**Filename** - specify the text file to read

**Put file contents into variable** - specify this option (and a variable) so that you can examine and manipulate the contents of the file during your build process

When the action executes, the "**OnReadLine**" event is called for each line of the text file which makes it easy to process the file line-by-line.

### 5.6.18 XCopy

The XCopy action enables you to automate file and directory copying using XCopy as part of your build process. XCopy is a powerful utility included in Windows which is faster and more capable than standard file and copy tools.

For more information on XCopy see:

<http://www.microsoft.com/resources/documentation/windows/xp/all/proddocs/en-us/xcopy.msp>

### 5.6.19 RoboCopy

The RoboCopy action enables you to automate file and directory copying using RoboCopy as part of your build process. Robocopy is a very powerful tool for copying, moving, and synchronising directories and files with error recovery. It is included in the Windows NT

Resource Kit.

The RoboCopy actions are:

Robocopy - perform file and directory copy operations

Robocopy Job - run Robocopy Jobs

Robocopy Move - perform file and directory move operations

Robocopy Mirror - synchronise source directory to destination directory

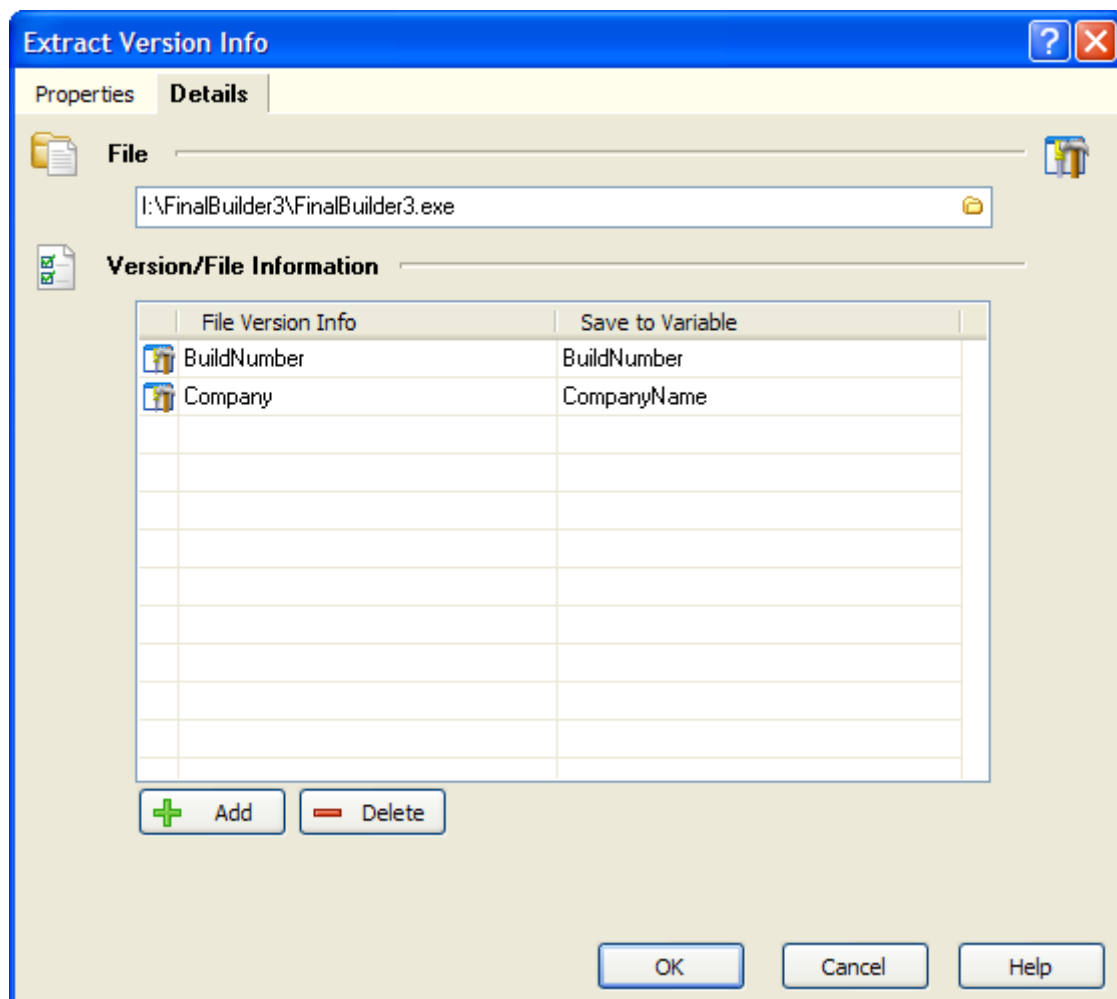
For more information on Robocopy see:

<http://www.microsoft.com/resources/documentation/WindowsServ/2003/all/techref/en-us/Default.asp?url=/Resources/Documentation/windowsserv/2003/all/techref/en-us/robocopy.asp>

### 5.6.20 Extract File Version

The Extract File/Version Information action enables you to extract the Win32 Version and File Information stored in the executable.

Extracting file information works with both normal Win32 files as well as .Net assemblies.



**File** - Specify a Win32 or .Net EXE, OCX, or DLL file (may work with other types too)

**File Version Info** - Specify the field you want to extract. You may either select from the predefined types, or type in your own value

**Save To Variable** - Specify the FinalBuilder variable to save the value to

### 5.6.21 Calculate File CRC32

The File CRC32 action enables you to automate the calculation of a file's CRC32 as part of your build process. The action can also fail if the CRC differs from another file or a specified CRC32.

#### What is CRC32?

The CRC is acronym for "Cyclic Redundancy Code" and 32 represent the length of checksum in bits. The "CRC" term is reserved for algorithms that are based on the "polynomial" division idea. The idea to compute the checksum is equal for all CRC algorithms: Take the data as a VERY long binary number and divide it by a constant divisor. If you do this with integer values you get a rest; this rest is the CRC checksum (for example  $7 / 3 = 2 + \text{rest } 1 \Rightarrow 1$  is the checksum of 8. CRC is a family of algorithms and CRC32 is one certain member of this family (other members are CRC16, XMODEM,...); CRC32 produces a checksum with a length of 32 Bit (= 4Byte).

**Calculate File CRC32**

Properties Details

**Main File**

Filename  
I:\FinalBuilder3\ActionStudio.exe

Save CRC32 to Variable  
ActionStudioCRC32

**Options**

☐ Fail if CRC32 is different to file

☐ Fail if CRC32 is different to CRC32

OK Cancel Help

**Filename** - specify the file which you want to calculate the CRC32 for.

**Variable** - specify the FinalBuilder variable to save the calculated file CRC32 to.

**Fail if CRC32 is different to file** - this calculates the CRC32 of the specified file and compares it against the CRC32 of the main file and fails the action if the two differ.

**Fail if CRC32 is different to CRC32** - specify a CRC32 to compare to the calculated CRC32 of the main file

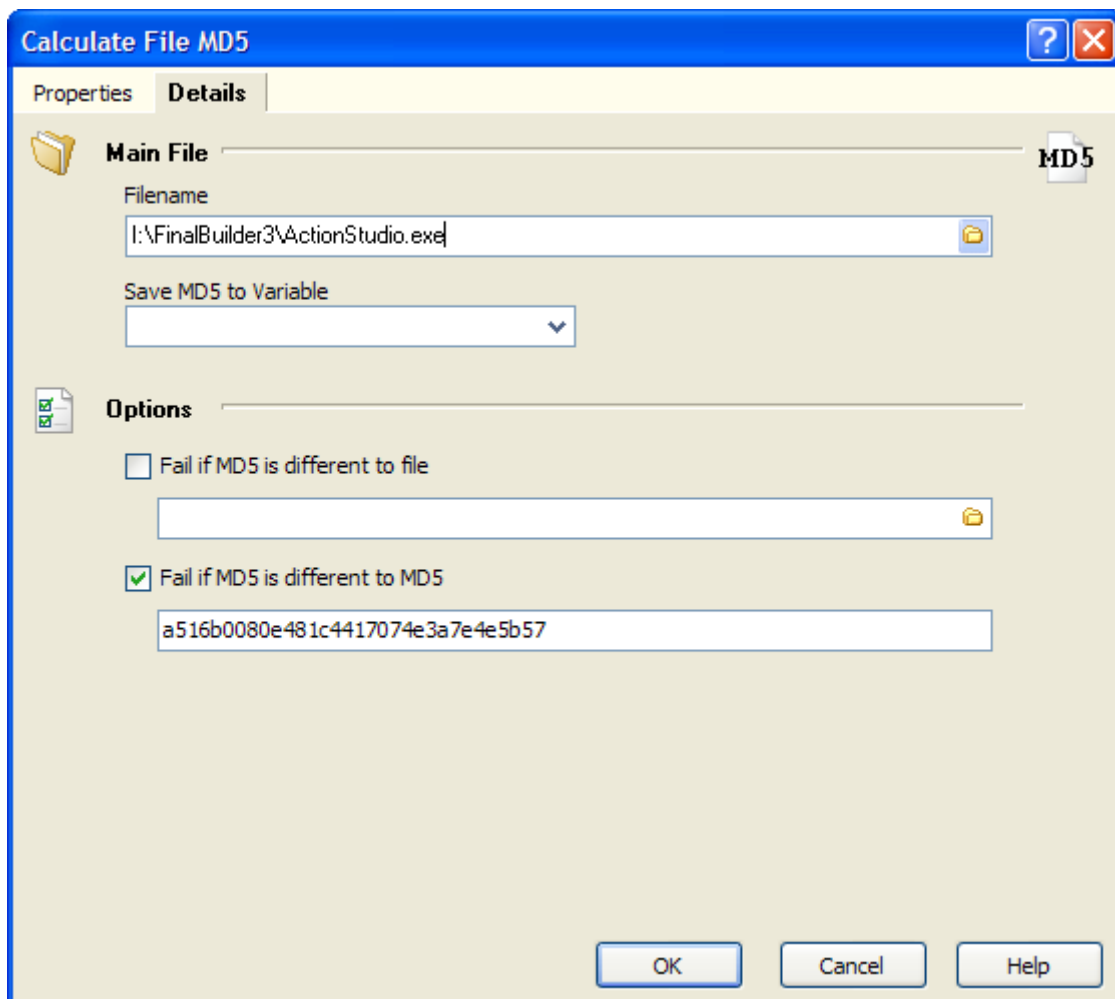
### 5.6.22 Calculate File MD5

The File MD5 action enables you to automate the calculation of a file's MD5 as part of your build process. The action can also fail if the MD5 differs from another file or a specified MD5.

#### **What is MD5?**

The MD5 algorithm takes as input a message of arbitrary length and produces as output a 128-bit "fingerprint" or "message digest" of the input. It is conjectured that it is computationally infeasible to produce two messages having the same message digest, or to produce any message having a given pre-specified target message digest. The MD5 algorithm is intended for digital signature applications, where a large file must be "compressed" in a secure manner before being encrypted with a private (secret) key under a public-key cryptosystem such as RSA.

In essence, MD5 is a way to verify data integrity, and is much more reliable than checksum and many other commonly used methods.



**Filename** - specify the file which you want to calculate the MD5 for.

**Variable** - specify the FinalBuilder variable to save the calculated file MD5 to.

**Fail if MD5 is different to file** - this calculates the MD5 of the specified file and compares it against the MD5 of the main file and fails the action if the two differ.

**Fail if MD5 is different to MD5** - specify a MD5 to compare to the calculated MD5 of the main file

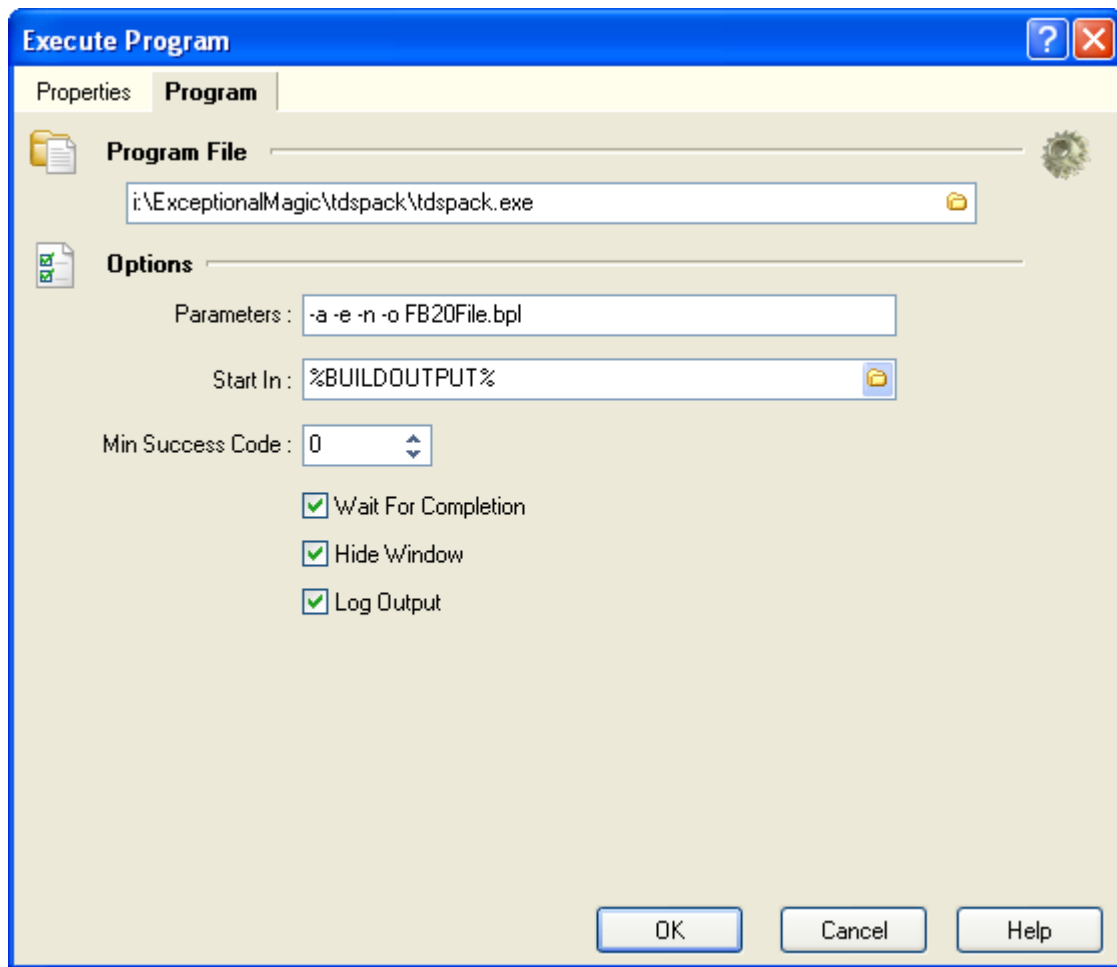
## 5.7 Windows OS

### 5.7.1 Execute Program Action

The Execute Program Action allows you to execute nearly any program from a FinalBuilder Build process. Finalbuilder can capture the output of console applications (such as command line compilers) and display the captured output in the FinalBuilder output window. By Default, FinalBuilder will wait for the program execution to complete, however you can turn this option off. when Wait For Completion is turned off, the action will complete as soon as the program begins executing. The output is not available for capture when Wait For Completion is turned off.

Some applications may not return a zero exit code when closing, even though they may have

completed successfully. In these rare cases you can set the min success code to an appropriate value so that FinalBuilder can detect if the application executed successfully.



### Scripting Info

The Action properties available are :

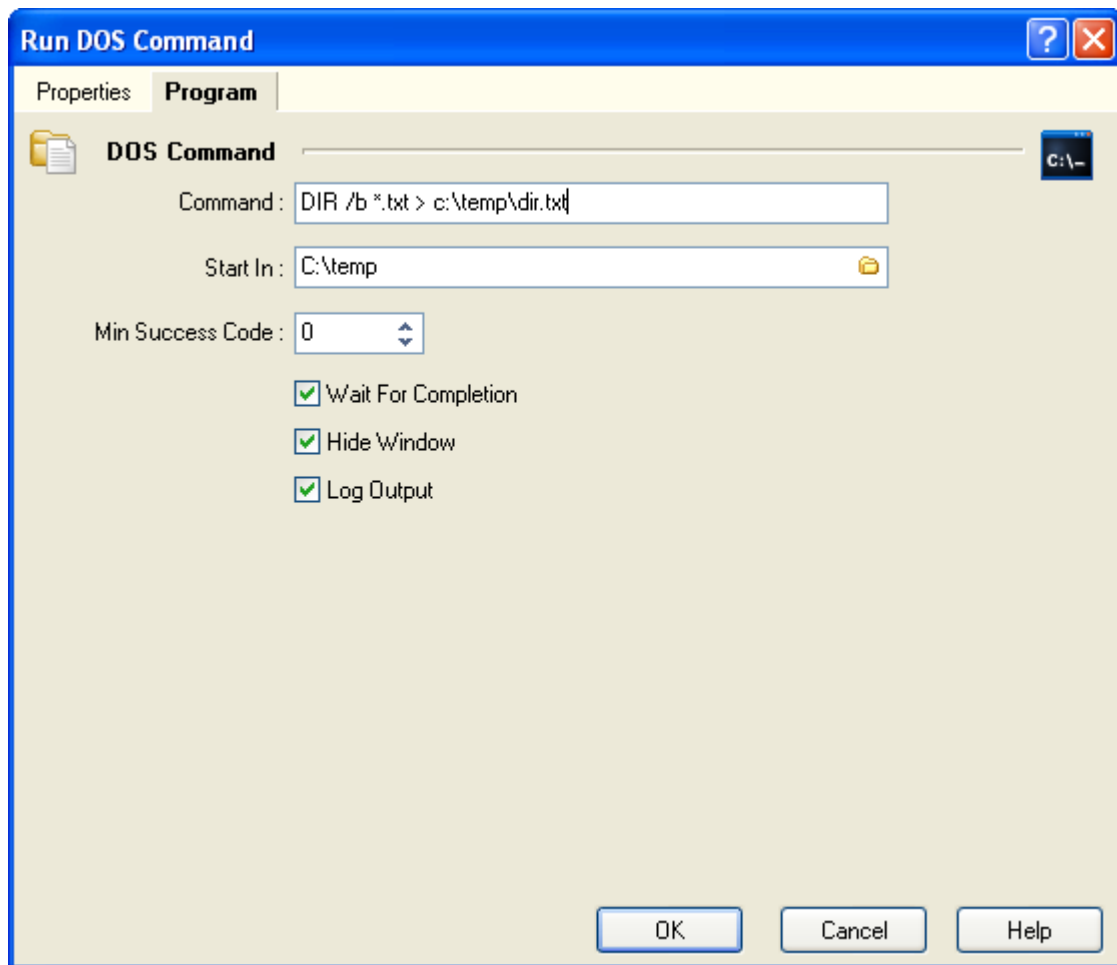
```

property ProgramName : WideString;
property Params : WideString;
property StartInDir : WideString;
property LogOutput : WordBool;
property WaitForCompletion : WordBool;
property ErrorCode : integer; //read only
property HideWindow : WordBool;
property SuccessRC : integer;
  
```

These properties may be set in the BeforeAction and AfterAction Script events.

### 5.7.2 Run DOS Command Action

The Run DOS Command allows you to execute any DOS command.



#### Scripting Info

The Action properties available are :

**property** StartInDir : WideString; // the starting directory for Dos

**property** LogOutput : WordBool; // capture the output of the command. Note that the output cannot always be captured ( for example XCopy)

**property** WaitForCompletion : WordBool; //wait for this command to complete for continuing to the next action

**property** ErrorCode : integer; //read only

**property** HideWindow : WordBool; // Hide the window while executing the dos command

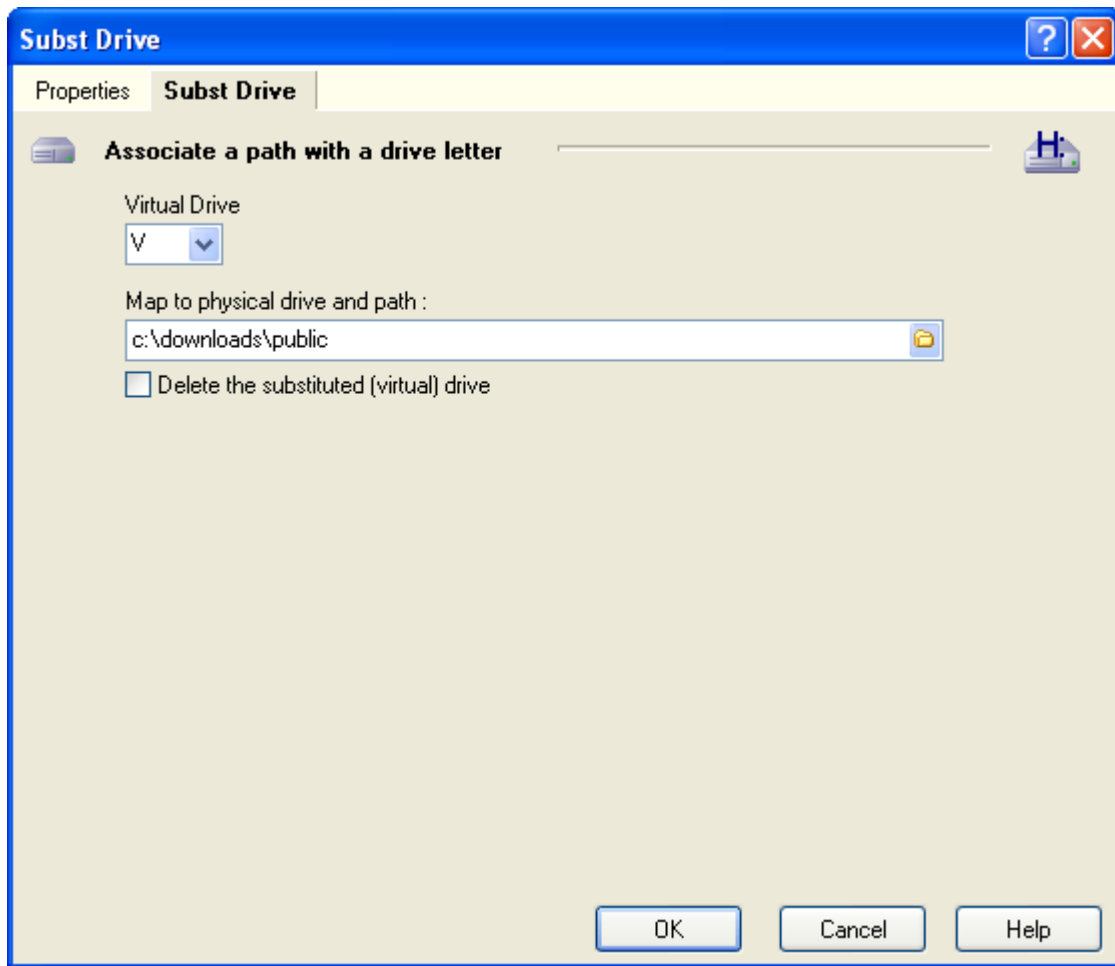
**property** SuccessRC : integer; // this is the lowest return code that should be deemed a successful execution. Some applications return a non-zero code even though they do actually execute ok.

**property** Command : WideString; // the dos command!

These properties may be set in the BeforeAction and AfterAction Script events.

### 5.7.3 Subst Drive Action

This action calls the dos subst.exe, to map a folder to a drive letter.

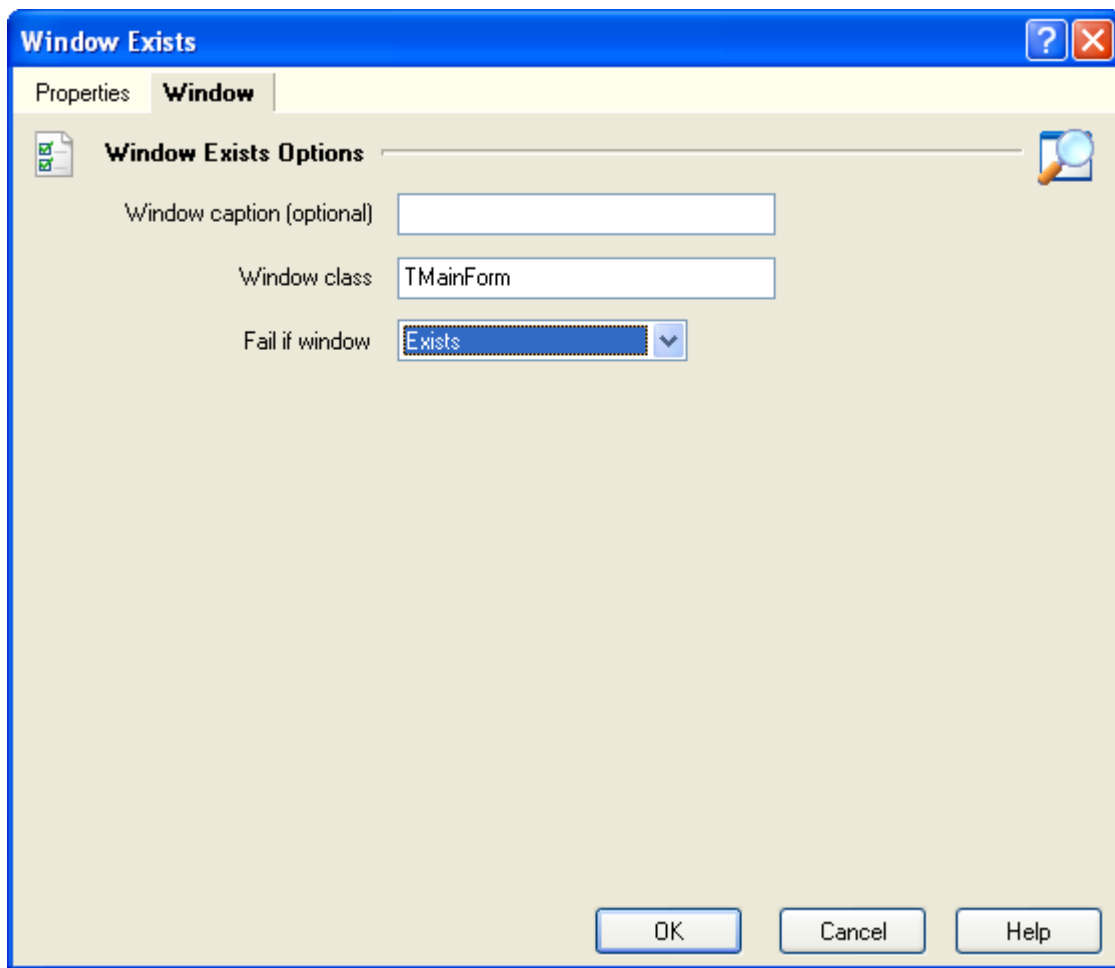


### 5.7.4 Window Exists Action

This Action enables you to check if an application is running by checking for a Window Caption or Window Class (windows API window class name), and choose to fail if the window exists or fail if the window doesn't exist.

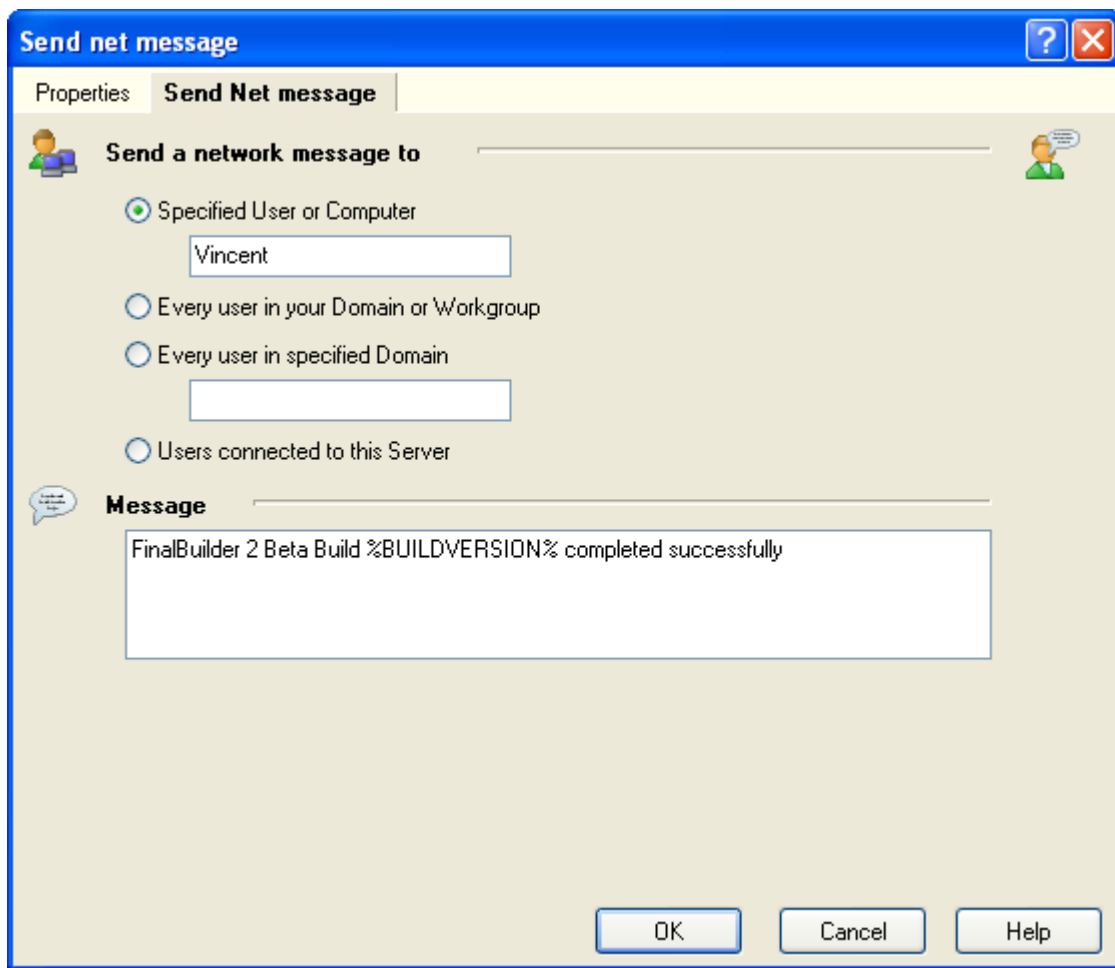
This action was written and kindly donated by Erik Berry, maintainer of [GExperts](http://www.gexperts.com)





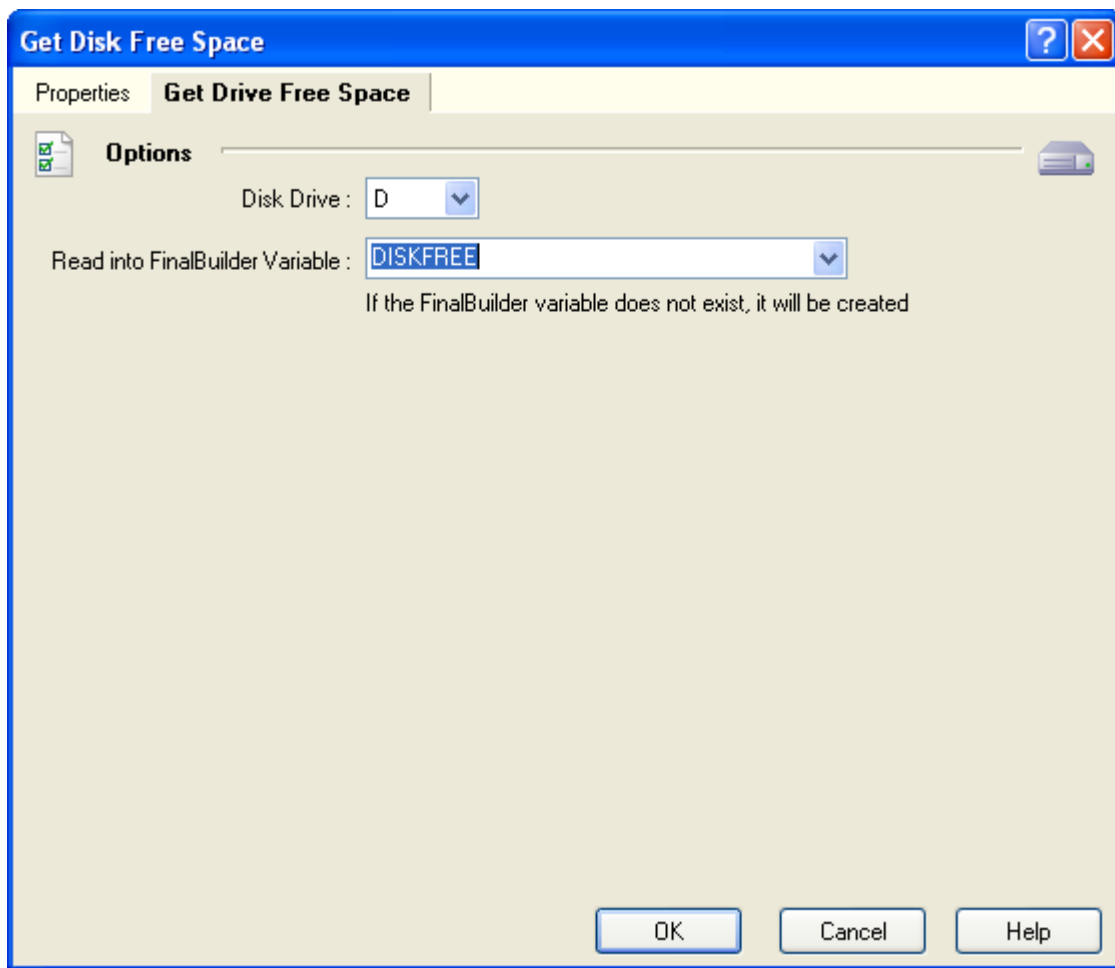
### 5.7.5 Net Send Message Action

Send a Message using windows NET SEND



### 5.7.6 Get Disk Free Space

Gets the Free Disk Space into a FinalBuilder variable. If the variables does not already exist then it will be created.



The disk space available on the drive is returned in BYTES.  
If the drive specified is not available, then the action will fail.

If you need to dynamically set the Disk Drive during your build, then you can access the Drive property in script. eg:

BeforeAction script:

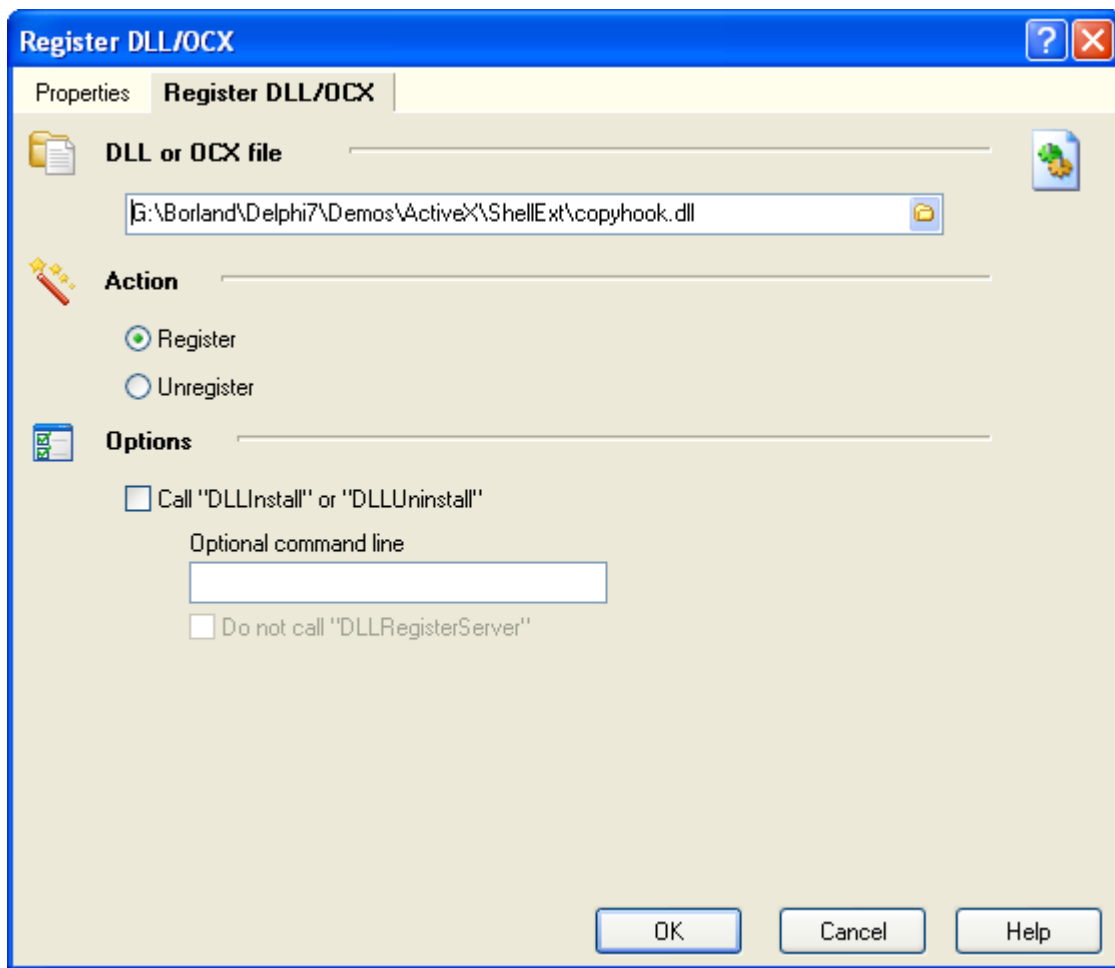
Action.Drive = 'R'

or

Action.Drive = MyDynamicDriveLetter

### 5.7.7 Register DLL/OCX Action

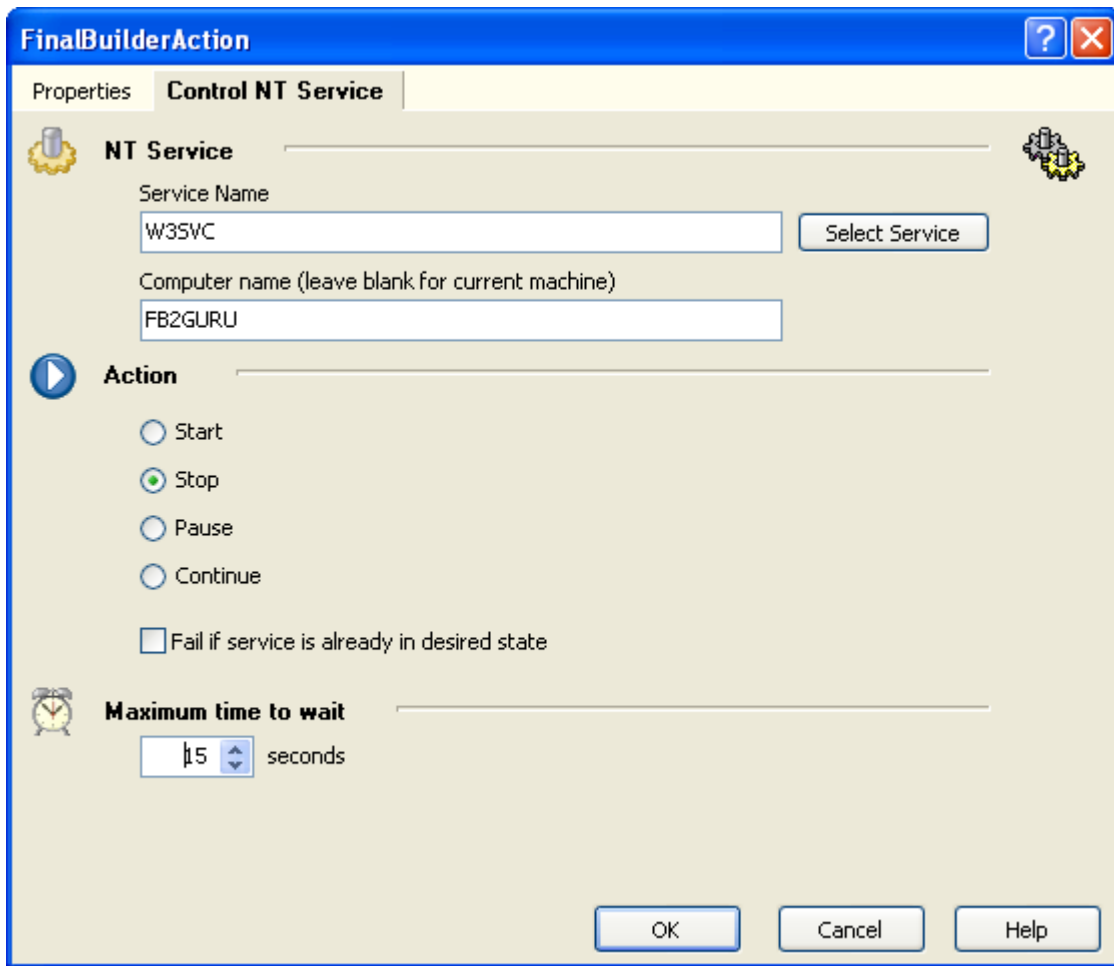
Register an activeX control (OCX) or COM DLL.



### 5.7.8 Control Service Action

[Professional Edition]

This action allows you to control the state of a windows service on the local machine or a remote machine.



The image shows a Windows-style dialog box titled "FinalBuilderAction". It has a blue title bar with a question mark icon and a close button. Below the title bar is a tabbed interface with two tabs: "Properties" and "Control NT Service". The "Control NT Service" tab is active. The dialog is divided into three main sections. The first section, "NT Service", is preceded by a gear icon. It contains a "Service Name" text box with "W3SVC" entered, a "Computer name (leave blank for current machine)" text box with "FB2GURU" entered, and a "Select Service" button. The second section, "Action", is preceded by a play button icon. It contains four radio buttons: "Start", "Stop" (which is selected), "Pause", and "Continue". Below these is a checkbox labeled "Fail if service is already in desired state", which is currently unchecked. The third section, "Maximum time to wait", is preceded by an alarm clock icon. It contains a spinner box showing "15" and the text "seconds". At the bottom of the dialog are three buttons: "OK", "Cancel", and "Help".

**FinalBuilderAction**

Properties **Control NT Service**

**NT Service**

Service Name  
W3SVC Select Service

Computer name (leave blank for current machine)  
FB2GURU

**Action**

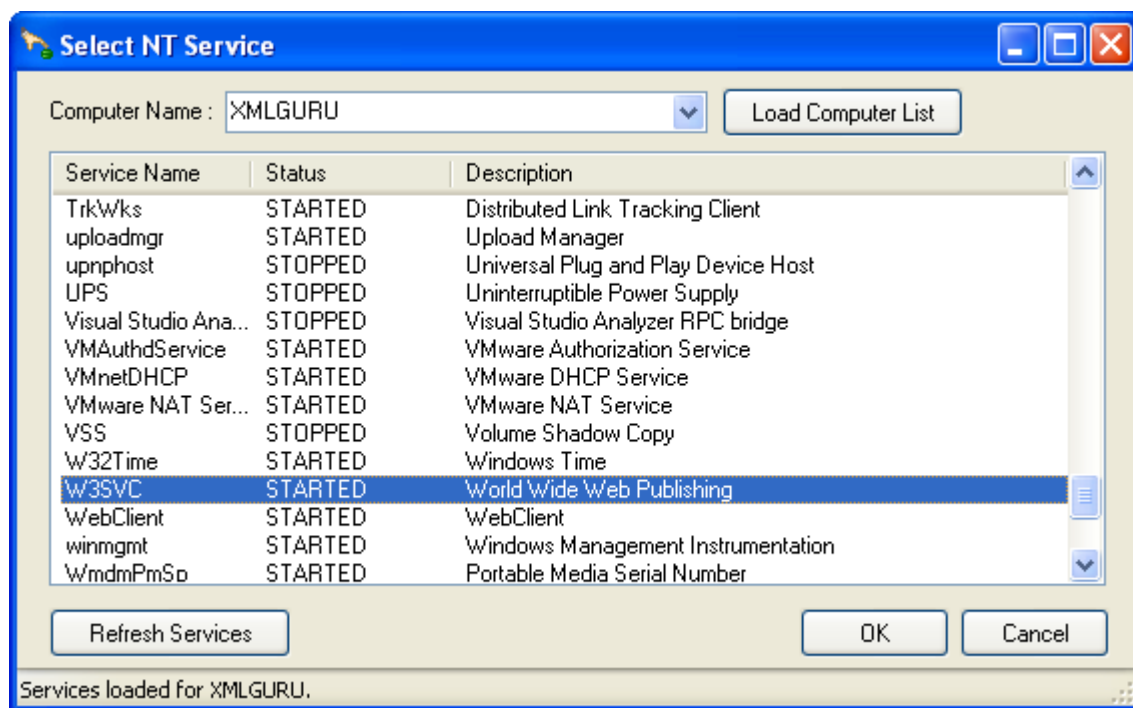
☐ Start  
☒ Stop  
☐ Pause  
☐ Continue

☐ Fail if service is already in desired state

**Maximum time to wait**  
15 seconds

OK Cancel Help

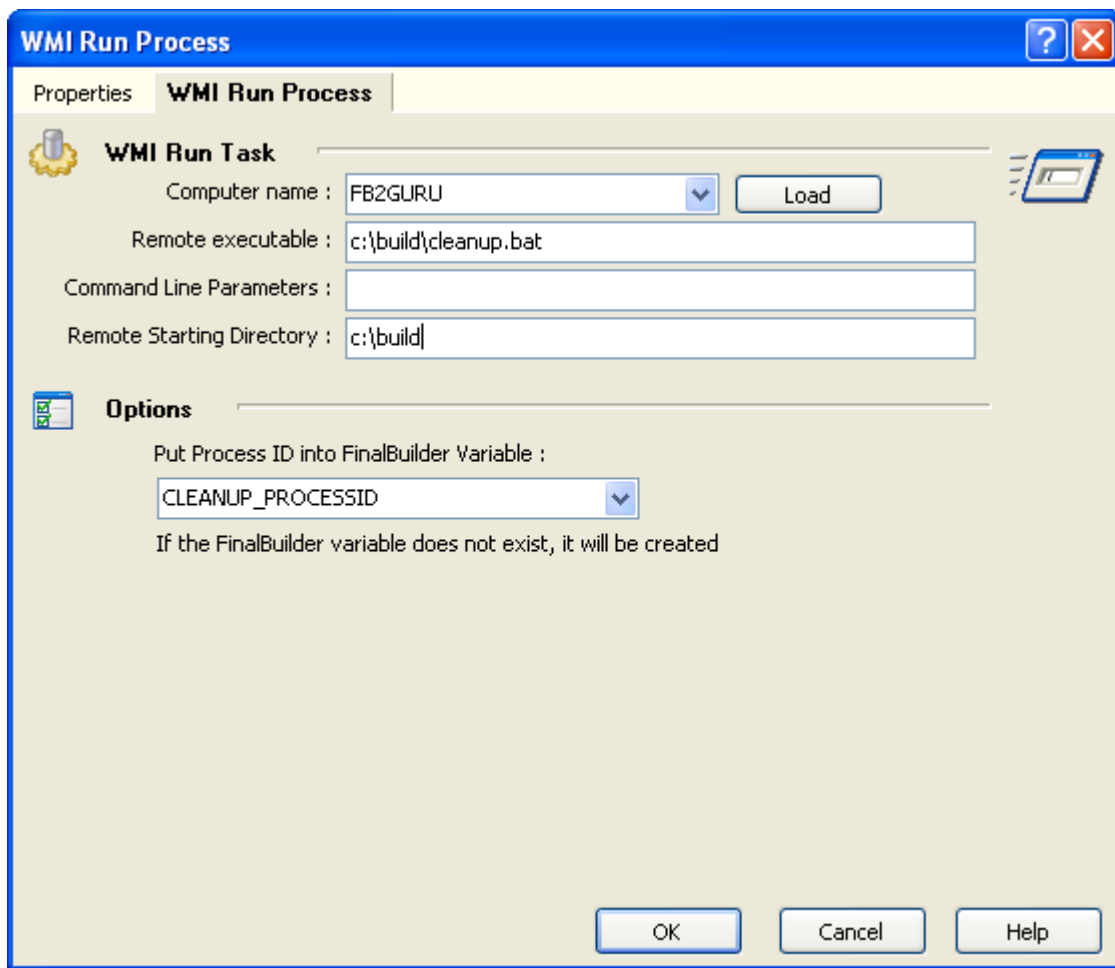
Click on the Select service to choose which service to control. By default the registered services of the local machine are listed, to change this either type the name of the computer or select from the drop down list (click on load to get a list, this is not done automatically as it can be slow in a large network) and the click on the Refresh Services button.



### 5.7.9 WMI Run Process Action

#### [Professional Edition]

This action uses the WMI (Windows Management Interface) API to execute a process on a remote or local machine. Note that the path to the executable should be the path on the specified machine, and the executable must exist on that machine. WMI is supported on NT4 (you need to download it from Microsoft), Windows 2000 and XP.



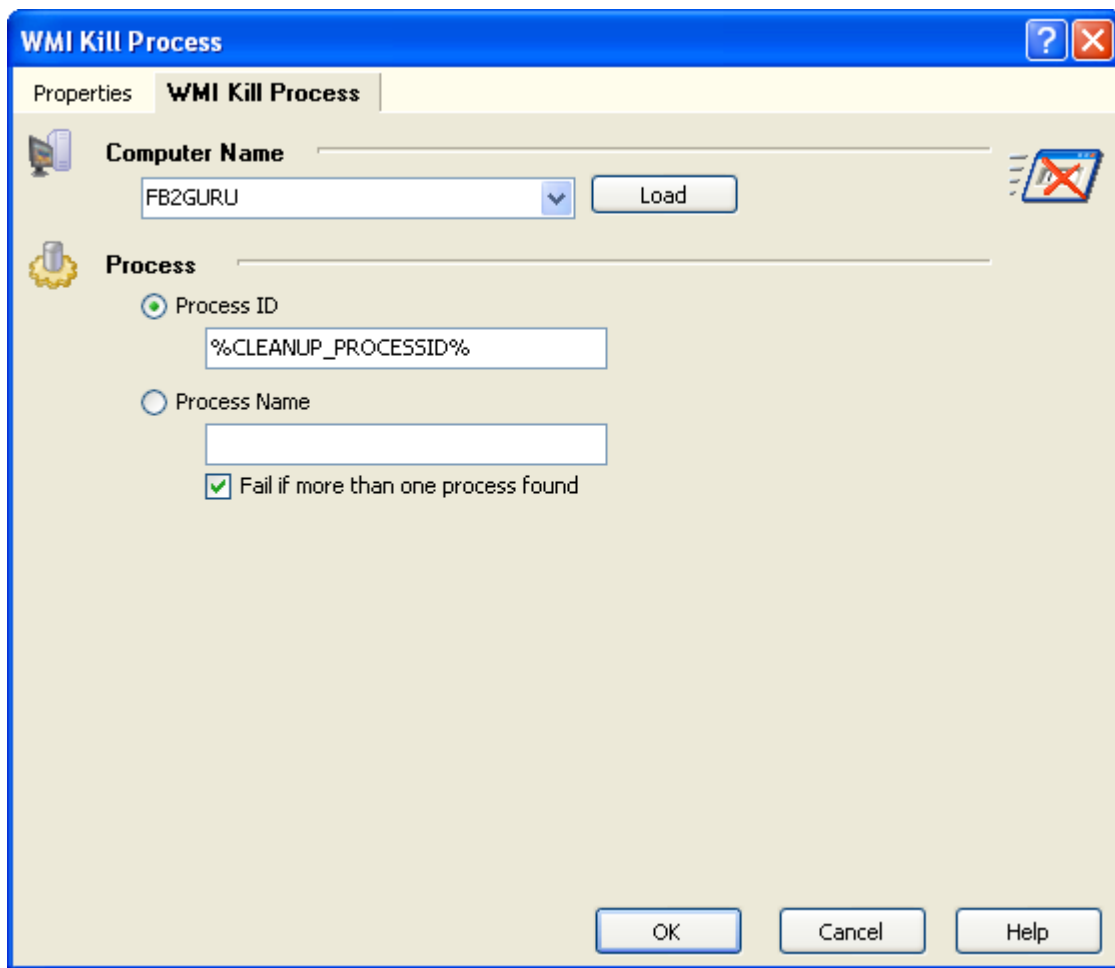
You can choose to save the process ID in a FinalBuilder variable so that it may be used in later actions such as the WMI Kill Process Action.

#### 5.7.10 WMI Kill Process Action

##### [Professional Edition]

This action use the WMI (Windows Management Interface) API to execute a process on a remote or local machine. You can specify a processID or the process name.

WMI is supported on NT4 (you need to download it from Microsoft), Windows 2000 and XP.



### 5.7.11 WMI Process Info Action

[Professional Edition]

This action allows you to interrogate a process for information, or just check if the process is running or not.



**WMI Process Info**

Properties **WMI Process Info**

**Computer Name**

XMLGURU Load

**Process**

☐ Process ID

☒ Process Name

inetinfo.exe

☒ Fail if more than one process found

**Options**

☐ Fail if process is running

☒ Fail if process is not running

OK Cancel Help

To get access to the process information you need to use script in the AfterAction script event. The following properties can be read from the action in the AfterAction event :

```

property ComputerName : string;
property ProcessID    : string;
property ProcessName : string;

property ProcessCaption      : string;
property ProcessDescription  : string;
property ProcessHandleCount  : LongWord;
property ProcessExecutablePath : string;
property ProcessThreadCount  : LongWord;
property ProcessPriority      : LongWord;
property ProcessWorkingSetSize : LongWord;
property ProcessPeakWorkingSetSize : LongWord;

property ProcessPageFaults      : LongWord;
property ProcessPageFileUsage   : LongWord;
property ProcessParentProcessId : LongWord;
property ProcessPeakPageFileUsage : LongWord;
property ProcessPeakVirtualSize : LongWord;
property ProcessCreationDate    : TDateTime;
property ProcessKernelModeTime  : LongWord;

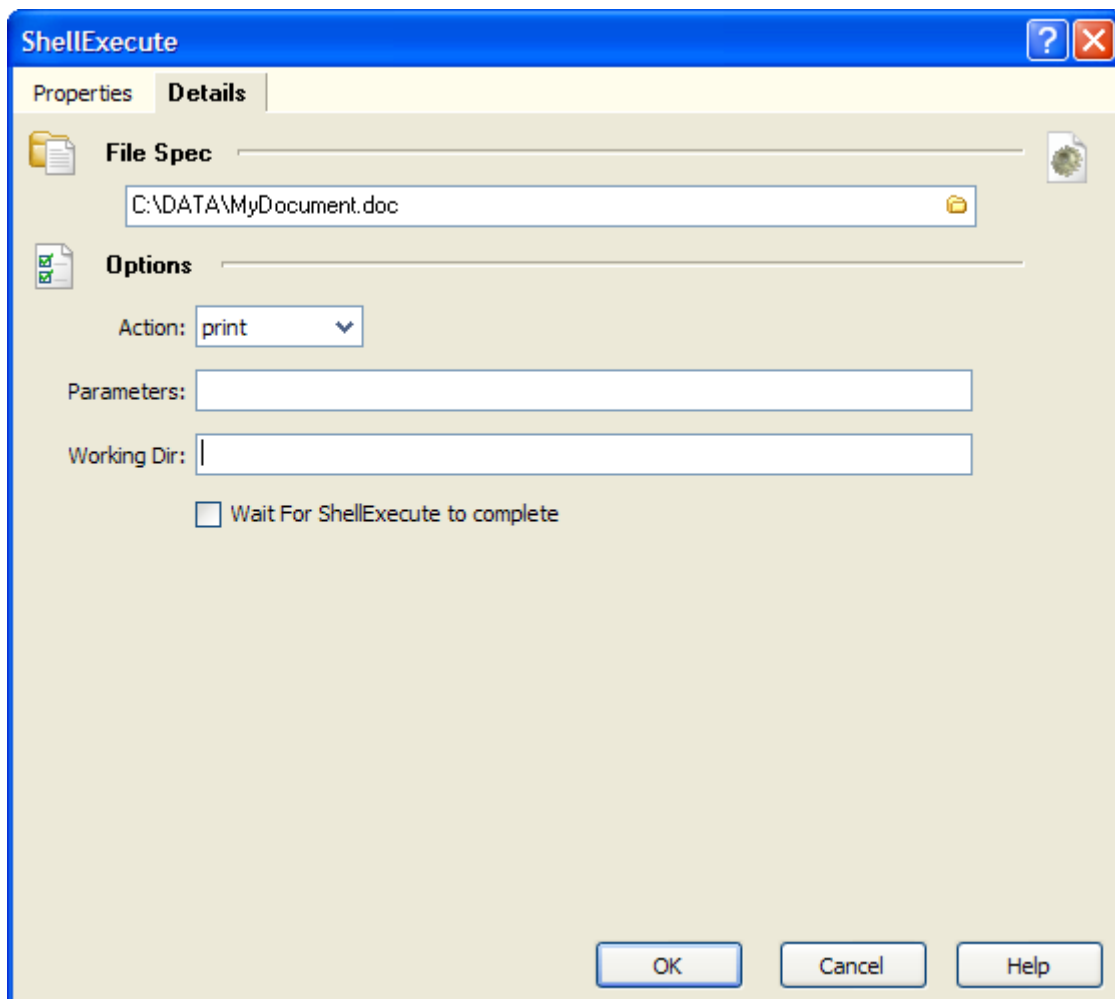
```

```
property ProcessMaxWorkingSetSize      : LongWord;  
property ProcessMinWorkingSetSize      : LongWord;  
property ProcessOtherOperationCount    : LongWord;  
property ProcessOtherTransferCount     : LongWord;  
property ProcessPrivatePageCount       : LongWord;  
property ProcessQuotaNonPagePoolUsage   : LongWord;  
property ProcessQuotaPagePoolUsage     : LongWord;  
property ProcessQuotaPeakNonPagePoolUsage : LongWord;  
property ProcessQuotaPeakPagePoolUsage : LongWord;  
property ProcessReadOperationCount     : LongWord;  
property ProcessReadTransferCount      : LongWord;  
property ProcessSessionId              : LongWord;  
property ProcessUserModeTime           : LongWord;  
property ProcessWriteOperationCount    : LongWord;  
property ProcessWriteTransferCount     : LongWord;
```

### 5.7.12 Shell Execute

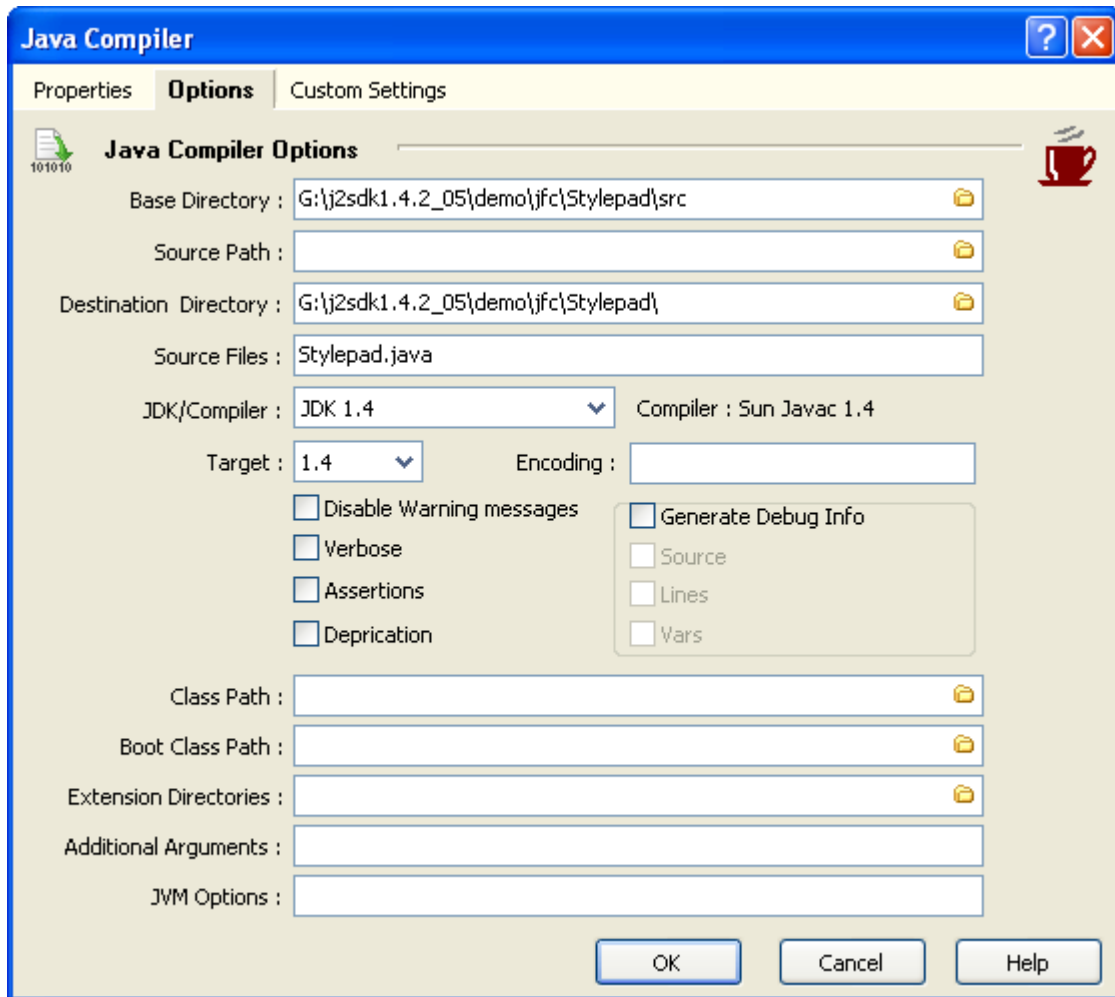
The Shell Execute action enables you to automate use the Windows Shell to act on a certain file.

For example, you could specify a .doc file and set the action to "print". This will ask the windows shell to print the document using the registered application for .doc files (eg. Word), see below:



## 5.8 Compilers

### 5.8.1 Java Compiler Action



The Java Compiler Action invokes the javac compiler which is included in the Sun Java JDK. It also supports invoking other java compilers such as Borland Java Compiler and IBM Jikes.

**Base Directory** - The current directory when the compiler is invoked.

**Source Path** - Specify the source code path to search for class or interface definitions. As with the user class path, source path entries are separated by semicolons (;) and can be directories, JAR archives, or ZIP archives. If packages are used, the local path name within the directory or archive must reflect the package name. Note that classes found through the classpath are subject to automatic recompilation if their sources are found.

**Destination Directory** - Sets the destination directory for class files. The destination directory must already exist; javac will not create the destination directory. If a class is part of a package, javac puts the class file in a subdirectory reflecting the package name,

creating directories as needed. For example, if you specify `-d c:\myclasses` and the class is called `com.mypackage.MyClass`, then the class file is called `c:\myclasses\com\mypackage\MyClass.class`.

If not specified, `javac` puts the class file in the same directory as the source file.

Note that the directory specified is automatically added to your user class path.

**Source Files** - One or more source files to be compiled (such as `MyClass.java`).

**JDK/Compiler** - The JDK Config to use. See the JDK Configuration Section in the Options Dialog (under compilers).

**Target** - Generate class files that will work on VMs with the specified version. The default is to generate class files to be compatible with the 1.2 VM in the Java 2 SDK. The versions supported by **javac** in the Java 2 SDK are:

**1.1** Ensure that generated class files will be compatible with 1.1 and VMs in the Java 2 SDK.

**1.2** Generate class files that will run on VMs in the Java 2 SDK, v 1.2 and later, but will not run on 1.1 VMs. This is the default.

**1.3** Generate class files that will run on VMs in the Java 2 SDK, v 1.3 and later, but will not run on 1.1 or 1.2 VMs.

**1.4** Generate class files that are compatible only with 1.4 VMs.

**ClassPath** - Set the user class path, overriding the user class path in the `CLASSPATH` environment variable. If neither `CLASSPATH` or `-classpath` is specified, the user class path consists of the current directory.

**BootClassPath** = Allows cross compilation using the bootstrap and extension classes of a different java platform implementation.

**Extension Directories** - Cross compile against the specified extension directories

**Additional Arguments** - Allows you to pass additional command line arguments to the compiler (ie arguments not exposed as properties of the action).

**JVM Options** - Allows you to specify arguments that will be passed to the `jvm` when it is invoked.

**Generate Debug Info** - Generates Debug info during the compile

Source - Source file debugging information

Lines - Line number debugging information

Vars - Local variable debugging information

**Verbose** - Verbose output. This includes information about each class loaded and each source file compiled

**Assertions** - Enables support for compiling source code containing assertions. When Target is set to 1.4, the compiler accepts code containing assertions. Assertions were

introduced in J2SE 1.4. When Target is set to 1.3, the compiler does not support assertions. Only valid if Assertions option enabled.

**Deprecation** - Show a description of each use or override of a deprecated member or class. Without **-deprecation**, **javac** shows the names of source files that use or override deprecated members or classes.

#### 5.8.1.1 Jikes Compiler Options

#### 5.8.1.2 Borland Compiler Options

The Borland Compiler does not have any extra settings.

#### 5.8.1.3 JDK Configurations

The JDK Configurations enable the Java action to execute using different JDK configurations, and makes it easy to switch between those configurations.

To Add a new JDK Config, open the FinalBuilder Options dialog, navigate to Compilers, JDK Configurations.

Click on the Add button, then fill in the Name field, for example "Sun JDK 1.4.2.0.5". The JDK Home Path field should point to the Root folder where the jdk is installed, for example G:\j2sdk1.4.2\_05

Select the Compiler Type, in this example "Sun Javac 1.4"

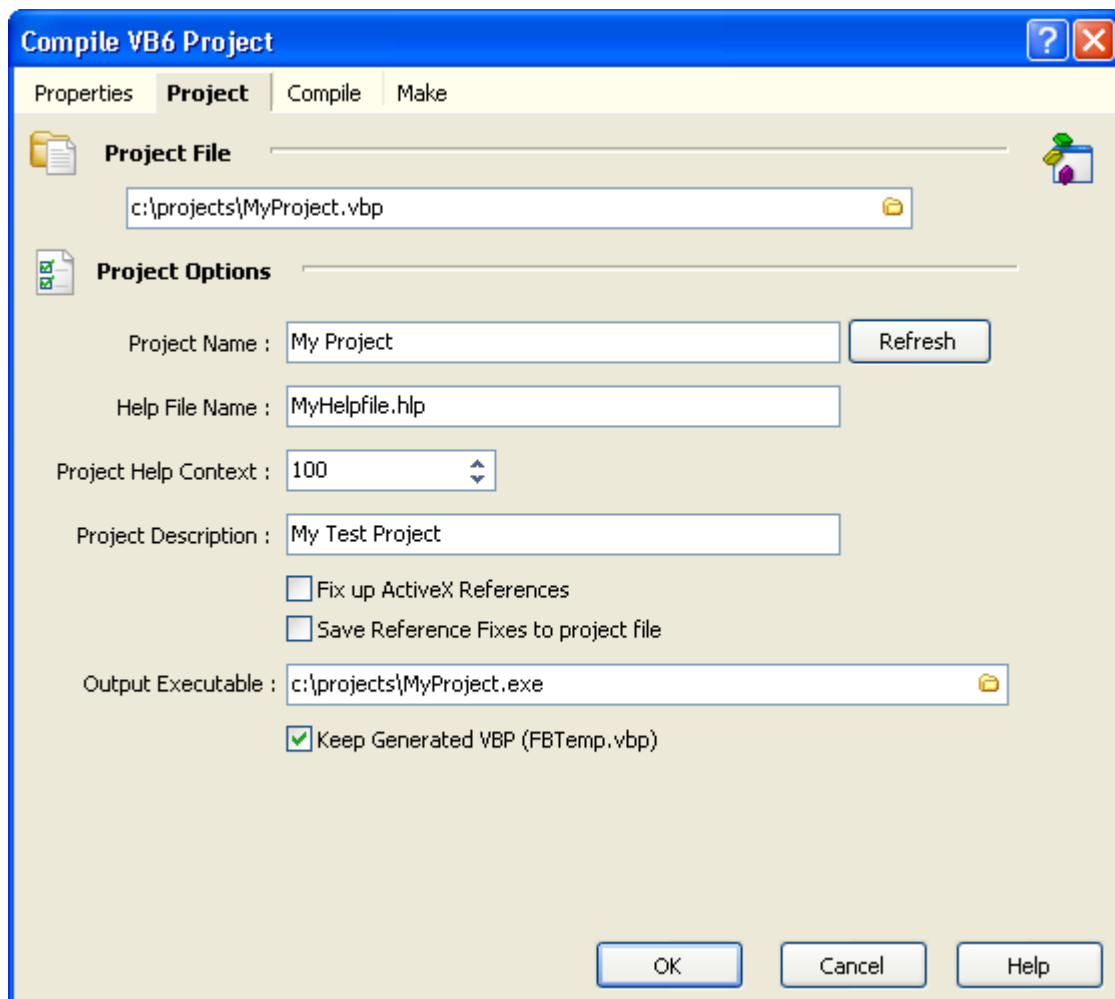
Now we need to add the jdk classpath setting. in the edit box at the bottom of the page, click on the browse button and navigate to the JDK lib folder, eg. G:\j2sdk1.4.2\_05\lib and click on ok. Then click on the Add Button, then close the options dialog.

You should now see the jdk config you just created in the dropdown list on the Java Compiler action's property page. You can create as many configurations as you like. Note that these configs are stored in the registry, if you move your project to another machine you should first define those configurations on the new machine.

### 5.8.2 Microsoft

#### 5.8.2.1 Compile Visual Basic Project

This action provides an interface to the VB6 compiler. This action maintains the Version info and some other settings separate from the visual basic project file (.vbp). Before running the VB compiler, FinalBuilder creates a temporary .vbp file using the settings from the FinalBuilder action and from the actual project.vbp file.



**Project File :** Specifies the path and file name of the Visual Basic project file that you wish to compile.

**Refresh Button :** Refreshes the settings from the project.vbp file. This will overwrite the changes that you made in FinalBuilder.

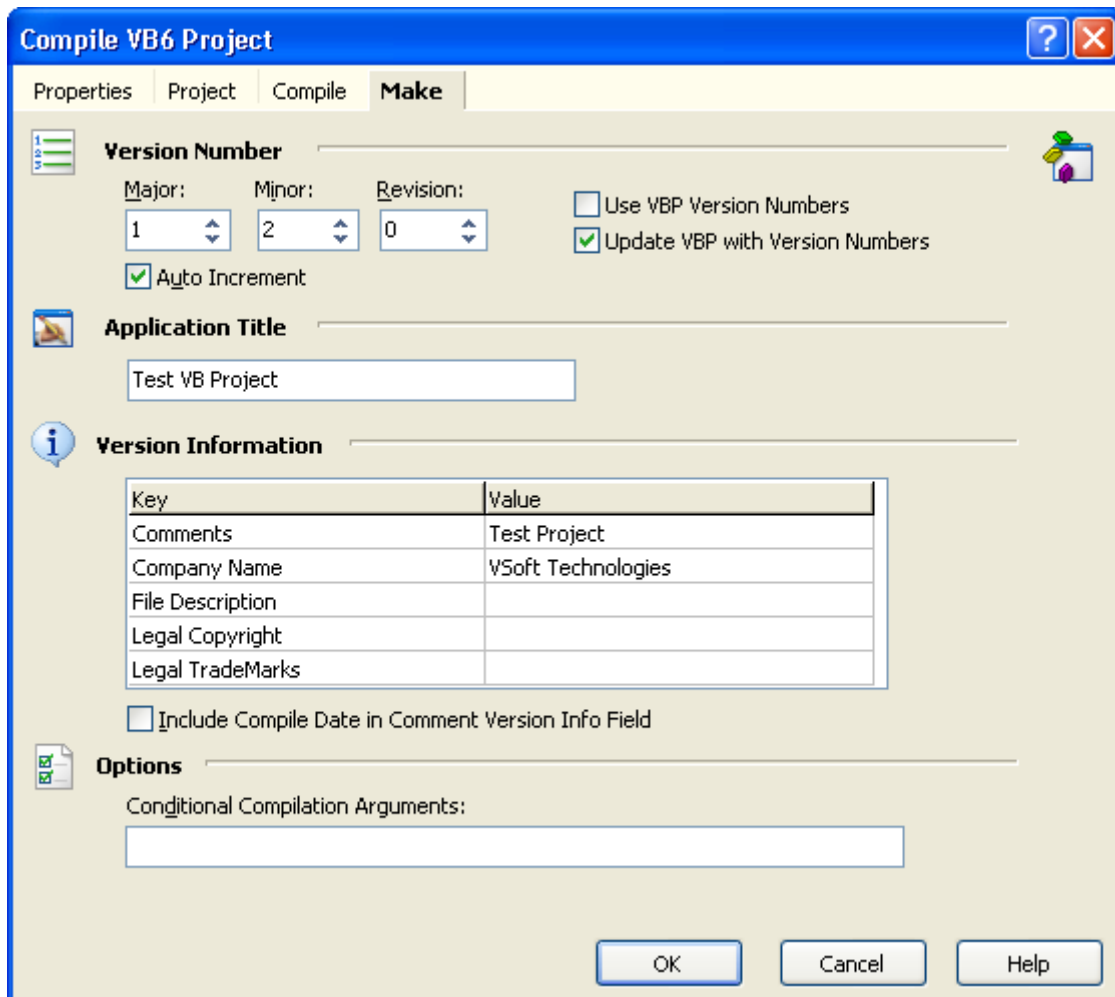
**Project Name :** Identifies the project in code. It can't contain periods (.), spaces, or start with a non-alphabetic character. For a public class name, the project name and class name cannot exceed a total of 37 characters.

**Help File Name :** The name of the Help file associated with the project.

**Help Context ID :** The context ID for the specific Help topic to be called when the user selects the "?" button while the application's object library is selected in the Object Browser.

**Project Description :** A user-friendly name for the project. Displayed in the References and Object Browser dialog boxes.

**Output Executable :** The full path and file name that the project will be compiled to.



Note that if you use the AutoInc Version Number, you need to save the FinalBuilder project after building for this to be saved. Alternatively, use a FinalBuilder Project variable and read the value from an ini file before compiling and then write the variable back out to the ini file after compiling.

#### **Note : Some important info on the "Fix up ActiveX References" setting on the Project Tab**

When using Project or No Compatibility with ActiveX DLL projects, the CLSID's for those projects change with each compile or interface change (depending on the compatibility setting). This will invalidate references to those dll's in other projects. When the Fix up ActiveX References option is checked, FinalBuilder will check that the references are correct, and if not correct them. It does this by locating the type library and extracting the CLSID and comparing it to the reference entry, if they are not the same then the reference entry is corrected. This will enable FinalBuilder to compile the projects correctly, and also help you avoid the dreaded 429 Ole Automation error. See also:

#### **Scripting Info**

The Action properties available are :

**property** ProjectFile : WideString  
**property** OutputPath : WideString  
**property** Conditionals: WideString



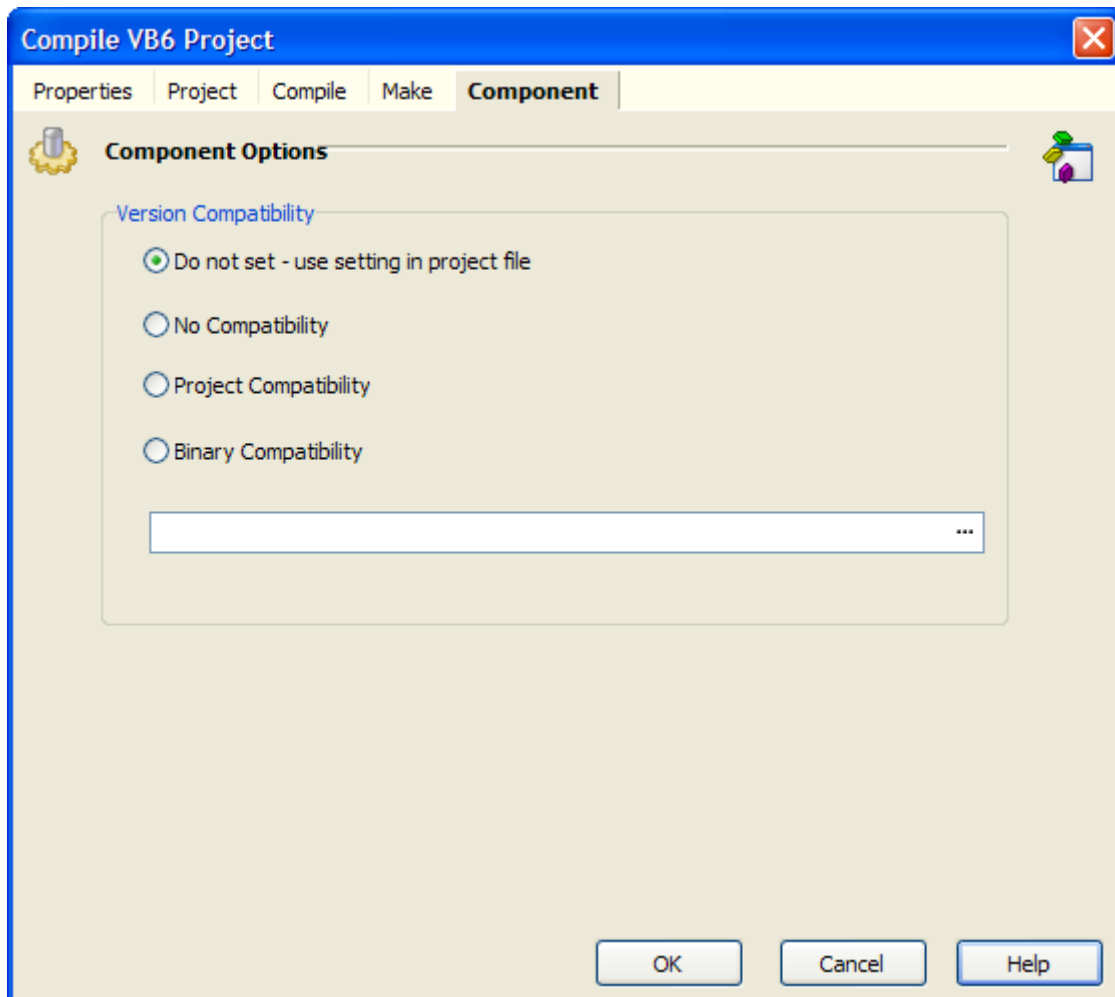
**property** ProjectName: WideString  
**property** ProjectDescription: WideString  
**property** ProjectHelpFile: WideString  
**property** ProjectHelpContextID: integer

**property** Title: WideString  
**property** ExeName: WideString  
**property** MajorVersion: integer  
**property** MinorVersion: integer  
**property** RevisionVersion: integer  
**property** AutoIncrementVersion: WordBool  
**property** VersionCompanyName: WideString  
**property** VersionComments: WideString  
**property** VersionFileDescription: WideString  
**property** VersionLegalCopyright: WideString  
**property** VersionLegalTrademarks: WideString  
**property** VersionProductName: WideString

**property** CompilationType: integer // valid values are ctPCode or ctNativeCode  
**property** OptimizationType: integer //valid values are otNone, otFast or otSmall  
**property** FavorPentiumPro: WordBool  
**property** CodeViewDebugInfo: WordBool  
**property** NoAliasing: WordBool  
**property** RemoveBoundsCheck: WordBool  
**property** RemoveOverflowCheck: WordBool  
**property** RemoveFPointCheck: WordBool  
**property** RemoveFDIVCheck: WordBool  
**property** AllowUnroundedFP: WordBool  
**property** DLLBaseAddress: Integer

#### 5.8.2.1.1 Version Compatibility

VB6 version compatibility.



This page allows you to decide which version compatibility mode you want to use when compiling.

**Do not set** - this option uses whatever compatibility mode has been set in the project file

#### **No Compatibility**

With this setting, new class ID's, new interface ID's and a new type library ID will be generated by VB each time the ActiveX component project is compiled. This will cause any compiled client components to fail (with error 429!) and report a missing reference to the 'VB ActiveX Test Component' when a client project is loaded in the VB IDE.

TIP: Use this setting to compile the initial release of a component to other developers.

#### **Project Compatibility**

With this setting, VB will generate new interface ID's for classes whose interfaces have changed, but will not change the class ID's or the type library ID. This will still cause any compiled client components to fail (with error 429!) but will not report a missing reference to the 'VB ActiveX Test Component' when a client project is loaded in the VB IDE. Recompile of client components will restore them to working order again.

TIP: Use this setting during the initial development and testing of a component within the

IDE and before the component is released to other developers.

### **Binary Compatibility**

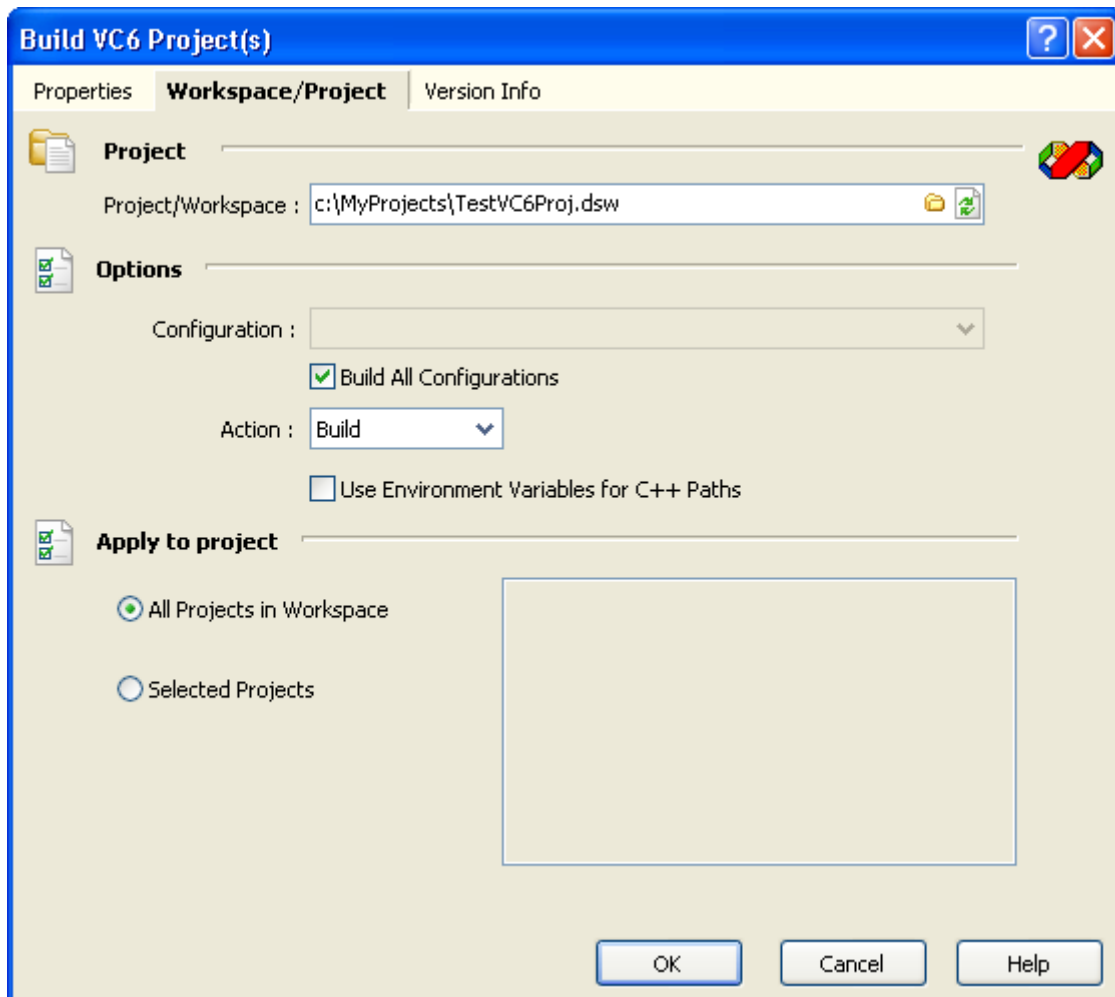
VB makes it possible to extend an existing class or interface by adding new methods and properties etc. and yet still retain binary compatibility. It can do this, because it silently creates a new interface ID for the extended interface and adds registration code to register the original interface ID but with a new Forward key containing the value of this new interface ID. COM will then substitute calls having the old ID with the new ID and hence applications built against the old interface will continue to work (assuming the inner workings of the component remain backward compatible!).

With this setting, VB will not change any of the existing class, interface or type library ID's, however in order that it can do so, VB requires the project to specify an existing compiled version that it can compare against to ensure that existing interfaces have not been broken.

TIP: Use this setting following the release of a component to other developers.

#### **5.8.2.2 Visual C++ 6 Action**

This action provides the ability to build Visual C++ 6 projects and Workspaces. You can choose to build/rebuild/clean all projects in a workspace or selected projects. You can also choose the Configuration to be used (this configuration must exist in all projects in the workspace).



FinalBuilder can also update the version info in the project. It does this by searching the .rc files in the project for a VERSIONINFO structure. At runtime it will backup the original xxxx.rc file to xxxx.rc.orig and generate a new one with the updated version info. You can choose to apply the version info to all projects or selected projects in the workspace (the list of projects is dependant on what is selected on the project/workspace tab). To Auto Increment the build number, make sure the Auto-Increment check box is checked.

**Build VC6 Project(s)**

Properties    Workspace/Project    **Version Info**

☒ Update version information in project

**Module version number**

Major: 1    Minor: 43    Release: 0    Build: 53    ☒ Auto-increment

**Module Attributes**

☐ Debug build    ☐ Special build    ☐ DLL  
☐ Pre-release    ☐ Private build

**Language**

English (United States)    Code Page: 1252    Locale ID: \$0409

**Version Information**

| Key             | Value              |
|-----------------|--------------------|
| Comments        |                    |
| CompanyName     | VSoft Technologies |
| FileDescription |                    |

☒ Include Compile Date in Version Info

**Apply to projects**

☒ All Projects    ☐ Selected Projects

OK    Cancel    Help

### Scripting Info

The Action properties available are :

**property** ProjectFile : string;  
**property** Configuration : string;  
**property** UseEnv : boolean;  
**property** BuildAction : TVC6BuildAction; Valid values are : baBuild,baRebuild and baClean.

The BuildType property determines whether All or selected projects in a workspace are built.

**property** BuildType : TVC6BuildType; Valid values are : btAllProjects and btSelectedProjects

The ConfigType property determines which configurations are built, Selected or All Configurations

**property** ConfigType : TVC6ConfigType; Valid values are : ctSelected,ctAll

**property** SelectedProjects : TStrings; //The selected projects that will be built.

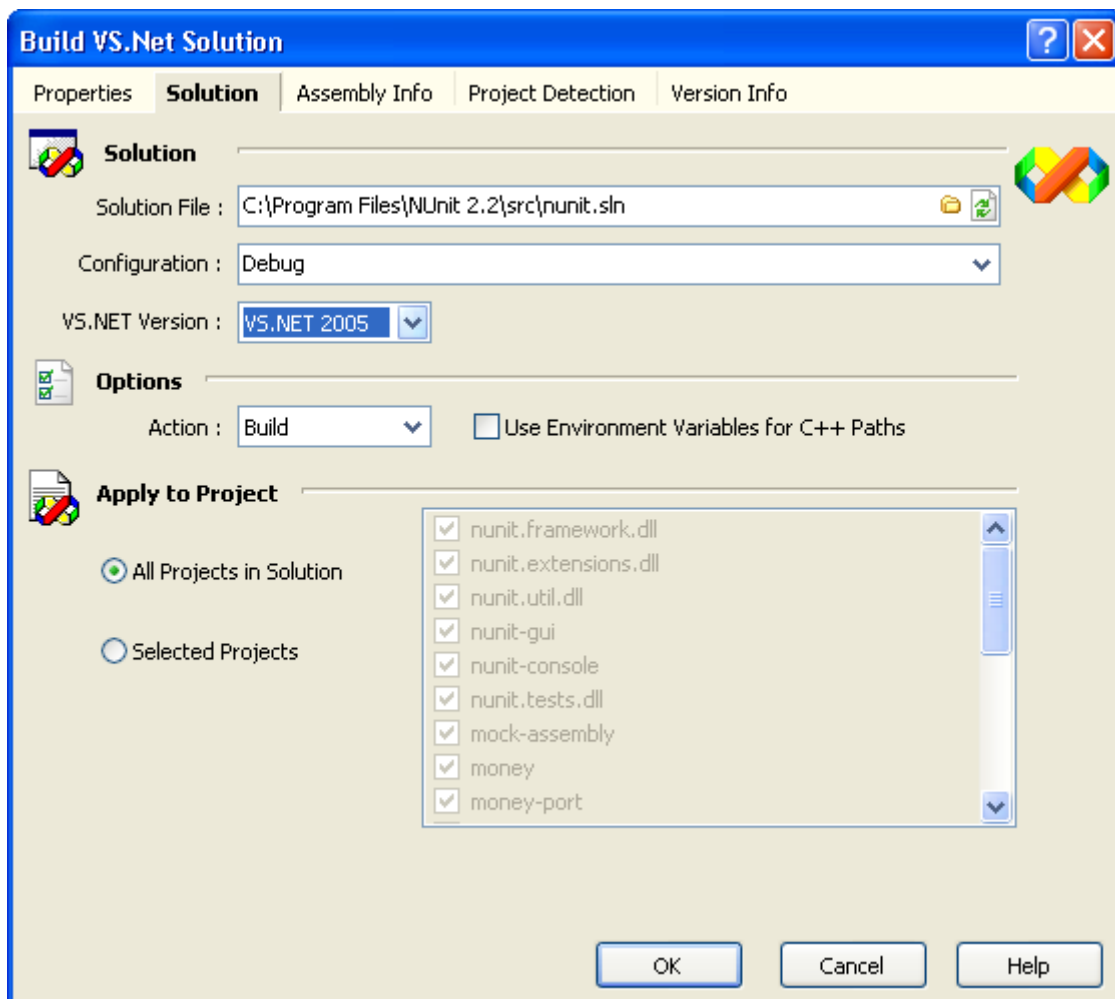
**property** SelectedVI : TStrings; // the projects the version info will be applied to.

**property** VIType : TVC6VIType; Valid Values are :  
 viAllProjects,viSelectedProjects //determines whether version info will be applied to all or  
 selected projects in the workspace.

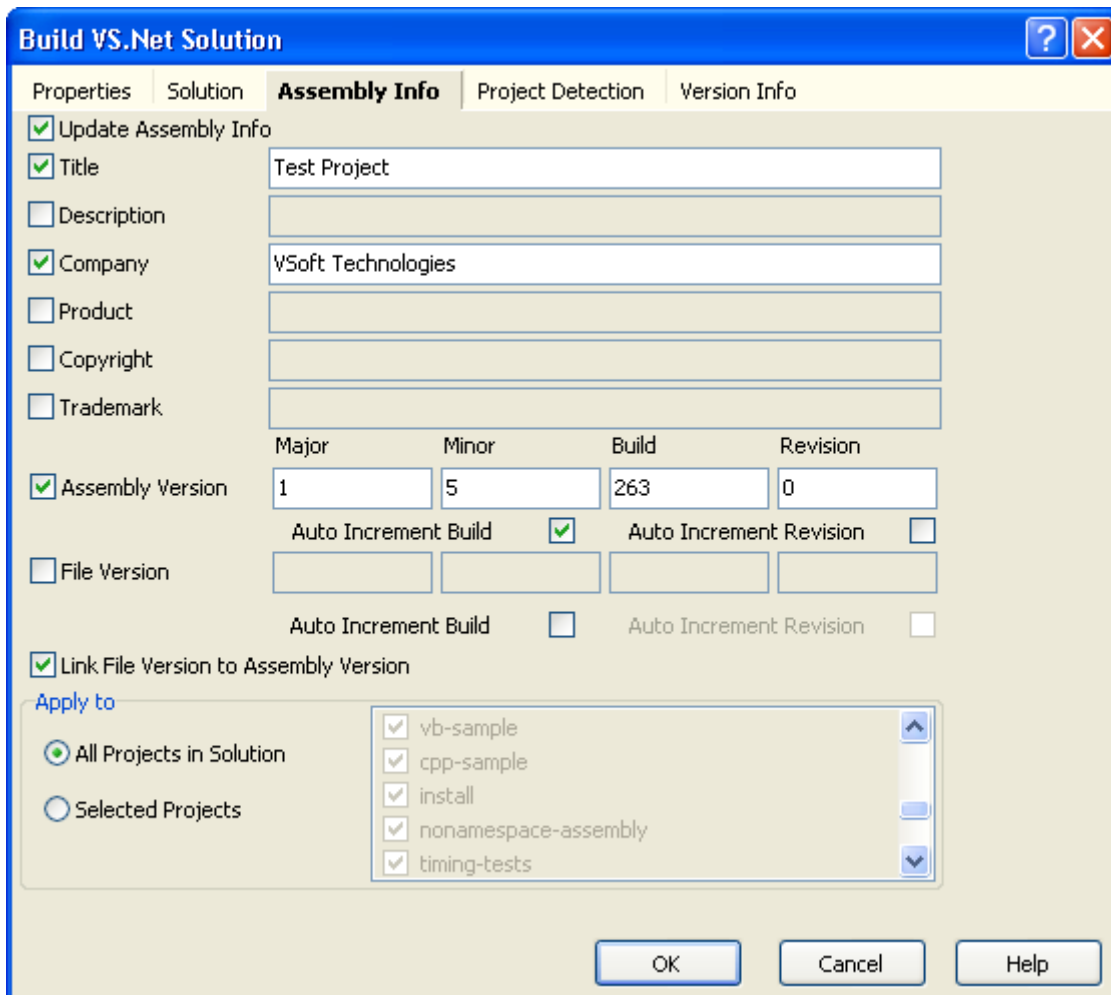
**property** InvokeMacro : string;  
**property** IncludeVerInfo : boolean;  
**property** AutoIncBuild : boolean;  
**property** MajorVersion : integer;  
**property** MinorVersion : integer;  
**property** ReleaseVersion : integer;  
**property** BuildVersion : integer;  
**property** IsDebug : boolean;  
**property** IsPreRelease : boolean;  
**property** IsSpecial : boolean;  
**property** IsPrivate : boolean;  
**property** IsDLL : boolean;  
**property** Locale : integer;  
**property** CodePage : integer;  
**property** VersionInfoKeys : WideString; // in the format :  
     CompanyName=VSoft Technologies Pty Ltd  
     FileDescription=FinalBuilder Core API  
     FileVersion=0.0.1.541  
     InternalName=FBCore10.bpl  
     LegalCopyright=© 2000 - 2003 VSoft Technologies Pty Ltd  
     LegalTrademarks=FinalBuilder™  
     OriginalFilename=FBCore10.bpl  
     ProductName=FinalBuilder™  
     ProductVersion=1.0.0.0  
     Comments=

### 5.8.2.3 Visual Studio .NET Action

This action provides the ability to build Visual Studio .NET solutions using FinalBuilder. You can choose to build, rebuild or clean the whole solution or selected projects in the solution. You can also choose which Solution Configuration to use.



The Assembly Info page provides the facility to update Assembly info in all or selected projects in the solution. Note that this will not work for ASP.NET projects in the solution. Support for updating win32 version info in unmanaged C++ projects will be added in the next update.



The image shows the 'Build VS.Net Solution' dialog box in FinalBuilder 3. The 'Assembly Info' tab is selected. The 'Update Assembly Info' checkbox is checked. The 'Title' is 'Test Project', 'Company' is 'VSoft Technologies', and 'Assembly Version' is '1.5.263.0'. The 'Auto Increment Build' checkbox is checked, and 'Auto Increment Revision' is unchecked. The 'Link File Version to Assembly Version' checkbox is checked. The 'Apply to' section has 'All Projects in Solution' selected. The list of projects includes 'vb-sample', 'cpp-sample', 'install', 'namespace-assembly', and 'timing-tests', all of which are checked.

**Build VS.Net Solution**

Properties | Solution | **Assembly Info** | Project Detection | Version Info

☒ Update Assembly Info

☒ Title: Test Project

☐ Description:

☒ Company: VSoft Technologies

☐ Product:

☐ Copyright:

☐ Trademark:

☒ Assembly Version: Major: 1 Minor: 5 Build: 263 Revision: 0

Auto Increment Build: ☒ Auto Increment Revision: ☐

☐ File Version: Auto Increment Build: ☐ Auto Increment Revision: ☐

☒ Link File Version to Assembly Version

Apply to:

☒ All Projects in Solution

☐ Selected Projects

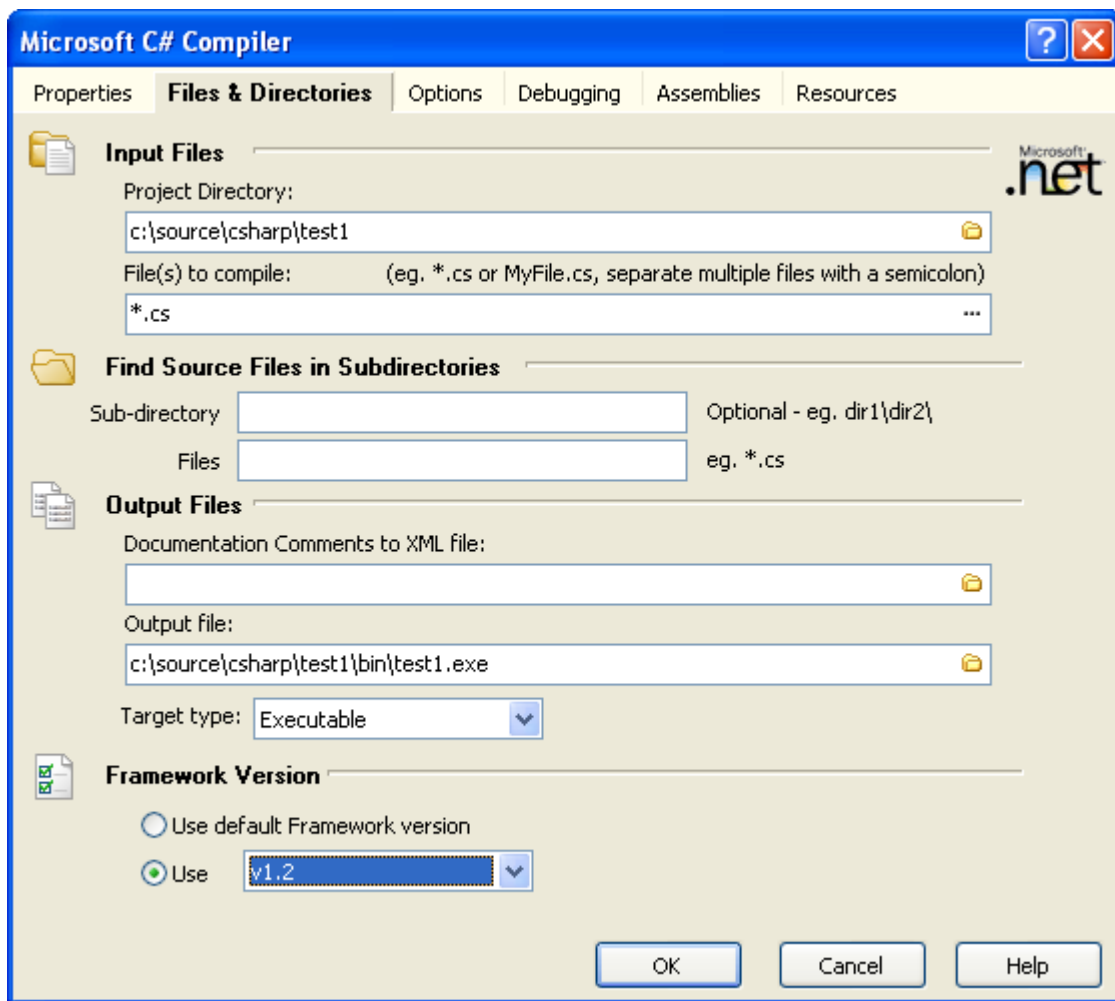
vb-sample  
cpp-sample  
install  
namespace-assembly  
timing-tests

OK Cancel Help

#### 5.8.2.4 Microsoft C# Compiler Action

This action executes the Microsoft .NET Framework C# command line compiler.



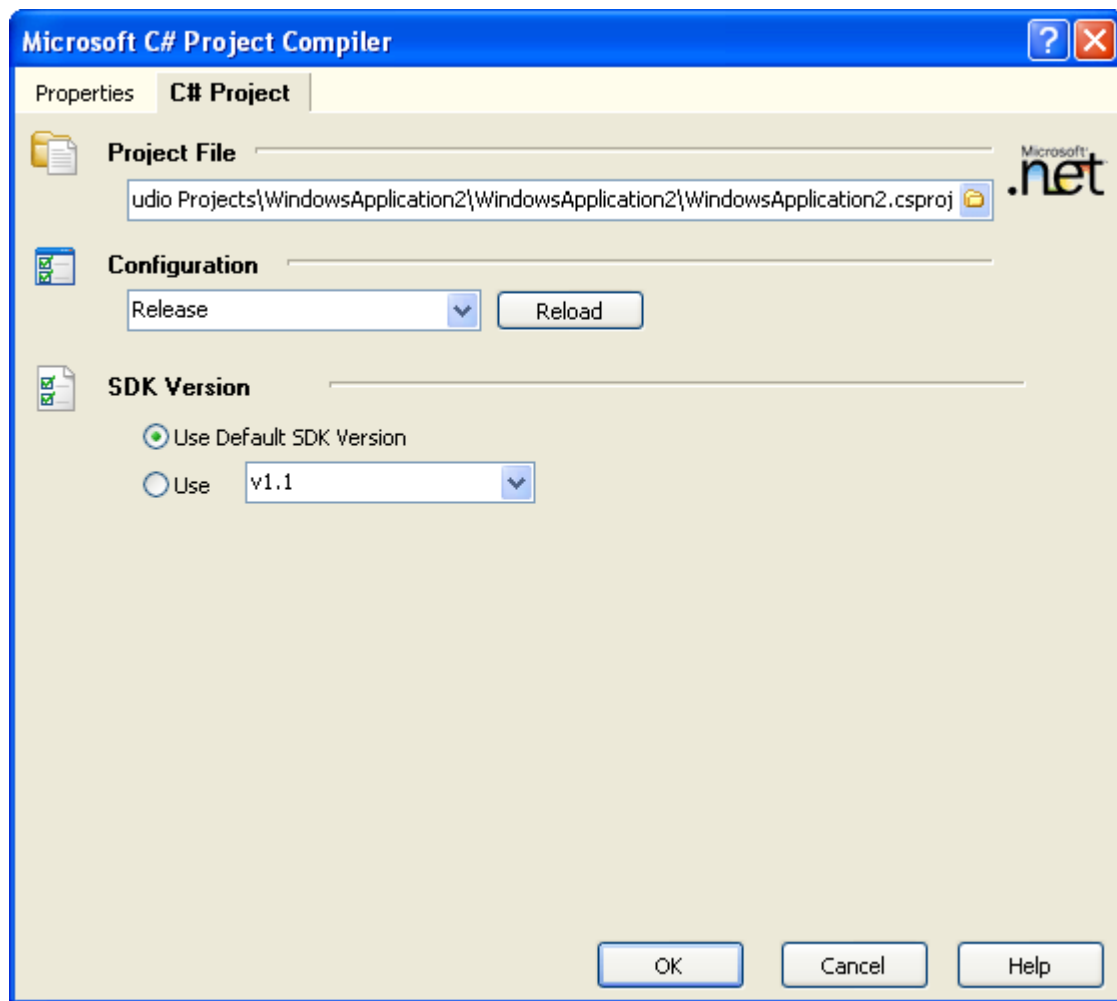


Information on the compiler options can be found in the .NET Framework documentation on MSDN :

<http://msdn.microsoft.com/library/default.asp?url=/library/en-us/cscomp/html/vcrefCompilerOptions.asp>

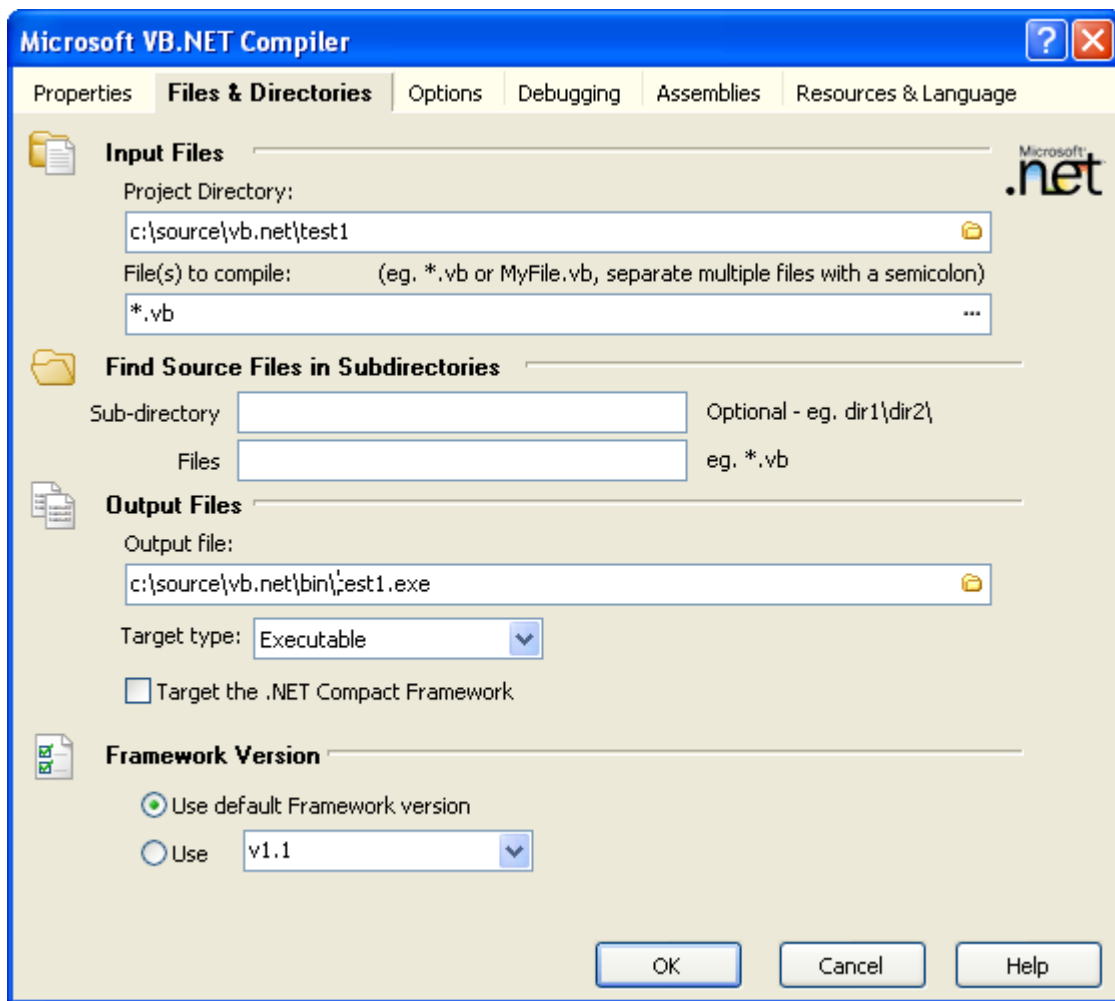
#### 5.8.2.5 Microsoft C# Project Compiler Action

This action enabled you to compile Microsoft C# Projects (.csproj) produced by Visual Studio.NET without having Visual Studio installed on your build machine.



#### 5.8.2.6 Microsoft VB.NET Compiler Action

This action executes the Microsoft .NET Framework VB.NET command line compiler.

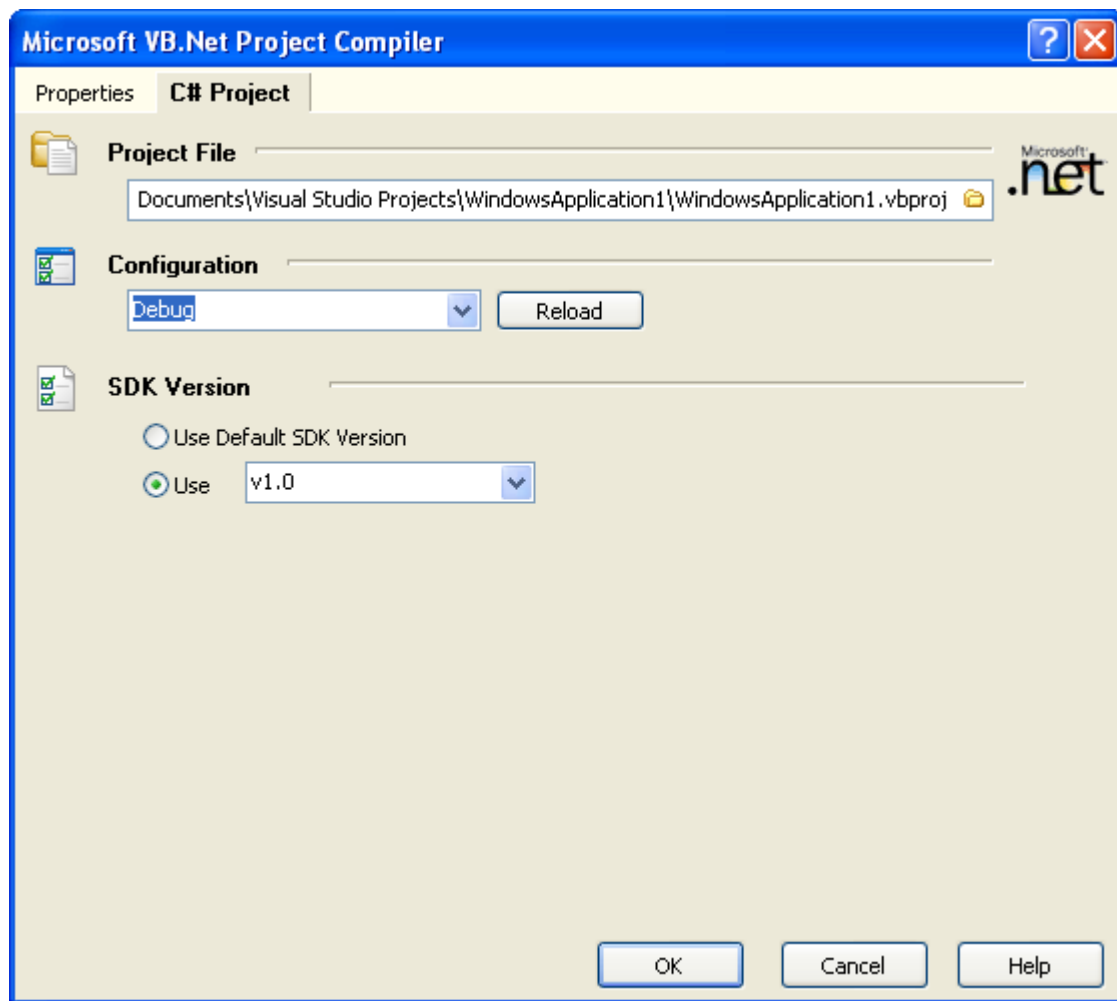


Information on the compiler options can be found in the .NET Framework documentation on MSDN :

<http://msdn.microsoft.com/library/default.asp?url=/library/en-us/vbclr7/html/vaconbuildingfromcommandline.asp>

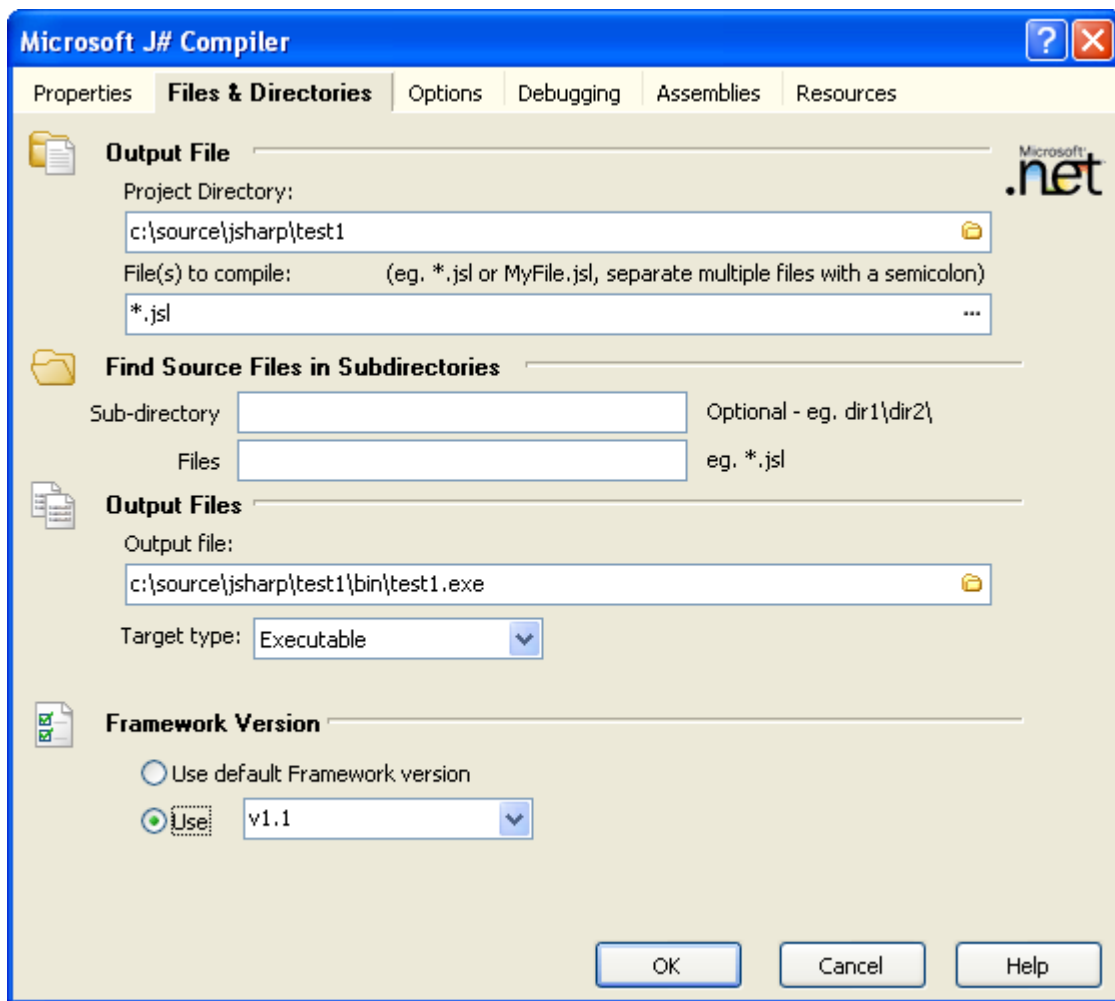
#### 5.8.2.7 Microsoft VB.NET Project Compiler Action

This action enabled you to compile Microsoft VB.NET Projects (.vbproj) produced by Visual Studio.NET without having Visual Studio installed on your build machine.



#### 5.8.2.8 Microsoft J# Compiler Action

This action executes the Microsoft .NET Framework J# command line compiler.

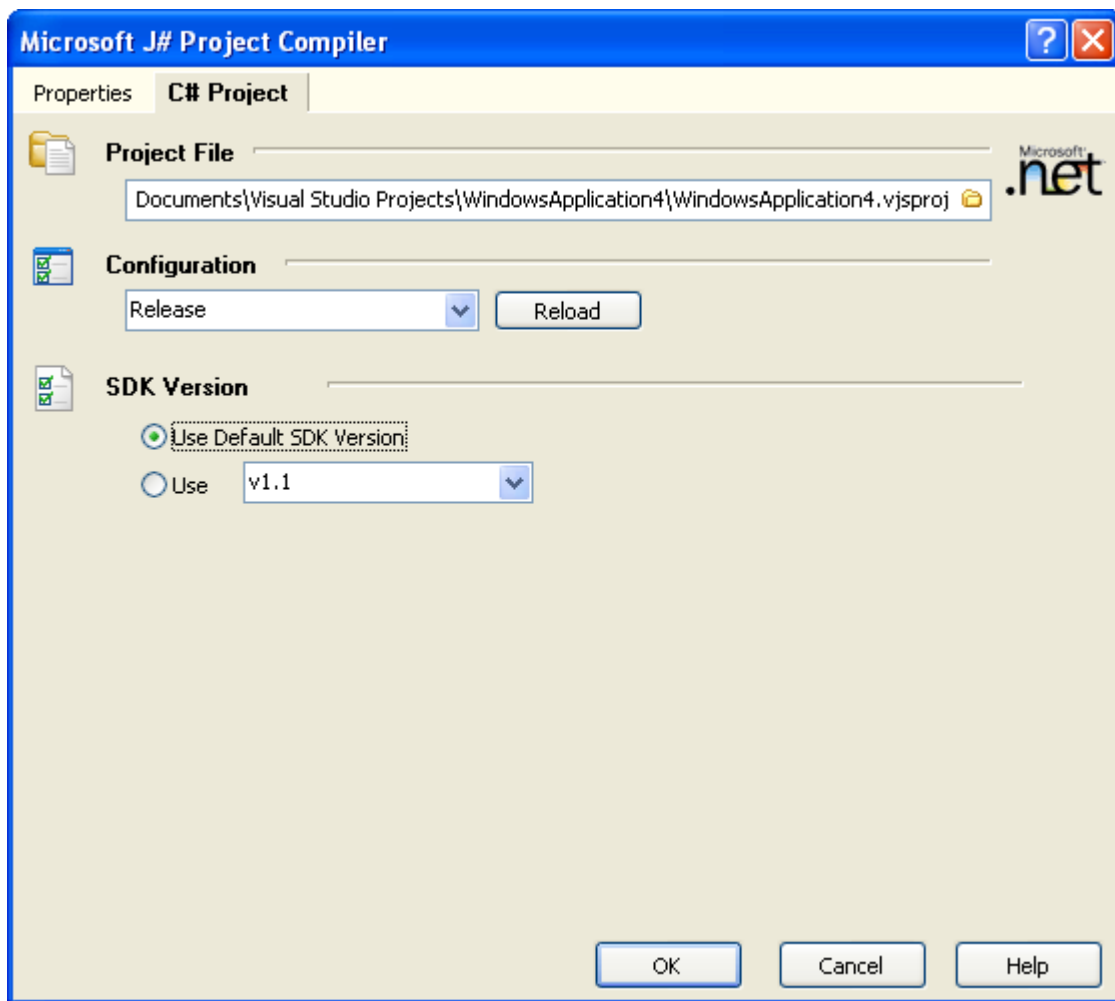


Information on the compiler options can be found in the .NET Framework documentation on MSDN :

[http://msdn.microsoft.com/library/en-us/dv\\_vjsharp/html/vjgrfVisualJCompilerOptions.asp](http://msdn.microsoft.com/library/en-us/dv_vjsharp/html/vjgrfVisualJCompilerOptions.asp)

#### 5.8.2.9 Microsoft J# Project Compiler Action

This action enabled you to compile Microsoft J# Projects (.vjsproj) produced by Visual Studio.NET without having Visual Studio installed on your build machine.

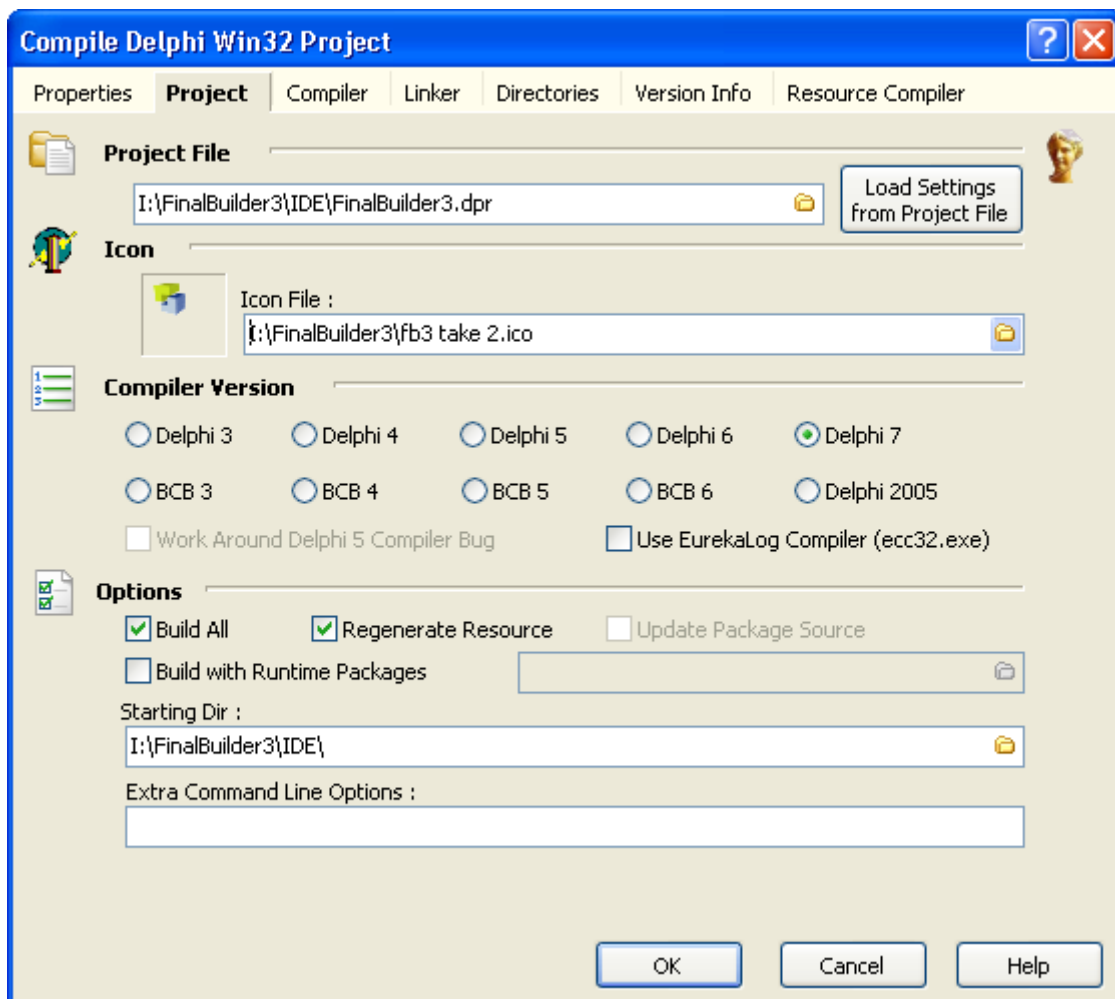


### 5.8.3 Borland

#### 5.8.3.1 Compile Delphi Project

This action provides a familiar interface to the Delphi command line compiler. The options on the Compiler, Linker, Directories and Version Info are described in the Delphi Help File. They provide the same options that are available when compiling a Delphi project from within the Delphi IDE. Note that you may use FinalBuilder Variables in the Directories dialog but no in the version info dialog.

This action can also maintain the version info for the selected Delphi project. When the AutoIncrement property (on the version info tab) is enabled, the version number properties (Major, Minor, Release and build) are persisted to a file (yourproject.fbd) after the action runs (and after the AfterAction script event). These values will be restored when the project is loaded. If the action fails, the values are not persisted. This makes the AutoIncrement property function the same as it does in the Delphi IDE.



The options that are specific to FinalBuilder are in the Project Tab pictured above :

**Project File :** This is the path to the Delphi project, Delphi package or unit file you wish to compile.

**Build All :** Tells the command line compiler to rebuild all files in the project. This is checked by default. When not checked, the command line compiler will only recompile modified units.

**Regenerate Resource :** When checked, FinalBuilder will regenerate the project resource file before compiling the project. It calls the Borland Resource Compiler to do this. If you use version info in your project and increment any version numbers then you should leave this checked.

**Load Settings from DOF File :** The Delphi IDE maintains a file with the extension .dof with the project compiler/linker settings, version info, search path and output directories. Clicking this button will cause FinalBuilder to read this file and apply those settings to this action. You should do this when you first create the action. After that, the settings are maintained separately from the Delphi IDE. If you make changes to your project in the IDE that affect the operation of the program then you should reload these settings in FinalBuilder so that it compiles correctly from FinalBuilder.

**Icon file :** This is the path to the icon that will be used as the main icon for the application. When you select a project file, FinalBuilder checks to see if there is a resource file with the same name (ie the default one generated by the Delphi IDE). If the file exists, FinalBuilder extracts the MAINICON resource and saves it in the same folder as mainicon.ico. This is needed for the resource

compiler.

**Work around Delphi 5 Compiler Bug:** - check this property (which is only enabled when the compiler version is D5) to work around the problem where the project compiles fine but no executable is produced. The will save having to work around it manually by adding a second action.

**NOTE:** You need write access to the directory you are compiling the project from, as FinalBuilder generates temporary .cfg file for the command line compiler to use.

### Scripting Info

The Action properties available are :

```

property CompilerOpt : IDelphiCompilerOptions; // see compiler options section below
property ProjectFile : WideString;
//version info stuff
property IncludeVerInfo : WordBool;
property AutoIncBuild : WordBool;
property MajorVersion : integer;
property MinorVersion : integer;
property ReleaseVersion : integer;
property BuildVersion : integer;
property IsDebug : WordBool;
property IsPreRelease : WordBool;
property IsSpecial : WordBool;
property IsPrivate : WordBool;
property IsDLL : WordBool;
property Locale : integer;
property CodePage : integer
property VersionInfoKeys : WideString; // in the format :
    CompanyName=VSoft Technologies Py Ltd
    FileDescription=FinalBuilder Core API
    FileVersion=0.0.1.541
    InternalName=FBCore10.bpl
    LegalCopyright=© 2000 - 2001 VSoft Technologies Pty Ltd
    LegalTrademarks=FinalBuilder™
    OriginalFilename=FBCore10.bpl
    ProductName=FinalBuilder™
    ProductVersion=1.0.0.0
    Comments=

property BuildAll : WordBool;
property CompilerVersion : integer; // D3 = 0, D4 = 1, D5 = 2, D6 = 3 D7 = 4
property ExtraCommandline : WideString;;
property IconFile : WideString;
property RegenerateResource : WordBool;
property WarningsAsError : WordBool;
property HintsAsError : WordBool;
property Packages : WideString
property UsePackages : WordBool

```

Compiler Options

```

property Optimisation : WordBool;
property AlignFields : WordBool;
property StackFrames : WordBool;

```



**property** SafeDivide : WordBool;  
**property** VarStringChecks : WordBool;  
**property** BoolEval : WordBool  
**property** ExtendedSyntax : WordBool  
**property** TypedPointers : WordBool  
**property** OpenStrings : WordBool  
**property** HugeStrings : WordBool  
**property** AssignableConst : WordBool  
**property** RangeChecking : WordBool  
**property** IOChecking : WordBool  
**property** OverflowChecking : WordBool  
**property** DebugInfo : WordBool  
**property** LocalSymbols : WordBool  
**property** ReferenceInfo : WordBool  
**property** DefinitionsOnly : WordBool  
**property** Assertions : WordBool  
**property** UseDebugDCU : WordBool  
**property** ShowHints : WordBool  
**property** ShowWarnings : WordBool  
**property** Mapfile : integer // valid values are : mfNone, mfSegments, mfPublics, mfDetailed  
**property** ConsoleApp : WordBool  
**property** IncludeTD32 : WordBool  
**property** IncludeRemoteSymbols : WordBool  
**property** LinkerOutput : integer; // valid values are loGenerateDCU, loGenerateCOBJ,  
loGenerateCPlusObj  
**property** IncludeNamespaces : WordBool  
**property** ExportAllSymbols : WordBool  
**property** MinStackSize : Cardinal  
**property** MaxStackSize : Cardinal  
**property** ImageBase : Cardinal  
**property** ExeDescription : WideString  
**property** OutputDir : WideString  
**property** UnitOutputDir : WideString  
**property** BPLOutputDir : WideString  
**property** DCPOutputDir : WideString  
**property** Conditionals : WideString  
**property** SearchPath : WideString  
**property** LibraryPath : WideString  
**property** UnitAliases : WideString

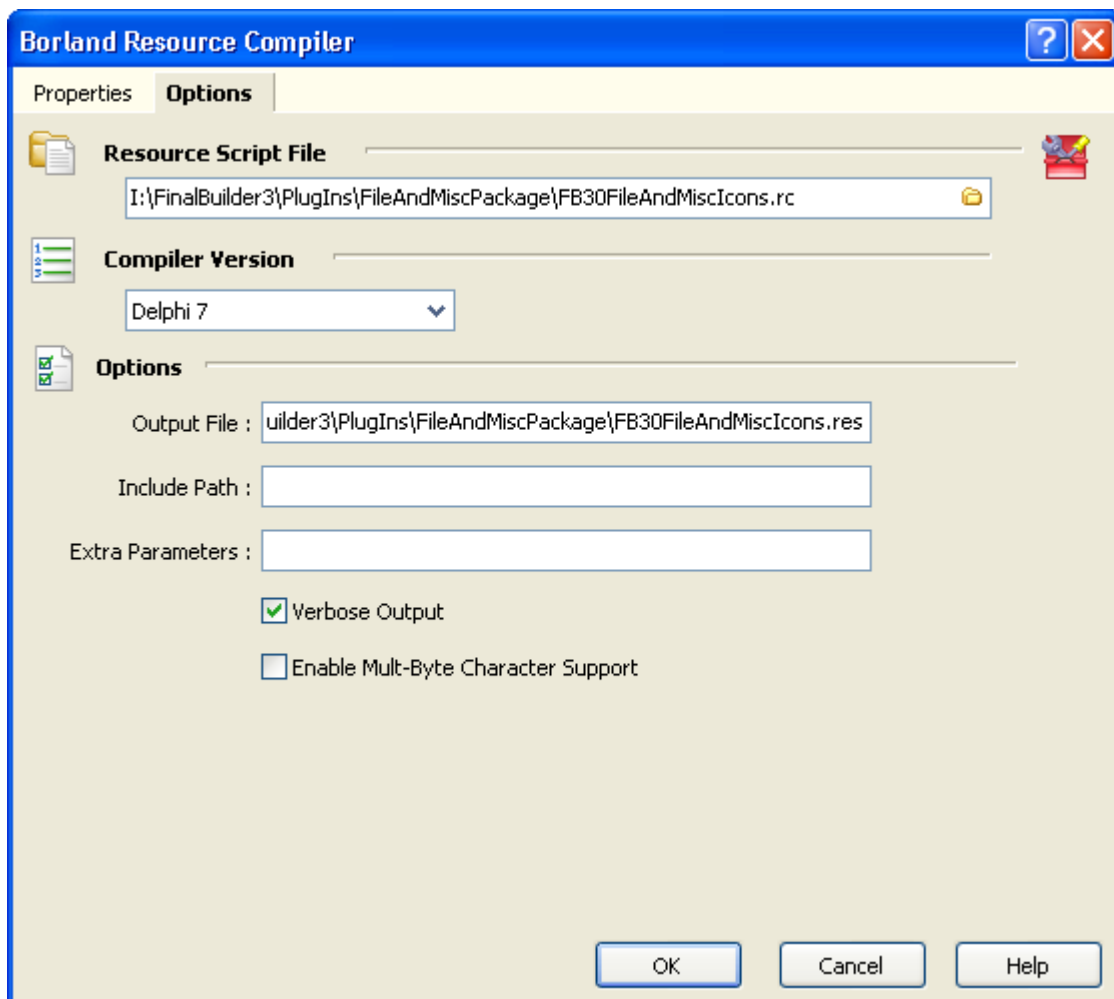
Action methods :

**procedure** SetVersionInfoKey(name : WideString; Value : WideString);

**function** GetVersionInfoKey(name : WideString) : WideString;

### 5.8.3.2 Compile Borland Resource Script

This action provides an easy to use interface for the Borland Resource Compiler.



**Resource Script :** The .rc file that you wish to compile.

**Output File :** The path to the Output .res File.

**Include Path :** The path to use for resource files included with the #include command

**Compiler Version :** The Borland Resource compiler is provided as part of Delphi And C++Builder. This option allows you to choose which version of dcc32 to use. Note that it includes Delphi 7 and C++Builder 6 & 7 which do not yet exist! This is for future use only!

**Verbose Output :** Tells the resource compile to report its compile status for each line. Otherwise, it will only report errors.

**Enable Multi-Byte Character Sets :** Enable multi-byte character support

### Scripting Info

The Action properties available are :

```
property ScriptFile : WideString;
property OutputFileName : WideString;
property VerboseOutput : WordBool;
```

**property** CompilerVersion : integer; // the valid values are :

- 0 = Delphi 3
- 1 = Delphi 4
- 2 = Delphi 5
- 3 = Delphi 6
- 4 = Delphi 7
- 5 = BCB 3
- 6 = BCB 4
- 7 = BCB 5
- 8 = BCB 6
- 9 = BCB 7

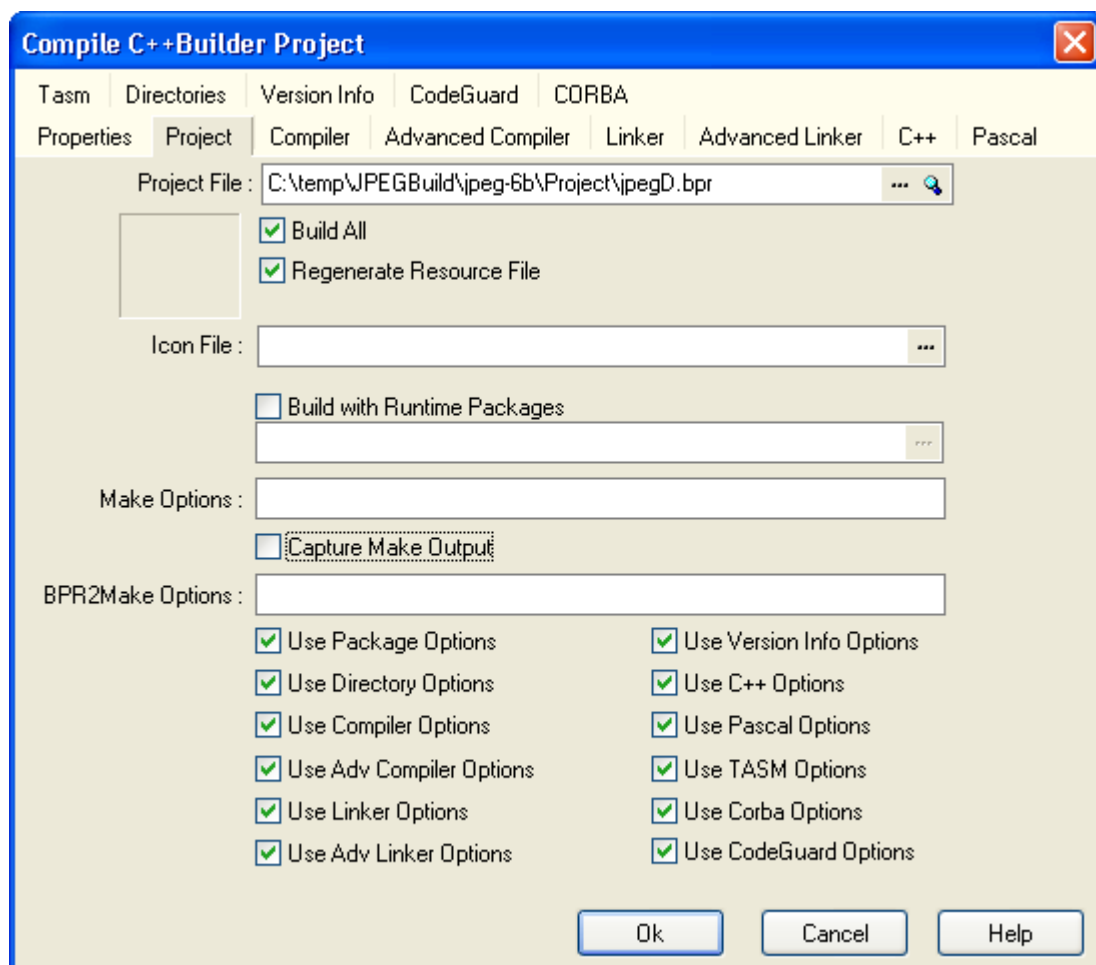
**property** IncludePath : WideString;

**property** EnableMBCS : WordBool;

### 5.8.3.3 Borland C++ Builder Action

This action provides support for Borland's C++ Builder versions 5 & 6. The action allows you to selectively override the settings in the project file with the settings in FinalBuilder. It provides the same interface as the Project Options dialog in BCB.

**Note:** there are some Known Problems with this action.



This action also allows you to generate Version Info for the project, and auto increment

build numbers.

The way the action works is this :

1) In the BCB Action's properties dialog, select a BCB 5 or 6 project. FinalBuilder will read the settings from the project file.

2) When you run the action, it will again read in the project file, and then apply the settings from the Action in FinalBuilder.

You can choose which settings from FinalBuilder are used on the project tab, so for example you can chose to use the compiler & linker settings from the project file and the directories & version info settings from the action.

3) If you have FinalBuilder generate version info, it will save a .rc file and invoke brcc32

4) FinalBuilder saves a temporary copy of the project file, with the merged settings, then calls BPR2MAK and the Make.

Borland C++Builder Scripting Reference

#### 5.8.3.3.1 Borland C++Builder Scripting Reference

##### Scripting Info

The Action properties available are :

**procedure** SetupRelease; //same as the Release button on the compiler options page

**procedure** SetupDebug; //same as the Debug button on the compiler options page

**procedure** SetVersionInfoKey(name : WideString; Value : WideString);

**function** GetVersionInfoKey(name : WideString) : WideString;

**property** ProjectFile : string;  
**property** IncludePath : string;  
**property** LibraryPath : string;  
**property** IntermediateOutputPath : string;  
**property** FinalOutputPath : string;  
**property** BPILIBOutputPath : string;  
**property** Packages : string;  
**property** UsePackages : boolean;  
**property** MakeOptions : string;  
**property** BuildAll : boolean;  
**property** Conditionals : string;  
**property** BPR2MakOptions : string;  
**property** RegenerateResource: boolean;  
**property** IncludeVerInfo: boolean;  
**property** MajorVersion: integer;  
**property** MinorVersion: integer;  
**property** ReleaseVersion: integer;  
**property** BuildVersion: integer;  
**property** AutoIncBuild: boolean;  
**property** IsDebug: boolean;

**property** IsPreRelease: boolean;  
**property** IsPrivate: boolean;  
**property** IsSpecial: boolean;  
**property** IsDLL: boolean;  
**property** VersionInfoKeys: string;

**property** Locale: integer;  
**property** CodePage: integer;  
**property** IncludeCompileDate: boolean;  
**property** IconFile: string;  
**property** CompilerOpt : OleVariant;  
**property** AdvCompilerOpt : OleVariant;

**property** LinkerOpt : OleVariant;  
**property** AdvLinkerOpt : OleVariant;  
**property** CppOpt : OleVariant;  
**property** PascalOpt : OleVariant;  
**property** TASMOpt : OleVariant;  
**property** CorbaOpt : OleVariant;  
**property** CodeGuardOpt : OleVariant;  
**property** TLibOpt : OleVariant read GetTLibOptions;

**property** CaptureMakeOutput : boolean;

**property** UseRegForBPLPath : boolean;  
**property** UseRegForBPIPath : boolean ;

**property** UseDirectorySettings: boolean;  
**property** UseCompilerSettings: boolean;  
**property** UseAdvCompilerSettings: boolean;  
**property** UseTLibSettings : boolean;  
**property** UseLinkerSettings: boolean;  
**property** UseAdvLinkerSettings: boolean;  
**property** UseCPPSettings: boolean;  
**property** UsePascalSettings: boolean;  
**property** UseVersionInfo: boolean;  
**property** UseTASMSettings: boolean;  
**property** UseCorbaSettings: boolean;  
**property** UsePackageSettings: boolean;  
**property** UseCodeGuardSettings: boolean;

Borland C++ Builder Compiler Options Scripting Reference

Borland C++ Builder Linker Options Scripting Reference

Borland C++ Builder Advanced Compiler Options Scripting Reference

Borland C++ Builder Advanced Linker Options Scripting Reference

Borland C++ Builder TLib Options Scripting Reference

Borland C++ Builder CPP Options Scripting Reference

Borland C++ Builder Pascal Options Scripting Reference

Borland C++ Builder TASM Options Scripting Reference

Borland C++ Builder Corba Options Scripting Reference

Borland C++ Builder CodeGuard Options Scripting Reference

#### 5.8.3.3.2 Borland C++Builder Compiler Options

##### Scripting Info

The Action.CompilerOpt properties available are :

**property** Optimisation : TBCBCompilerOptimisations;  
Valid values are combinations of :  
coPentiumScheduling,coInlineIntrinsic,coInductionVariables,coCommonSubExpression

**property** OutputWarnings : TBCBCompilerWarnings;  
valid values are :cwNone,cwAll,cwSelected

**property** DebugInfo : boolean;

**property** LineNoInfo : boolean;

**property** DisableInline: boolean;

**property** PreCompiledHeaders : TBCBCompilerPreCompiledHeader;  
Valid Values are pcNone,pcUse,pcCache

**property** PreCompFile : string;

**property** StopAfterFile : string;

**property** MergeDupStrings : boolean;

**property** StackFrames : boolean;

**property** EnumsAsInt : boolean;

**property** ShowGeneralMessages : boolean;

**property** ExtendedErrorInfo : boolean;

#### 5.8.3.3.3 Borland C++Builder Advanced Compiler Options

##### Scripting Info

The Action.AdvCompOpt properties available are :

**property** InstructionSet : TCBuilderInstructionSet;  
Valid values are : is386,is486,isPentium,isPentiumPro

**property** DataAlignment : TCBuilderDataAlignment;  
Valid values are : daByte,daWord,daDoubleWord,daQuadWord

**property** CallingConvention : TCBuilderCallingConvention;  
Valid values are : ccC,ccPascal,ccRegister,ccStdCall

**property** RegisterVariables : TCBuilderRegisterVariables;

Valid values are : rvNone,rvAuto,rvKeyword

**property** OutputAutoDepInfo: boolean;  
**property** GenerateUnderscores: boolean;  
**property** FloatingPointNone : boolean;  
**property** FloatingPointFast : boolean;  
**property** FloatingPointFixFDIV : boolean;  
**property** LanguageCompliance : TCBUILDERLanguageCompliance;  
Valid values are : lcBorland,lcANSI,lcUnixV,lcKR

**property** NestedComments : boolean;  
**property** MFCCompatibility : boolean;  
**property** IdentifierLength : integer;

#### 5.8.3.3.4 Borland C++Builder Linker Options

##### Scripting Info

The Action.LinkerOpt properties available are :

**property** DebugInfo : boolean;  
**property** UseDynamicRTL: boolean;  
**property** UseDebugLibraries: boolean;  
**property** GenerateImportLib: boolean;  
**property** GenerateLibFile: boolean;  
**property** DontGenerateStateFile: boolean;  
**property** ShowMangledNames: boolean;  
**property** MaxErrors: integer;  
**property** MapFile : TCBUILDERMapFile;  
Valid values are : cmOff,cmSegments,cmPublics,cmDetailed

**property** LinkerWarnings : TCBUILDERLinkerWarnings;  
Valid Values are : lwAll,lwSelected

**property** MaxStackSize: Cardinal;  
**property** MinStackSize: Cardinal;  
**property** MinHeapSize: Cardinal;  
**property** MaxHeapSize: Cardinal;  
**property** ImageBase: Cardinal;  
**property** SubSystemMinor : Cardinal;  
**property** SubSystemMajor : Cardinal;

#### 5.8.3.3.5 Borland C++Builder Advanced Linker Options

##### Scripting Info

The Action.AdvLinkerOpt properties available are :

**property** CaseInsensitiveLink: boolean;  
**property** CalcChecksum: boolean;  
**property** ReplaceResources: boolean;  
**property** UserMajorVersion: integer;  
**property** UserMinorVersion: integer;  
**property** ImageComment : string;

**property** DelayLoadDLLS : string;

#### 5.8.3.3.6 Borland C++Builder CPP Options

##### Scripting Info

The Action.CppOpt properties available are :

**property** MemberPointers : TCBuilderMemberPointers;

Valid values are :

mpAllCases,mpMultipleInheritance,mpSingleInheritance,mpSmallest

**property** HonourMemberPrecision: boolean;

**property** DontRestrictForLoop: boolean;

**property** DontMangleCodeModifiers: boolean;

**property** ZeroLengthEmptyBaseClasses: boolean;

**property** ZeroLengthEmptyClassMembers: boolean;

**property** ExternalTemplates: boolean;

**property** EnabledRTTI: boolean read;

**property** EnabledExceptions: boolean;

**property** LocalInformation: boolean

**property** DestructorCleanup: boolean;

**property** FastExceptionPrologs: boolean;

**property** VirtualTables : TCBuilderVirtualTables;

Valid values are : vtSmart,vtLocal,vtExternal,vtPublic

**property** ForceCPPCompile: boolean

#### 5.8.3.3.7 Borland C++Builder TASM Options

##### Scripting Info

The Action.TASMOpt properties available are :

**property** DebugInfo : TCBuilderTASMDebugInfo;

Valid values are : tdNone,tdLineNumbers,tdFull

**property** CaseSensitivity : TCBuilderTASMCASESensitivity;

Valid values are : csNone,csGlobals,csAll

**property** Warnings : TCBuilderTASMWarnings;

Valid values are : twNone,twLevel1,twLevel2

**property** HashTableCapacity : Cardinal read;

**property** MaxPasses : Cardinal rea;

**property** MaxSymbolLength : Cardinal;

**property** GenerateListing: boolean;

**property** ListingType: TCBuilderTASMListingType;

Valid values are : ltNormal,ltExpanded

**property** CrossReference: boolean read;

**property** SymbolTablesInfo: boolean;

**property** IncludeErrorMessages: boolean;

**property** IncludeFalseConditionals: boolean;



**property** AddDirective : string;

#### 5.8.3.3.8 Borland C++Builder TLib Options

##### Scripting Info

The Action.TLibOpt properties available are :

**property** CaseSensitive : boolean;  
**property** CreateExtendedDictionary : boolean;  
**property** PurgeComments : boolean;  
**property** PageSize : integer;  
**property** ListFile : string;  
**property** UseRTL : boolean;

#### 5.8.3.3.9 Borland C++Builder Pascal Options

##### Scripting Info

The Action.PascalOpt properties available are :

**property** Optimisation : boolean;  
**property** AlignFields : TCBUILDERPascalFieldAlign;  
Valid values are : fa1,fa2,fa4,fa8

**property** StackFrames : boolean;  
**property** SafeDivide : boolean;

**property** VarStringChecks : boolean;  
**property** BoolEval : boolean;  
**property** ExtendedSyntax : boolean;  
**property** TypedPointers : boolean;  
**property** OpenStrings : boolean;  
**property** HugeStrings : boolean;  
**property** AssignableConst : boolean;

**property** RangeChecking : boolean;  
**property** IOChecking : boolean;  
**property** OverflowChecking : boolean;  
**property** DebugInfo : boolean;  
**property** LocalSymbols : boolean;  
**property** ReferenceInfo : boolean;  
**property** DefinitionsOnly : boolean;  
**property** Assertions : boolean;

**property** ShowHints : boolean;  
**property** ShowWarnings : boolean;

#### 5.8.3.3.10 Borland C++Builder Corba Options

##### Scripting Info

The Action.CorbaOpt properties available are :

**property** GenerateObjectWrappers: boolean;  
**property** GenerateTie: boolean read;  
**property** GenerateVirtualImplInh: boolean;  
**property** GenerateTypecodeInfo: boolean;  
**property** GenerateIncludeFilesCode: boolean;  
**property** GenerateStreamOperators: boolean;  
**property** HeaderExtension: string;  
**property** AdditionalIDLOptions: string;  
**property** IDLAddServerUnit: boolean;  
**property** IDLAddClientUnit: boolean;  
**property** IDLPrecompiledHeaders: boolean;  
**property** CorbaIncludeIR: boolean;  
**property** CorbaIncludeDSI: boolean;

#### 5.8.3.3.11 Borland C++Builder CodeGuard Options

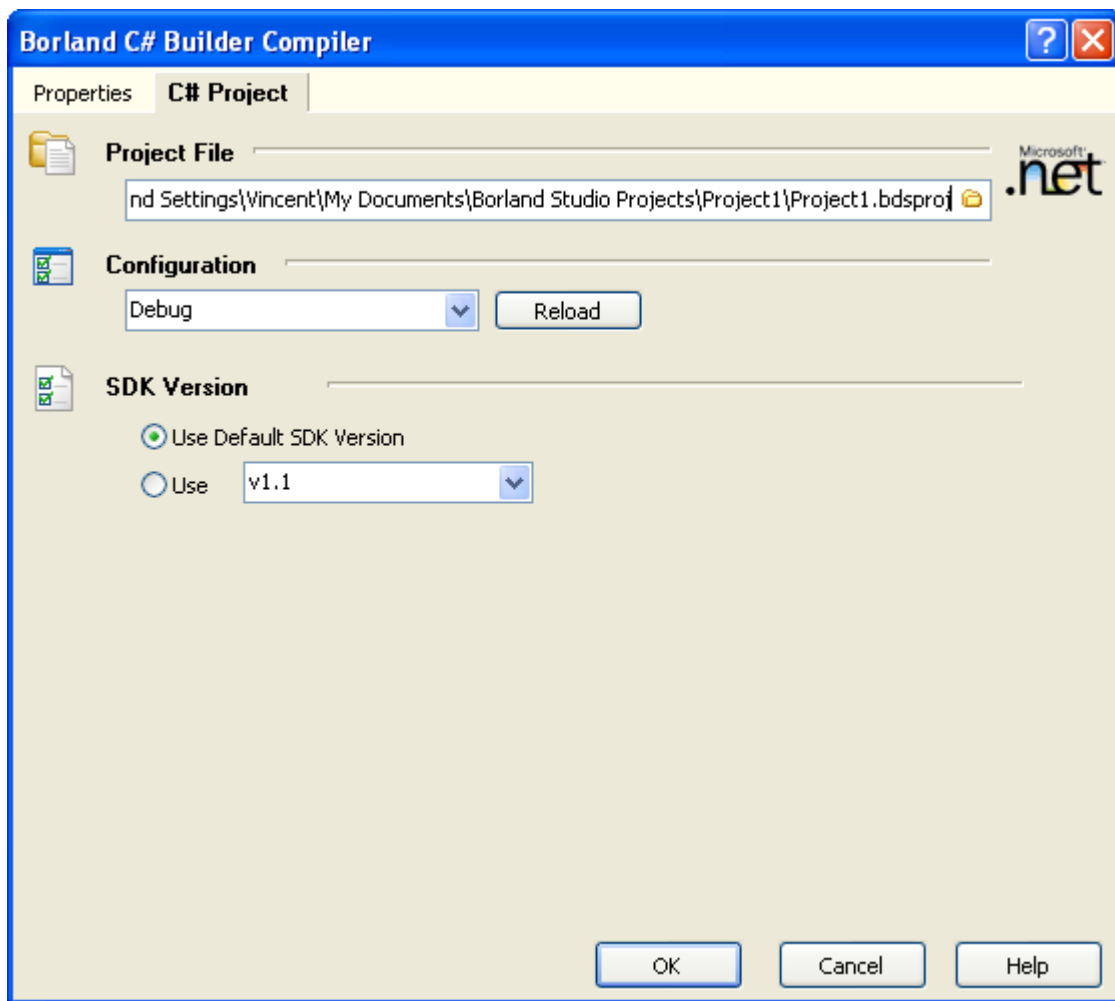
##### **Scripting Info**

The Action.CodeGuardOpt properties available are :

**property** CodeGuardValidation: boolean;  
**property** ValidateGlobalAndStackAccess: boolean;  
**property** ValidateThisPointer: boolean;  
**property** ValidatePointerAccesses: boolean;

#### 5.8.3.4 Borland C# Builder Project Compiler Action

This action will build Borland C# Builder 1.x projects (.bdsproj) without requiring that C#Builder be installed on your build machine.



#### 5.8.3.5 Borland Delphi 8 for .NET Action

This action will build Borland Delphi™ 8 for .NET projects (.bdsproj). Delphi 8 is significantly different from earlier versions of Delphi, thus requiring a separate action.

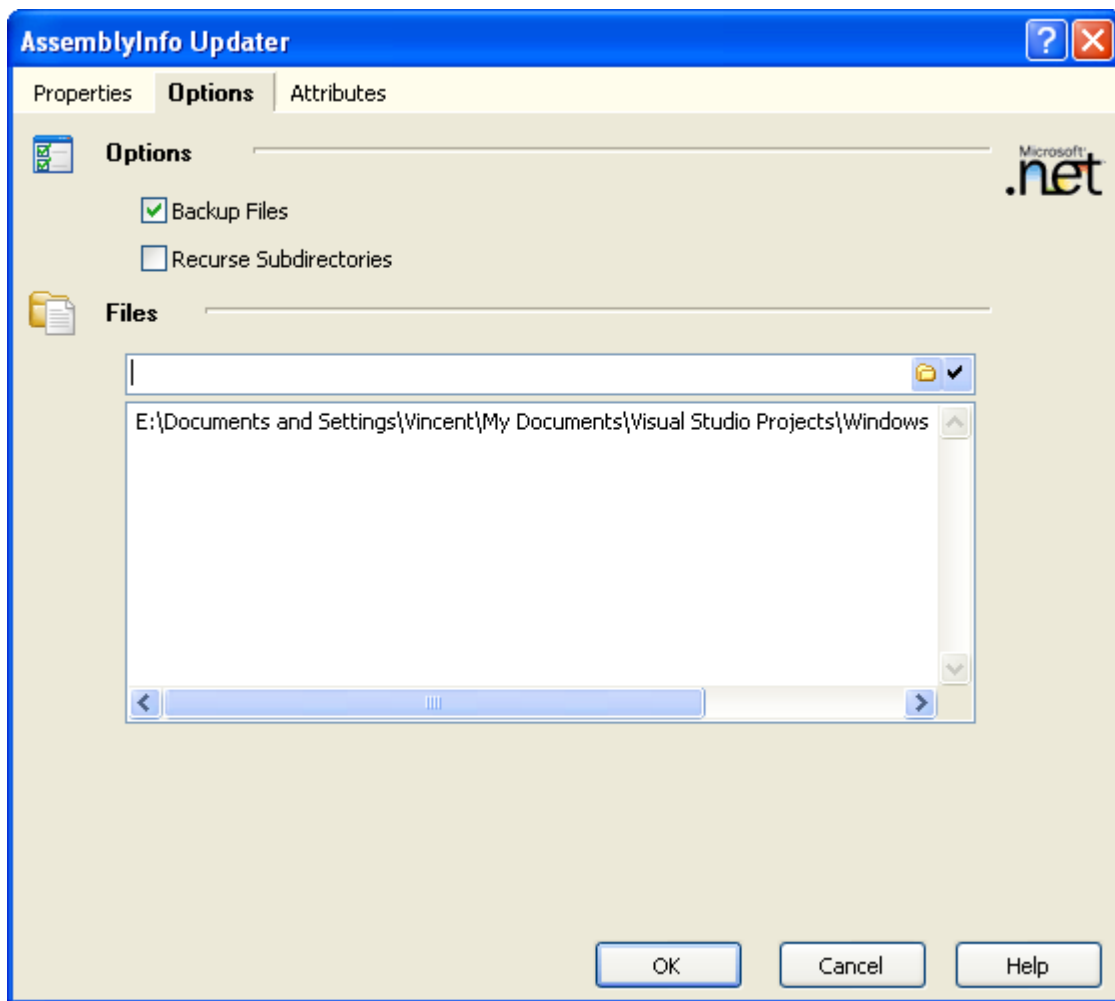
#### 5.8.4 MadExcept Compiler Action

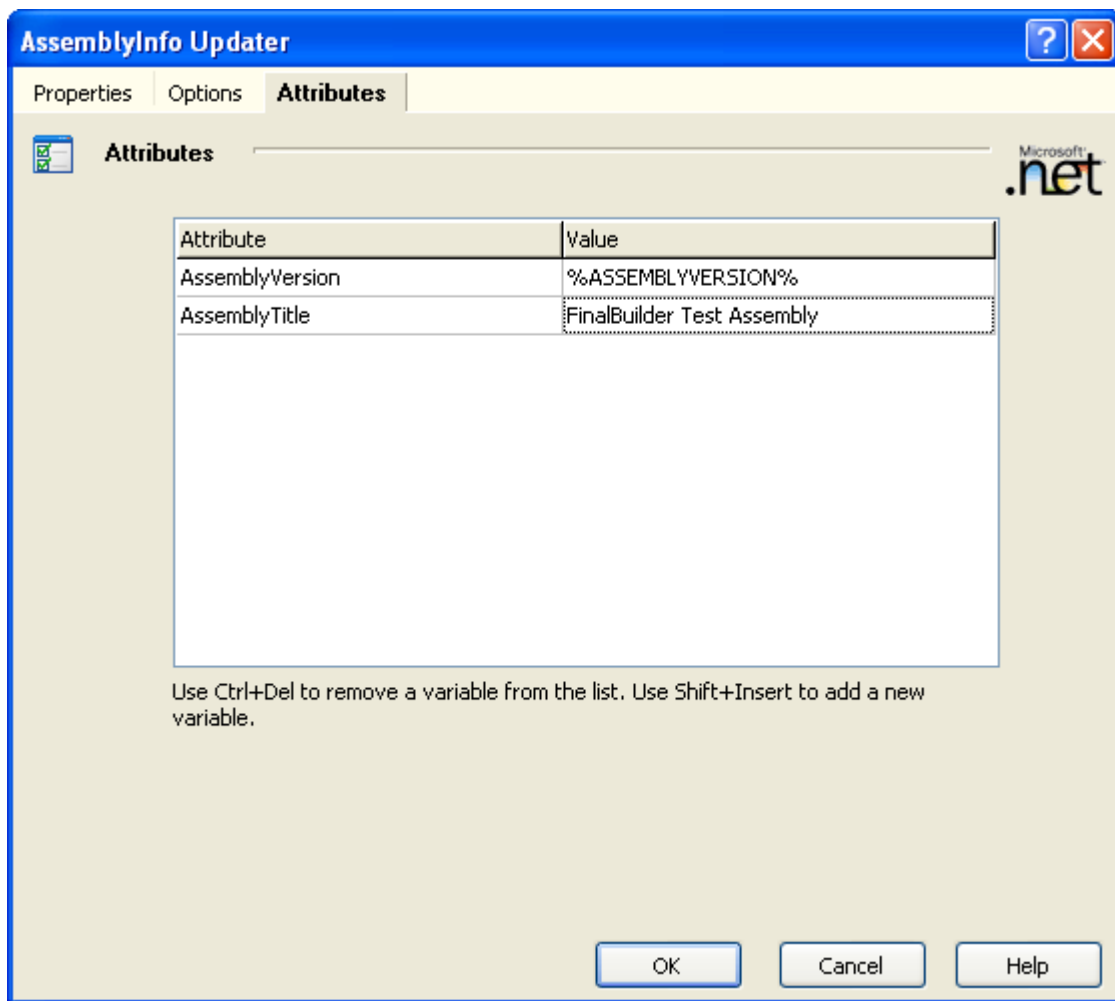
This Action provided support for the MadExcept product from <http://www.madshi.net> (while not strictly a compiler, it's a commonly used add on to the Borland Delphi compiler and that's why it's in the Compiler category).

#### 5.8.5 AssemblyInfo Updater Action

This action allows you to update the Assembly info in one or more source files. Languages supported are :

C#, VB.NET, J#, C++ and Delphi for .NET.





If the value you want to add to the assembly requires quotes, eg.  
[assembly: AssemblyFileVersion("1.0.0.3")]

Then you need to specify the quotes around the Value. FB doesn't know any specifics about the attributes you are adding, so you have to manually specify the quotes. Although, if you are updating an existing value, then FB will look to see if the value is quoted and preserve the quotes.

### 5.8.6 Chrome

The Chrome action enables you to automate compilation of your Chrome projects as part of your build process

Chrome™ is RemObjects' next generation Object Pascal language for the .NET and Mono Platforms. While implementing a language that stays true to the beauty and elegance of Object Pascal, Chrome adds useful design elements from other languages such as C#, Java and Eiffel, and it introduces its own language innovations.

Use Chrome to write fully managed native .NET applications for the Microsoft .NET Framework, the Compact Framework or the Mono Platform, and develop your applications inside the well known Visual Studio .NET IDE.

For more information on Chrome, see <http://www.chromesville.com/>

### **5.8.7 Incredibuild**

The IncrediBuild actions enable you to automate IncrediBuild distributed compiles as part of your build process.

IncrediBuild Actions:

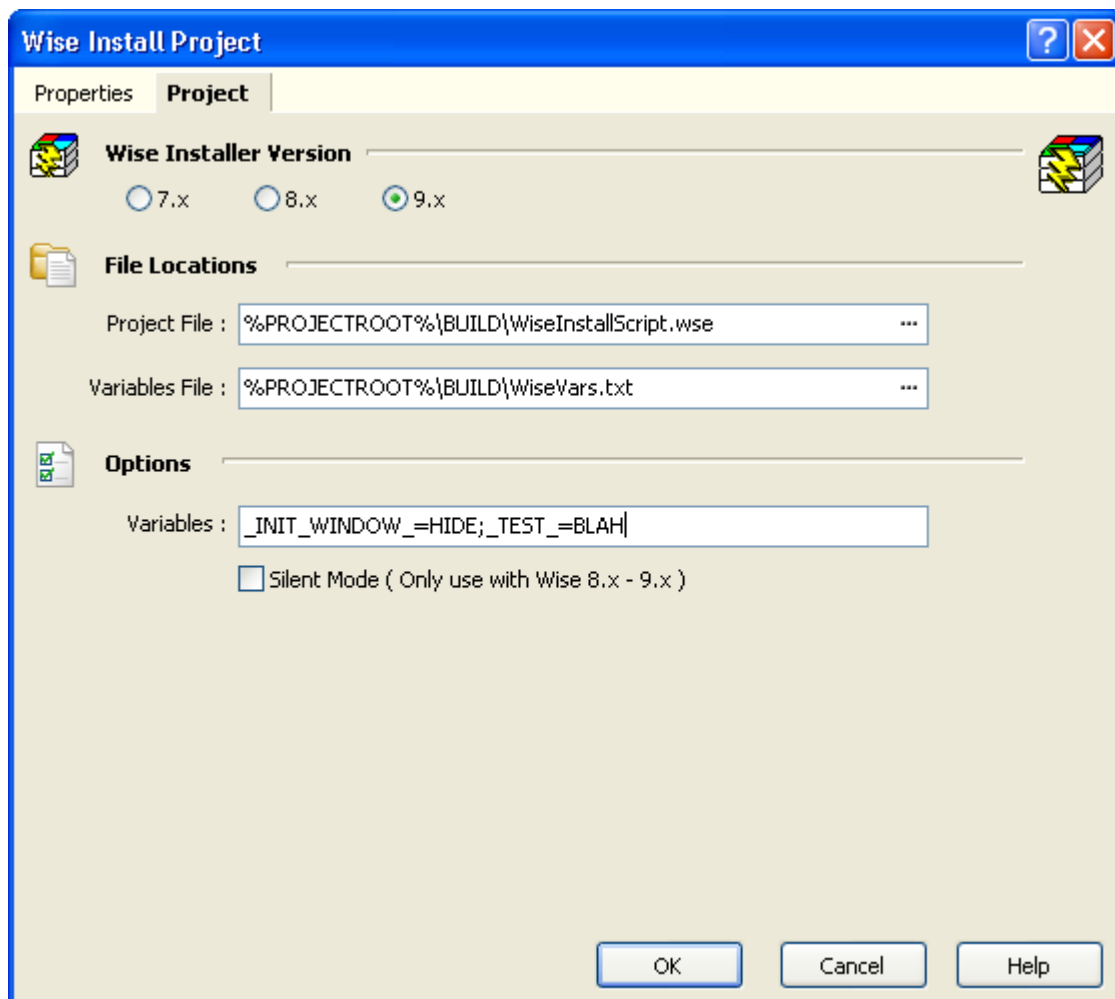
- IncrediBuild - executes an IncrediBuild distributed compilation
- Enable IncrediBuild Agent - enables the agent on the client machine
- Disable IncrediBuild Agent - disables the agent on the client machine
- Stop Current IncrediBuild Compile - stops any running IncrediBuild compilation
- Reset IncrediBuild Swapfile - resets the client machines IncrediBuild swap file

For more information on Xoreax IncrediBuild, please see <http://www.xoreax.com/>

## **5.9 Installers**

### **5.9.1 Wise InstallBuilder/InstallMaster**

This action provides an interface to Wise InstallMaster/InstallBuilder 7.0 and 8.x .



**Project File :** The full path and file name of the wise project file (.wse).

**Variables File :** The full path and file name to a file that contains compiler variables. Format is `_VAR_=value`, one entry per line.

**Variables :** Compiler variables, format is `_VAR_=value;_ANOTHER_VAR_=value`

**NOTE:** The Wise compiler does not return a non zero return code when an error occurs during compilation. This means that FinalBuilder has no way of detecting if the compile of the installer failed or not. Usually Wise displays a message box when an error occurred.

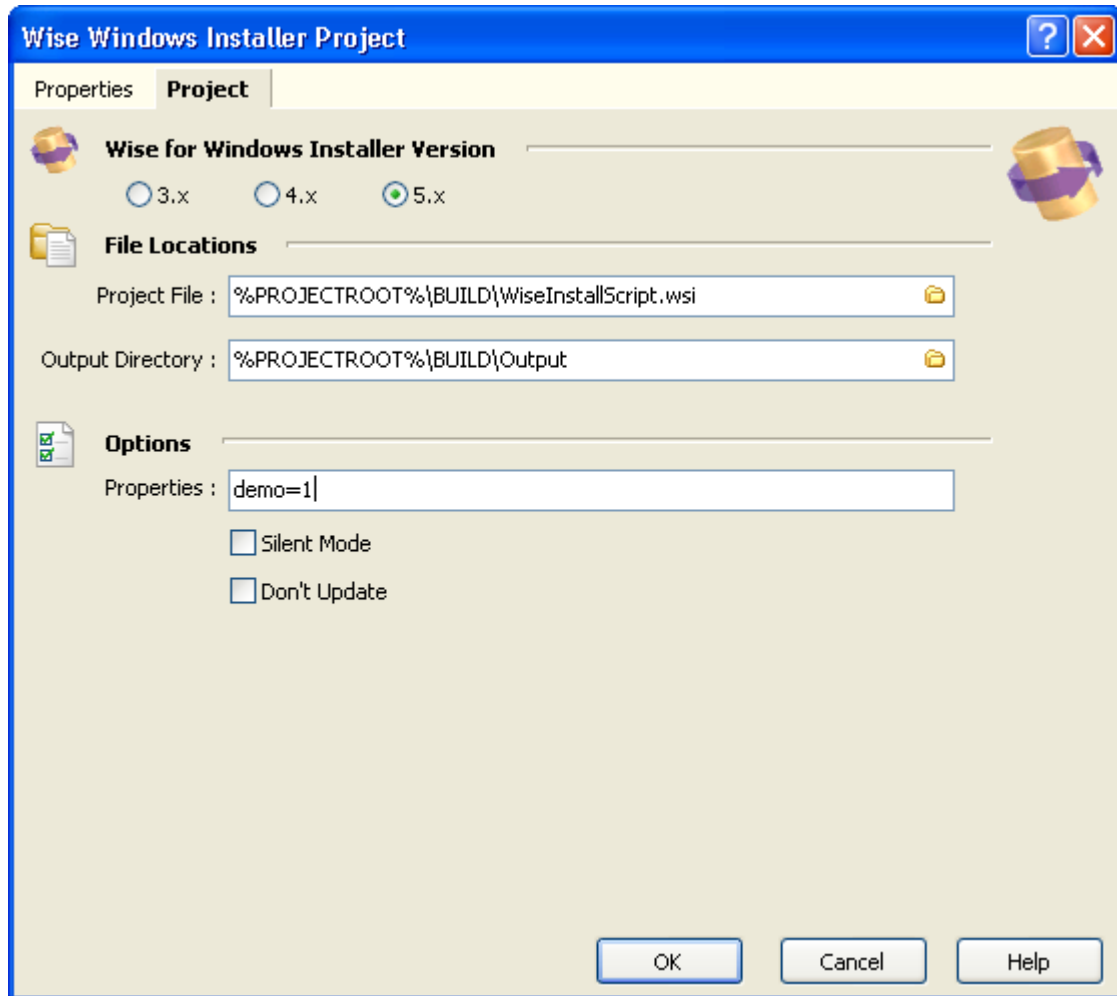
### Scripting Info

The Action properties available are :

**property** ProjectFile : WideString  
**property** Variables : WideString  
**property** VariablesFile : WideString

### 5.9.2 Wise For Windows Installer

This action provides an interface to Wise for Windows Installer version 3.



**Project File :** The full path and file name of the wise project file (.wse).

**Output Directory :** Output File Path.

**Properties :** Project Properties, format is name=value;name=value

#### Scripting Info

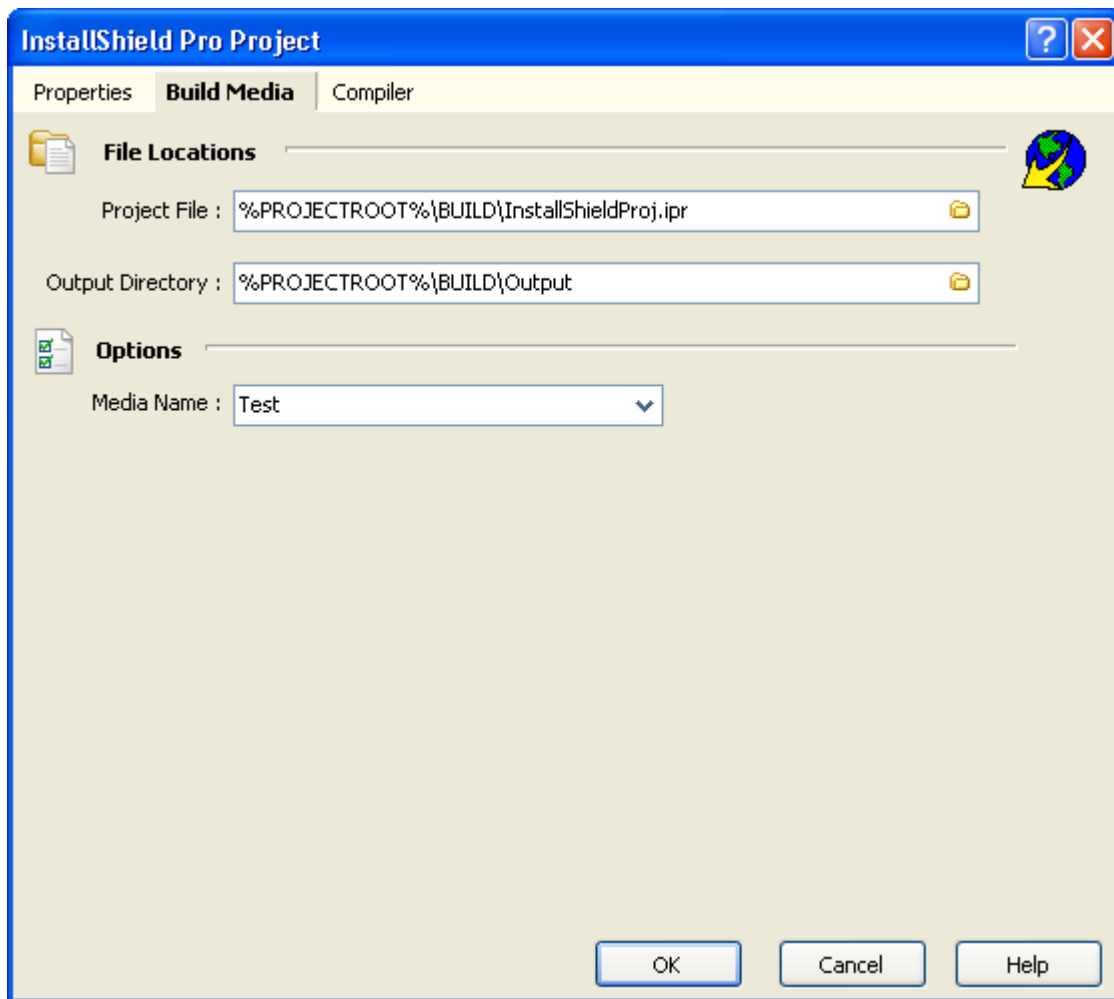
The Action properties available are :

**property** ProjectFile : WideString  
**property** Properties : WideString  
**property** OutputFilePath : WideString



### 5.9.3 InstallShield Pro - Std Edition

This Action provides an interface to InstallShield Pro - Standard Edition version 6.2.x - 7.x

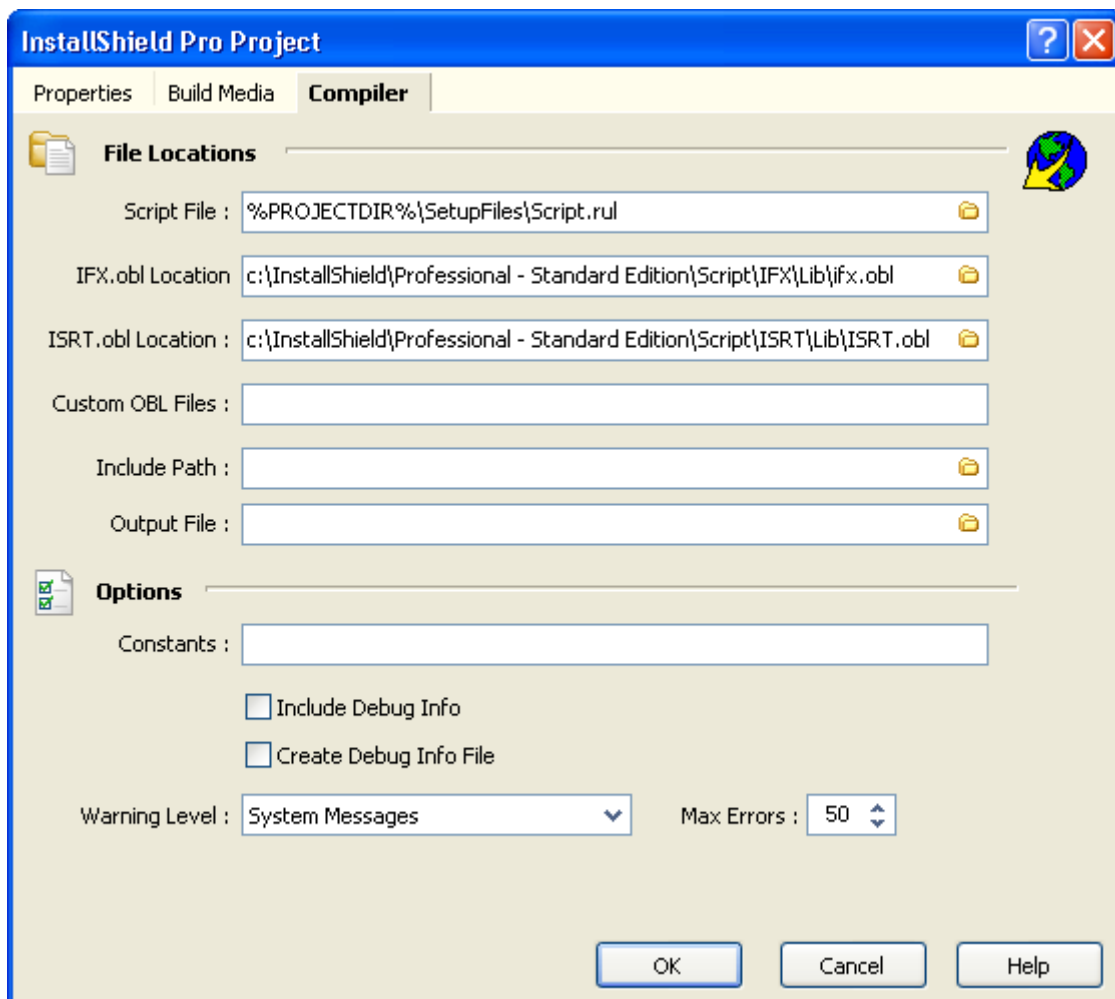


**Project File :** The fully qualified path to the setup project file (\*.ipr).

You can also compile Installshield Objects (.ipo) but you'll have to specify the .ipo file without using the file dialog. You will also have to manually enter the Media name, as the automatically detected media names only work for .ipr files.

**Output Directory :** The fully qualified path to where you want the output folders and files to be placed. The built setup files will be placed in the "Disk Images\Disk1" subfolder in the location you specify.

**Media Name :** The media name that you entered into the Media Wizard - Media Name Panel in InstallShield.



**Script File :** Specifies the name of the Setup Script.

**IFX.obl Location :** The location of the ifx.obl file, defaults to the value specified in the Options dialog.

**ISRT.obl Location :** The location of the isrt.obl file, defaults to the value specified in the Options dialog.

**Custom OBL Files :** The fully qualified path (separated by semi-colons) of any custom library files your setup requires

**Include Path :** Specifies the search path for source files that have been included in the setup by means of #include statements.

**Constants :** Specify any constants you require in the form identifier=value , separate multiple constants with semicolons

**Output File :** Specifies the file name to assigned to the compiled script.

**Include Debug Info :** Specifies that debugging information should be included in the compiled script file, so a debugging information file is not needed.

**Create Debug Info File :** Specifies that a debugging information file should be produced.

**Warning Level :** Sets error reporting level for the compiler.

**Max Errors :** The maximum number of errors before the compiler aborts.

### Scripting Info

The Action properties available are :

**property** BuildMedia : WordBool // Build the Media

**property** CompileSetup : WordBool //CompileSetup // compile the setup program

**property** ProjectFile : WideString

**property** OutputDirectory : WideString

**property** MediaName : WideString;

**property** SetupScriptFile : WideString

**property** IFXLocation : WideString

**property** ISRTLLocation : WideString

**property** CustomOBLFiles : WideStringSetCustomOBLFiles;

**property** Constants : WideString

**property** MaxErrors : integer

**property** CreateDebugInfoFile : WordBool

**property** IncludeDebugInfo : WordBool

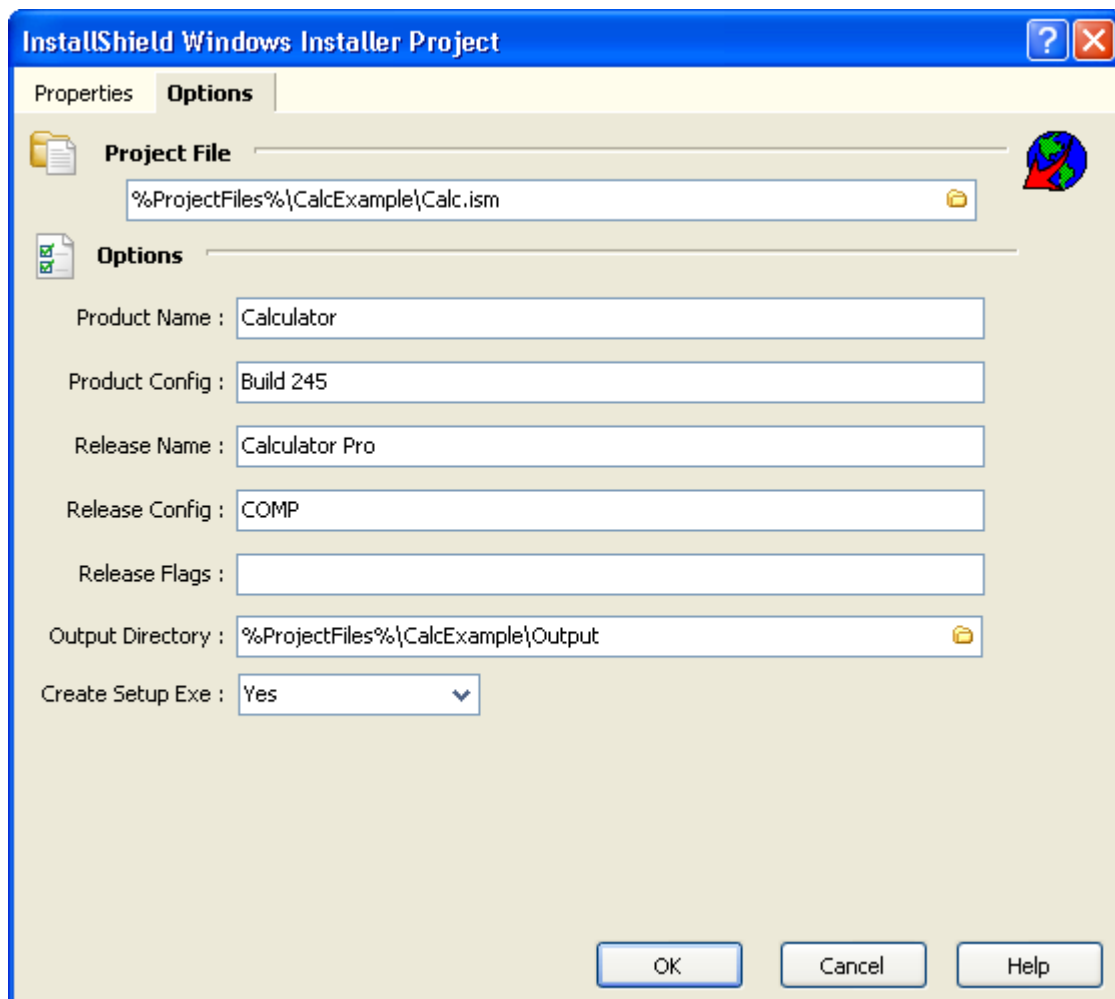
**property** IncludePath : WideString

**property** WarningLevel : integer

**property** OutputFileName : WideString

## 5.9.4 InstallShield Pro - Windows Installer Edition

This Action provides an interface to InstallShield Pro - Windows Installer Edition



**Project File :** Project File (.ism)

**Product Name :** The Product Name in the .ism file

**Product Config :** Product Configuration for the release. If it does not exist it will be created. Although this parameter is not required, it is a good idea to include it if you are specifying a Release Name.

**Release Name :** The Release Name as specified in the Release Wizard (in the InstallShield IDE) You can use an existing release name or create a new one. Although this option is not required, it is a good idea to include it if you are specifying the flag for product configurations.

**Release Config :** This option allows to specify whether you would like to have your release compressed into one file or remain uncompressed in multiple files. This is optional, if the release name already exists, the configuration will be based on what is specified in the InstallShield IDE. If this is ignored for a new release, the new package will be uncompressed.

**Release Flags :** Use this option to specify any Release Flags that you would like to include in your build. Separate multiple flags with a comma.

**Output Directory :** Qualified path to where you want the output folders to be placed. This is optional, if not specified then the build will place the build package and files in the directory specified in the Project Location section of the Options panel in InstallShield.

**Create Setup.Exe :** Specify whether or not you would like to create a setup.exe along with your setup project.

### Scripting Info

The Action properties available are :

**property** ProjectFile : WideString  
**property** ProductName : WideString  
**property** ReleaseName : WideString  
**property** ProductConfig : WideString  
**property** ReleaseConfig : WideString  
**property** OutputDirectory : WideString  
**property** ReleaseFlags : WideString  
**property** CreateSetupExe : WordBool

## 5.9.5 InstallShield Developer

This Action provides an interface to InstallShield Developer 7.x - 9.x

The screenshot shows the 'InstallShield Developer Project' dialog box with the 'Options' tab selected. The dialog has three tabs: 'Properties', 'Options', and 'Stand Alone'. The 'Options' tab contains several sections:

- Project File:** A text field containing '%ProjectFiles%\CalcExample\Calc.ism' with a folder icon to its right.
- Options:** A section with several text fields and a dropdown:
  - Product Config : Default
  - Release Name : Release 1
  - Release Config : Default
  - Release Flags : (empty text field)
  - Output Directory : %ProjectFiles%\CalcExample\Output (with a folder icon)
  - Create Setup Exe : Default (dropdown menu)
- Checkboxes:** Three checkboxes are located below the 'Options' section:
  - ☐ Silent Mode
  - ☒ Stop On First Error
  - ☒ Treat Warnings as Errors
- Developer 8/9/X Only Options:** A section with two text fields and a dropdown:
  - Validation File : (empty text field with a folder icon)
  - Build Option : Compile Setup (dropdown menu)

At the bottom of the dialog are three buttons: 'OK', 'Cancel', and 'Help'.

**Project File :** Project File (.ism)

**Product Config :** Product Configuration for the release. If it does not exist it will be created. Although this parameter is not required, it is a good idea to include it if you are specifying a Release Name.

**Release Name :** The Release Name as specified in the Release Wizard (in the InstallShield IDE) You can use an existing release name or create a new one. Although this option is not required, it is a good idea to include it if you are specifying the flag for product configurations.

**Release Config :** This option allows to specify whether you would like to have your release compressed into one file or remain uncompressed in multiple files. This is optional, if the release name already exists, the configuration will be based on what is specified in the InstallShield IDE. If this is ignored for a new release, the new package will be uncompressed.

**Release Flags :** Use this option to specify any Release Flags that you would like to include in your build. Separate multiple flags with a comma.

**Output Directory :** Qualified path to where you want the output folders to be placed. This is optional, if not specified then the build will place the build package and files in the directory specified in the Project Location section of the Options panel in InstallShield.

**Create Setup.Exe :** Specify whether or not you would like to create a setup.exe along with your setup project.

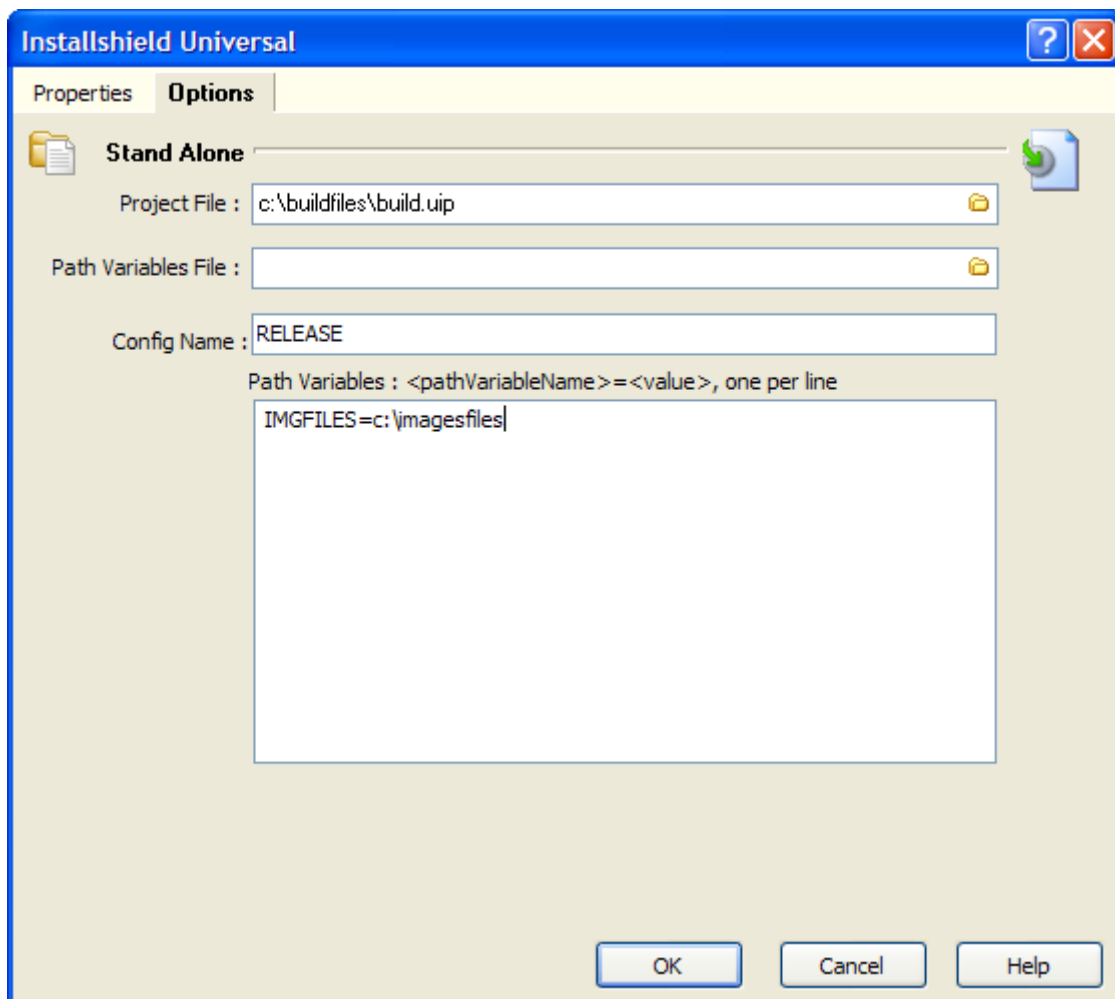
**Silent Mode :** Run the InstallShield compiler in silent mode, this will disable the output of compiler messages.

**Treat Warnings as Errors:** Any warnings issued by the Installshield compiler will be treated as errors.

**Stop on First Error :** By default, the Installshield compiler does not stop when it encounters an error, it just keeps a count of errors to report at the end. Enable this option to stop the build as soon as the compiler encounters any error.

### 5.9.6 InstallShield Universal Installer

The InstallShield Universal action enables you to automate the process of building InstallShield Universal installations as part of your build process.



**Project File** - specify your InstallShield Universal project file

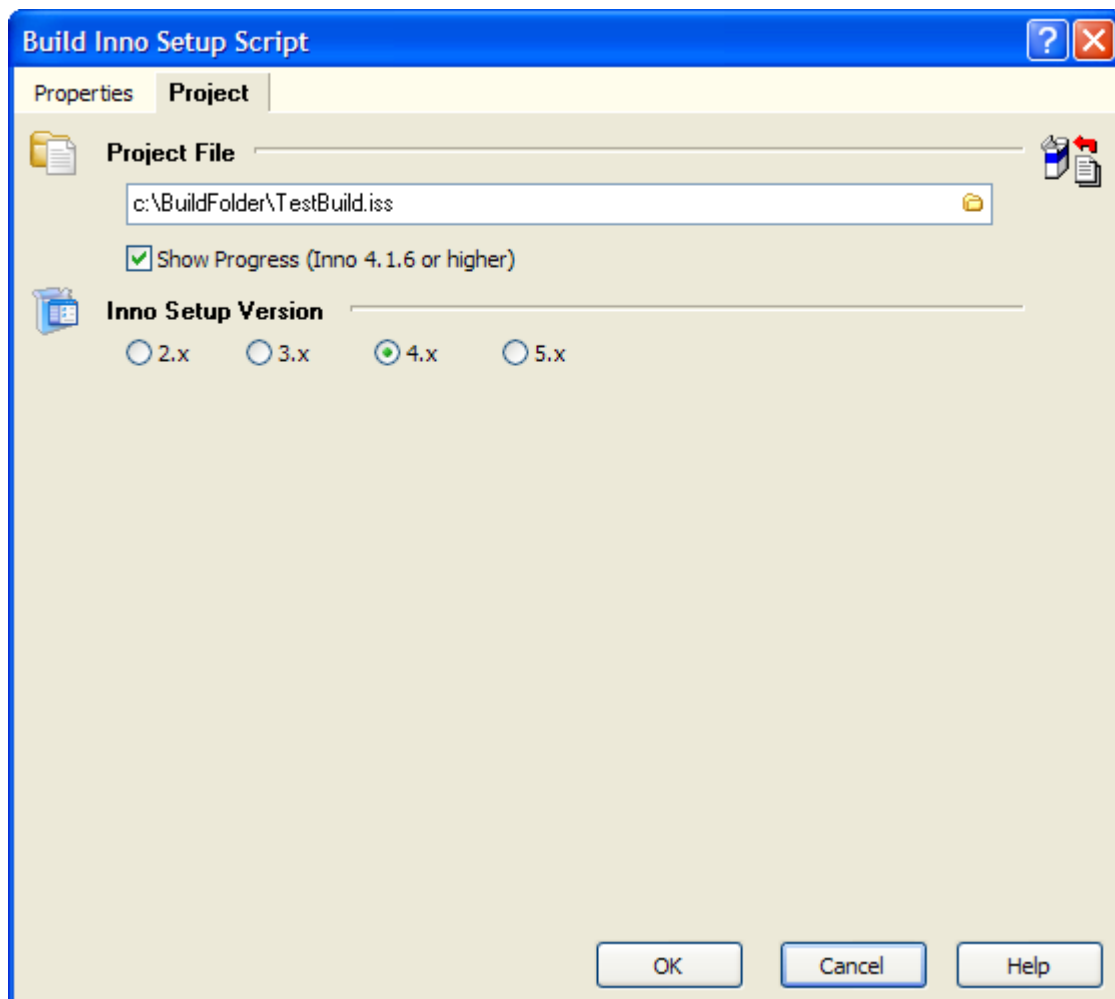
**Path Variables File** - specify the fully qualified path to your variables file

**Config Name** - enter the configuration you want to run. The configurations are defined in your project file.

**Path Variables** - enter name/value pairs defining the values for directory variables

### 5.9.7 Inno Setup

This action provides an interface to the Inno Setup Compiler. To use this action you need to specify the path to the compiler dll (ISComplr.dll) in the FinalBuilder Options dialog. FinalBuilder supports InnoSetup versions 2.x - 5.x



**Project file :** The fully qualified path to the setup script file (\*.iss)

### Scripting Info

The Action properties available are :

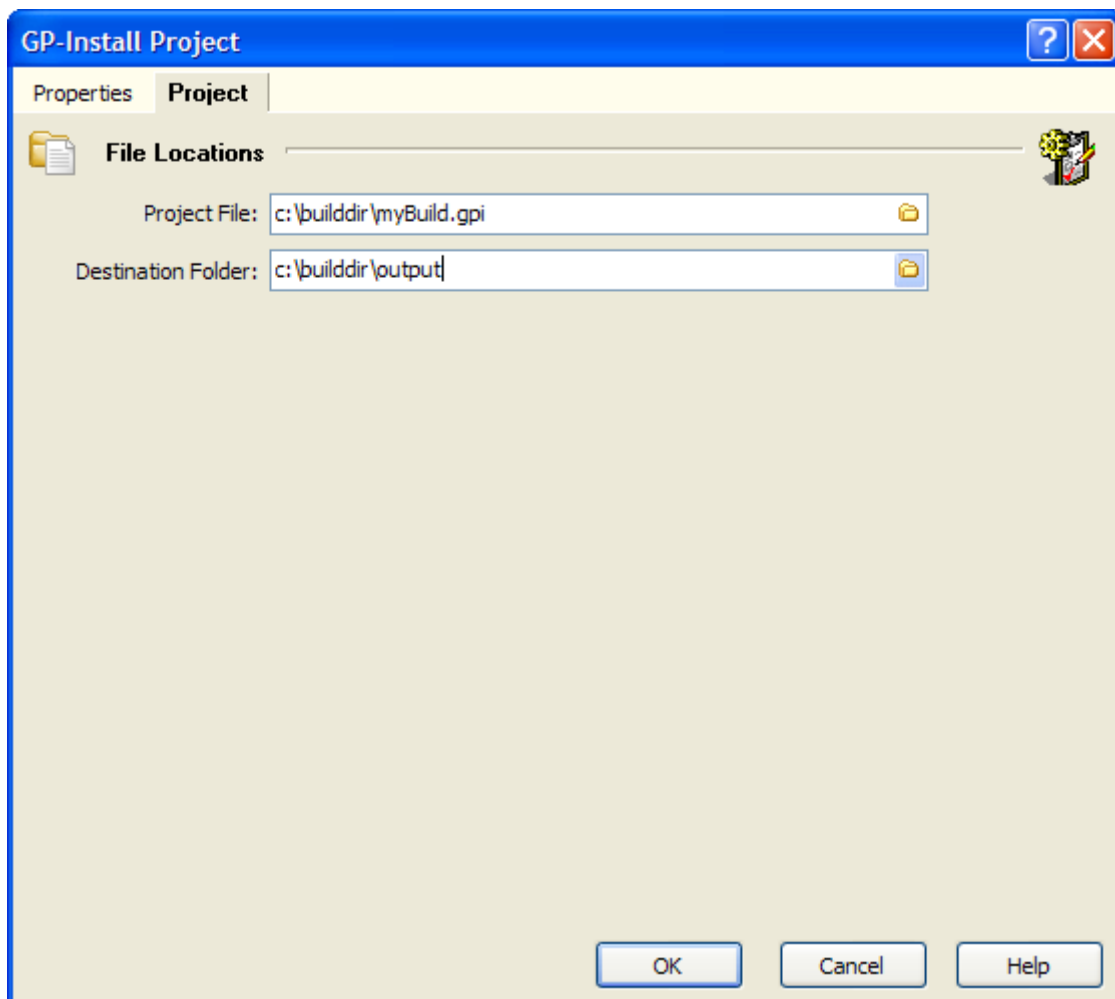
**property** ScriptFile : WideString

Inno Setup is a freeware setup compiler which can be downloaded from :  
<http://www.jrsoftware.org/isinfo.htm>

### 5.9.8 GPInstall Action

This Action provides support for the popular Freeware install tool GPInstall.





### Scripting Info

The Action properties available are :

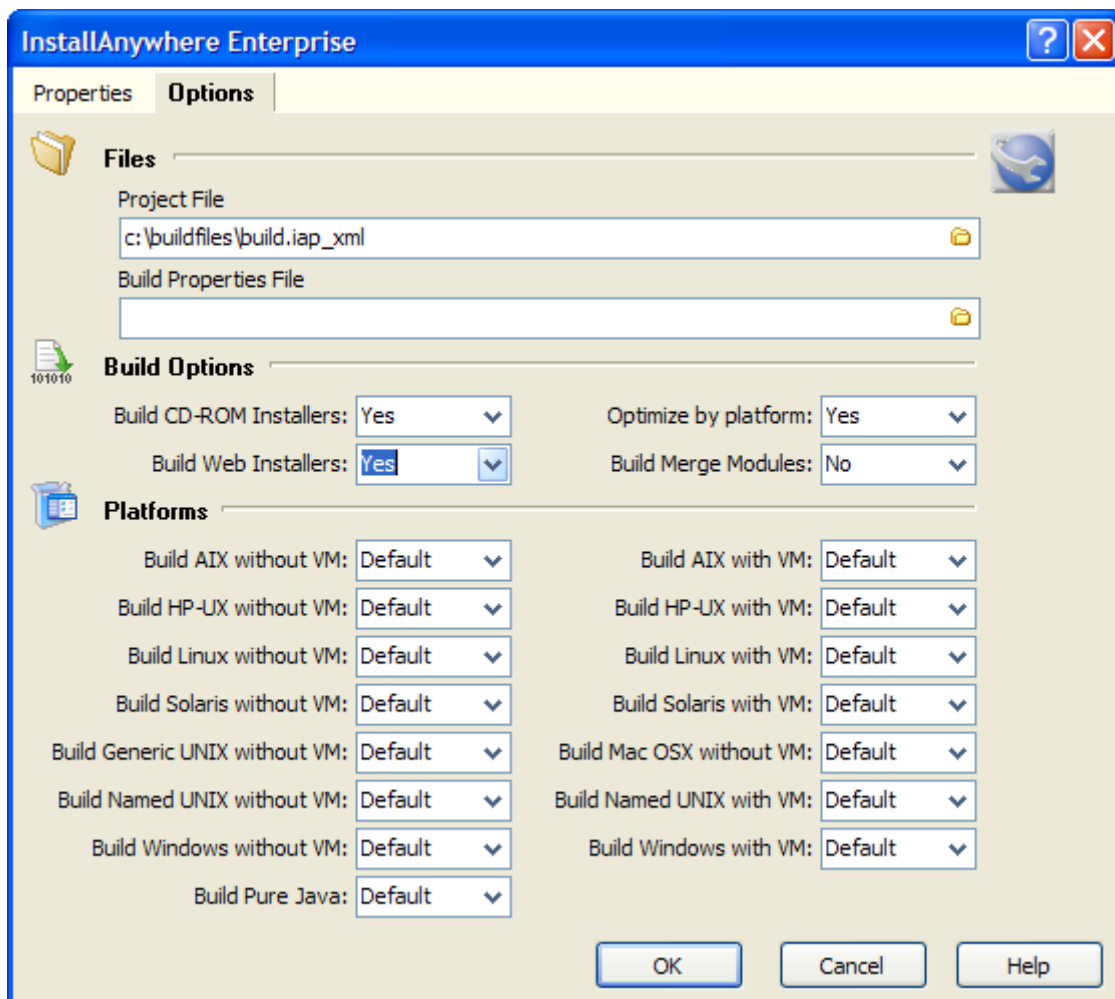
**property** DestinationFolder : WideString;  
**property** ProjectFile : WideString;

### 5.9.9 InstallAnywhere Enterprise

The InstallAnywhere Enterprise action enables you to automate the process of building ZeroG InstallAnywhere Enterprise installations as part of your build process.

Zero G's multiplatform solutions make deploying, delivering, and updating software applications easier and faster. Our award-winning solutions are full-featured, multiplatform, and perfect for enterprise development teams.

For more information, see <http://www.zerog.com>

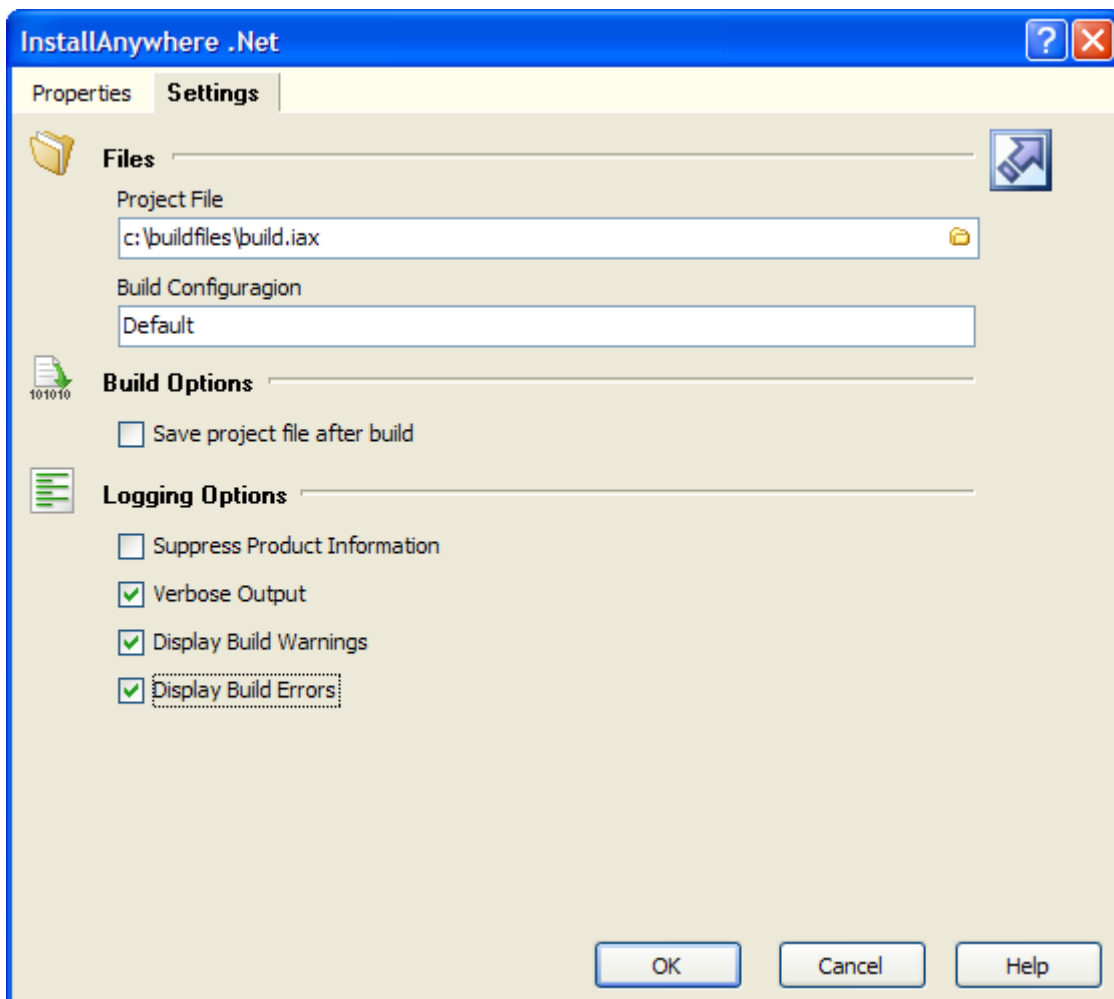


### 5.9.10 InstallAnywhere .Net

The InstallAnywhere.Net action enables you to automate the process of building ZeroG InstallAnywhere.Net installations as part of your build process.

Zero G's multiplatform solutions make deploying, delivering, and updating software applications easier and faster. Our award-winning solutions are full-featured, multiplatform, and perfect for enterprise development teams.

For more information, see <http://www.zerog.com>

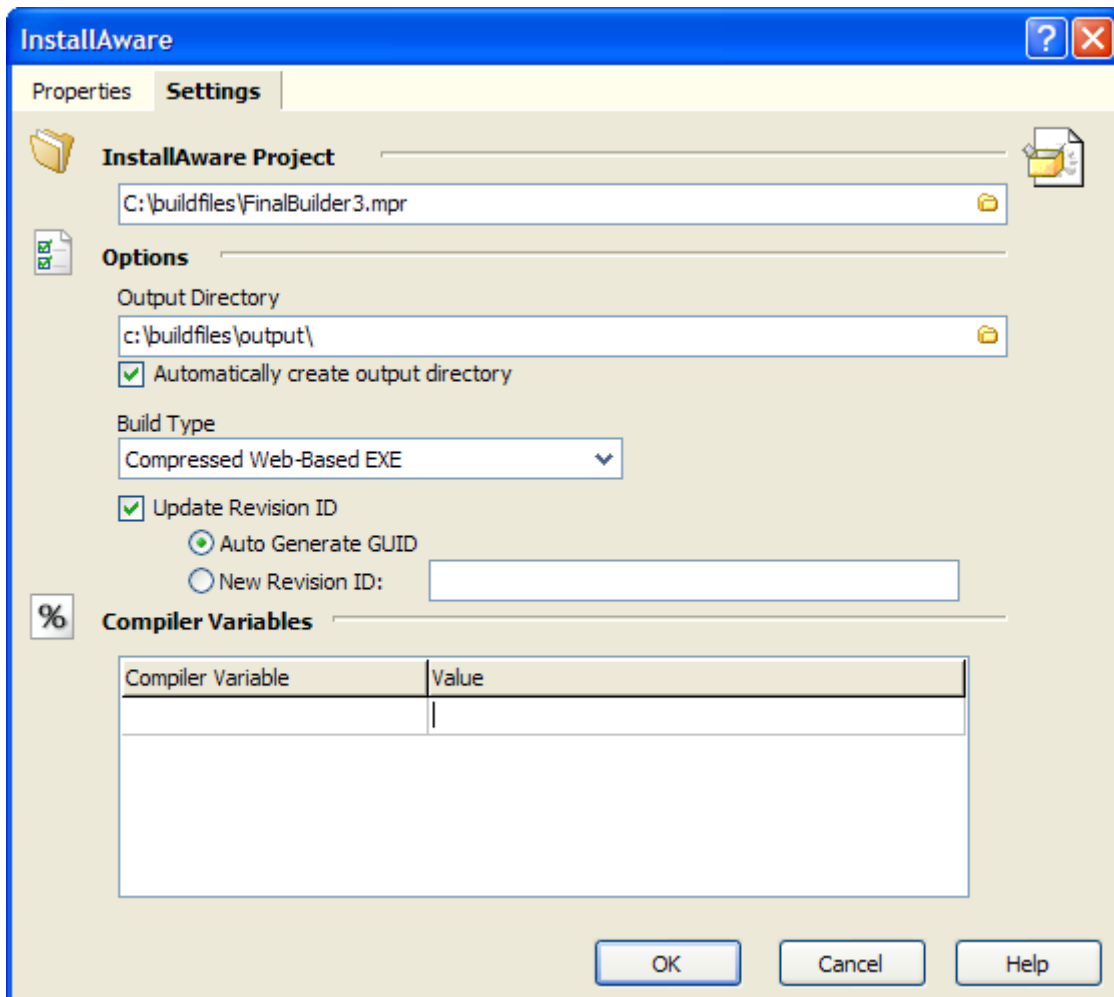


### 5.9.11 InstallAware

The InstallAware action enables you to automate the process of building InstallAware installations as part of your build process.

The InstallAware product family provides a complete set of installation authoring tools and environments targeting Microsoft Windows and Microsoft .NET platforms

For more information, see <http://www.installaware.com>



### Update Revision ID

This will modify the InstallAware project file with a new Revision ID. The revision ID uniquely identifies the setup to Windows Installer. Each revision should have a new Revision ID otherwise Windows Installer will attempt to do a Repair/Remove instead of uninstalling an old build and then running the new build.

### Auto Generate GUID

FinalBuilder will automatically generate a new GUID for the RevisionID each time the action runs.

### New Revision ID

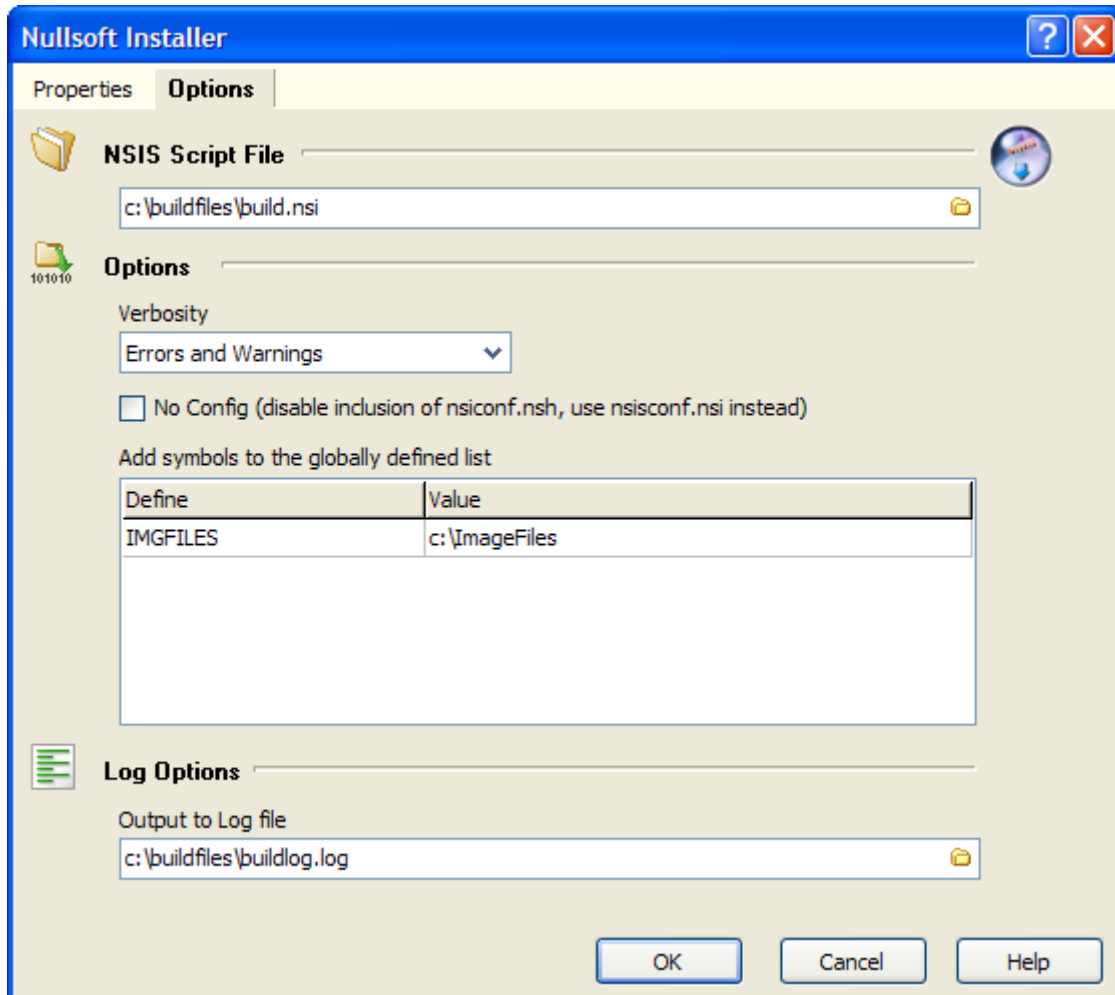
Specify the new Revision ID that you want this version to have.

## 5.9.12 Nullsoft NSIS

The Nullsoft Installer action enables you to automate building Nullsoft Installer projects as part of your build process.

NSIS (Nullsoft Scriptable Install System) is a tool that allows programmers to create installers for Windows. It is released under an open source license and is completely free for any use.

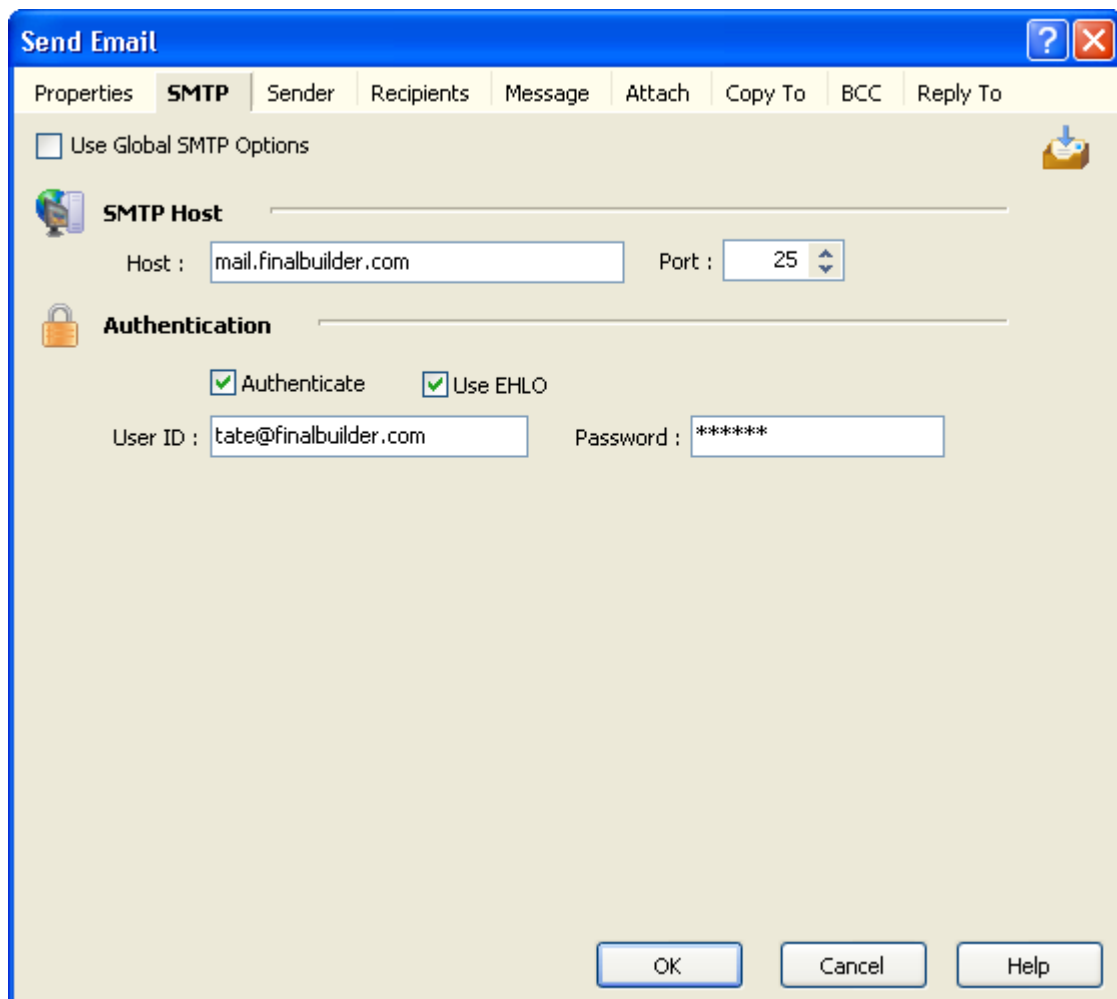
For more information see <http://nsis.sourceforge.net/>



## 5.10 Internet

### 5.10.1 Send Email (SMTP)

This action allows your FinalBuilder projects to send email via an SMTP server.



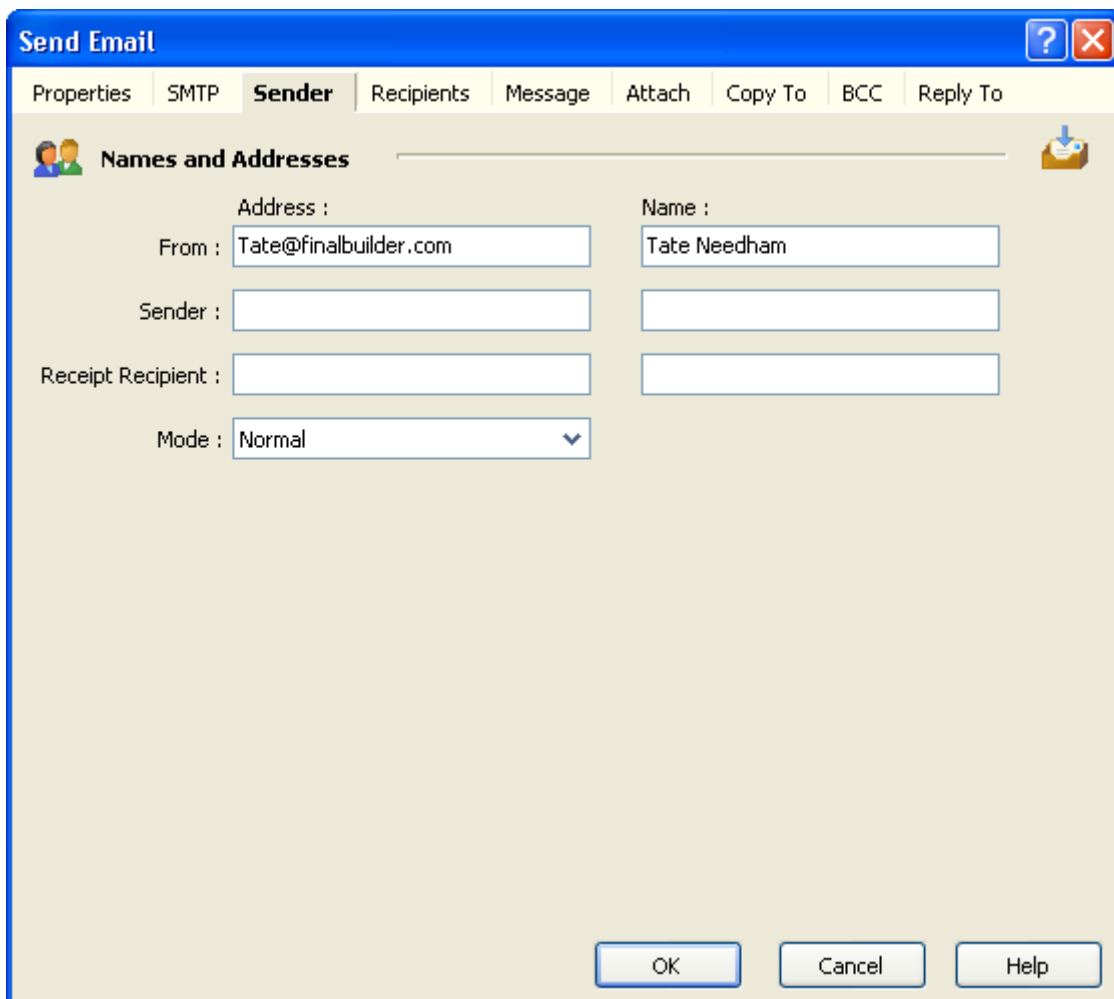
The image shows a 'Send Email' dialog box with a blue title bar and standard Windows window controls. It features a tabbed interface with 'Properties', 'SMTP', 'Sender', 'Recipients', 'Message', 'Attach', 'Copy To', 'BCC', and 'Reply To'. The 'SMTP' tab is active. At the top, there is a checkbox for 'Use Global SMTP Options' which is unchecked. Below this, the 'SMTP Host' section includes a text field for 'Host' containing 'mail.finalbuilder.com' and a spinner box for 'Port' set to '25'. The 'Authentication' section, marked with a lock icon, contains two checked checkboxes: 'Authenticate' and 'Use EHLO'. Below these are text fields for 'User ID' (containing 'tate@finalbuilder.com') and 'Password' (containing '\*\*\*\*\*'). At the bottom right are 'OK', 'Cancel', and 'Help' buttons.

**Host :** The host name (or IP address) of the SMTP server

**Port :** The port on which to connect, defaults to 25

**UserID /Password:** If your smtp server requires authentication, provide your userid and password

**Use EHLO:** Enabling EHLO uses a slightly different protocol to communicate with the SMTP server. EHLO allows the SMTP server to report it's capabilities to the client (in this case FinalBuilder), and then the client can adjust it's protocol to support the reported capabilities. If your SMTP server is Lotus Domino and using it's ESTMP protocol, then you need to enable EHLO.



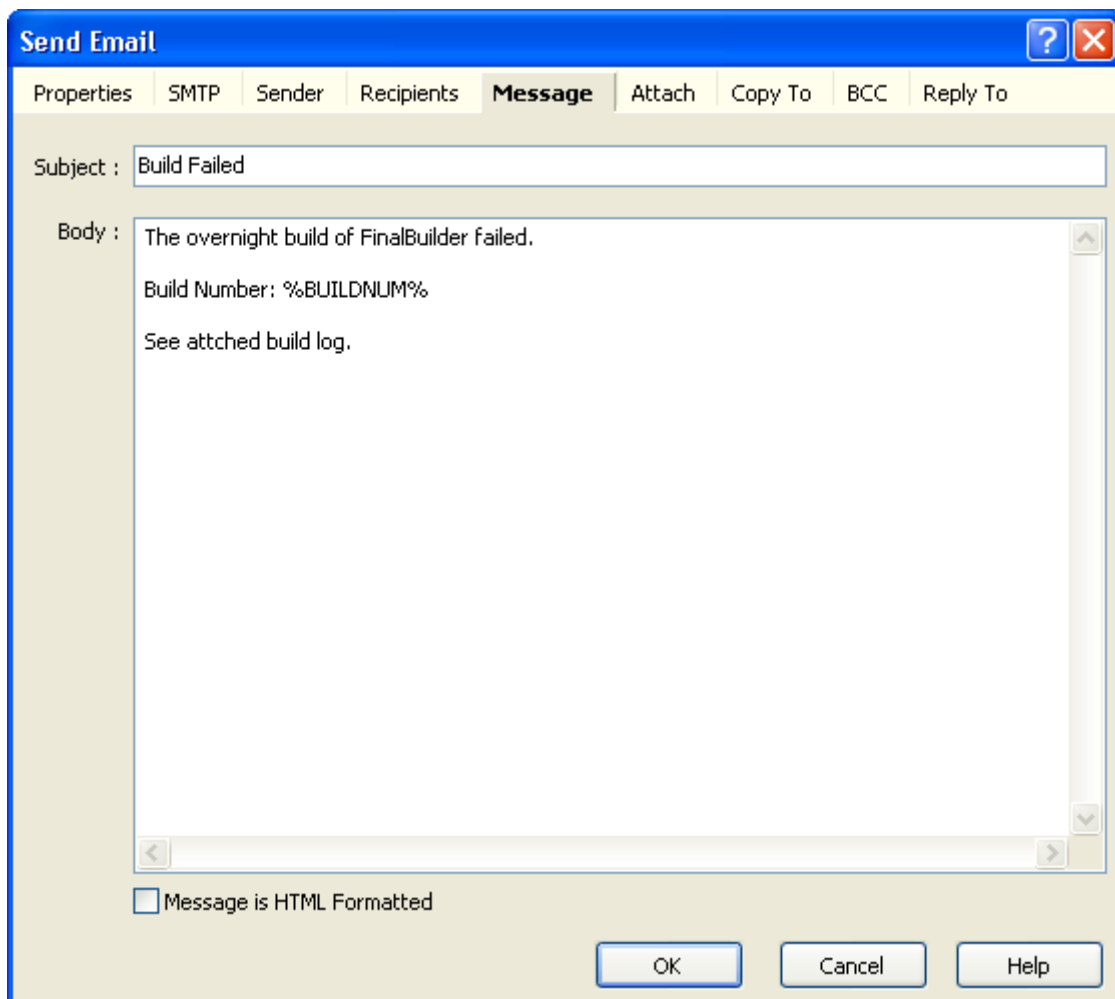
The image shows a 'Send Email' dialog box with a blue title bar and standard Windows window controls. It features a tabbed interface with tabs for 'Properties', 'SMTP', 'Sender', 'Recipients', 'Message', 'Attach', 'Copy To', 'BCC', and 'Reply To'. The 'Sender' tab is currently selected. Below the tabs is a section titled 'Names and Addresses' with a small icon of two people. This section contains several input fields: 'From' (with the value 'Tate@finalbuilder.com'), 'Name' (with the value 'Tate Needham'), 'Sender', 'Receipt Recipient', and 'Mode' (a dropdown menu currently set to 'Normal'). At the bottom of the dialog are three buttons: 'OK', 'Cancel', and 'Help'.

**From :** This must be set to a valid email address

**Sender :** Set this when sending mail on behalf of someone else

**Receipt Recipient :** If you require a notice receipt then set the address for the receipt message to be sent to

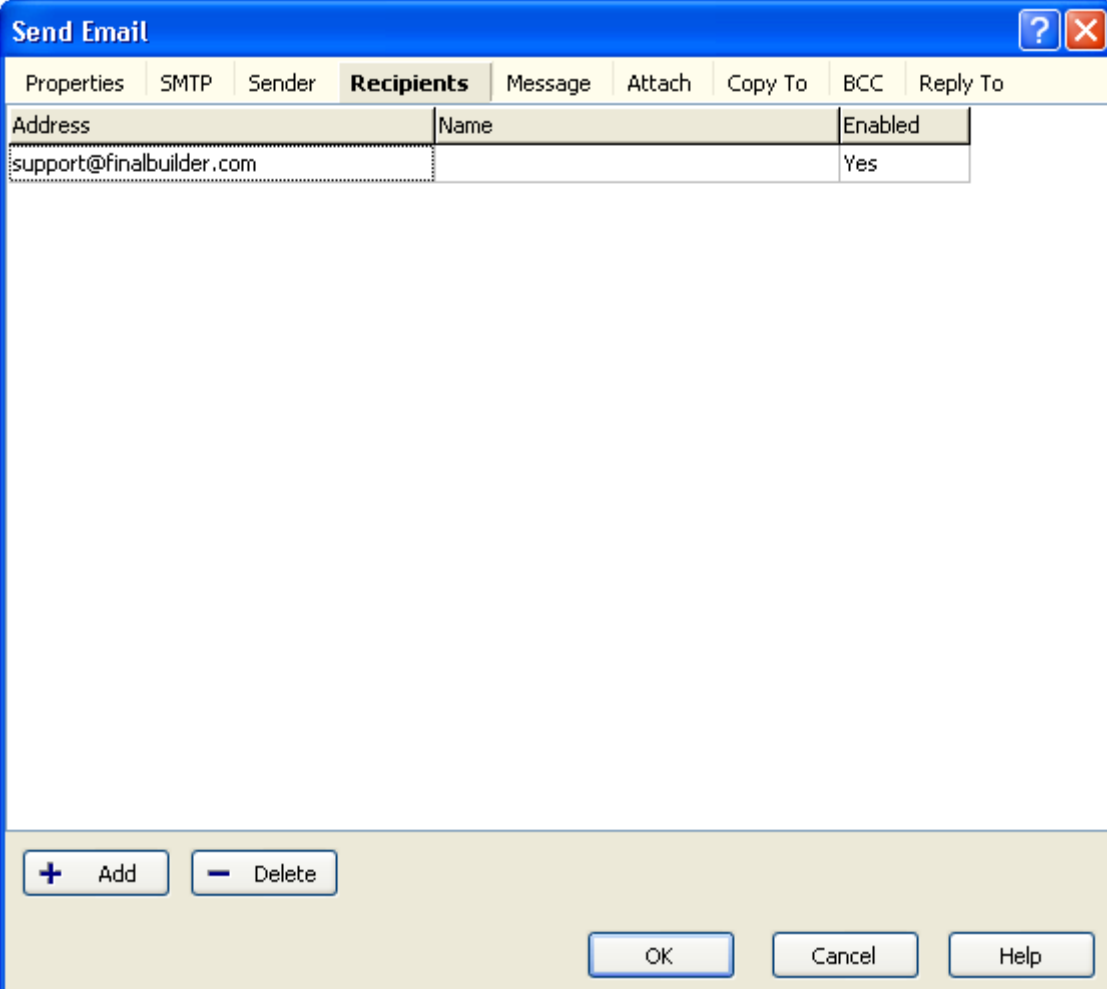
**Mode :** When operating in normal Mode, the action sends a single email message. When operating in individual mode, it sends a message to each individual specified in the recipients property. Note that the copy to and blind copy to lists are not used in this mode.



**Subject :** The email message's subject

**Body :** The actual message.

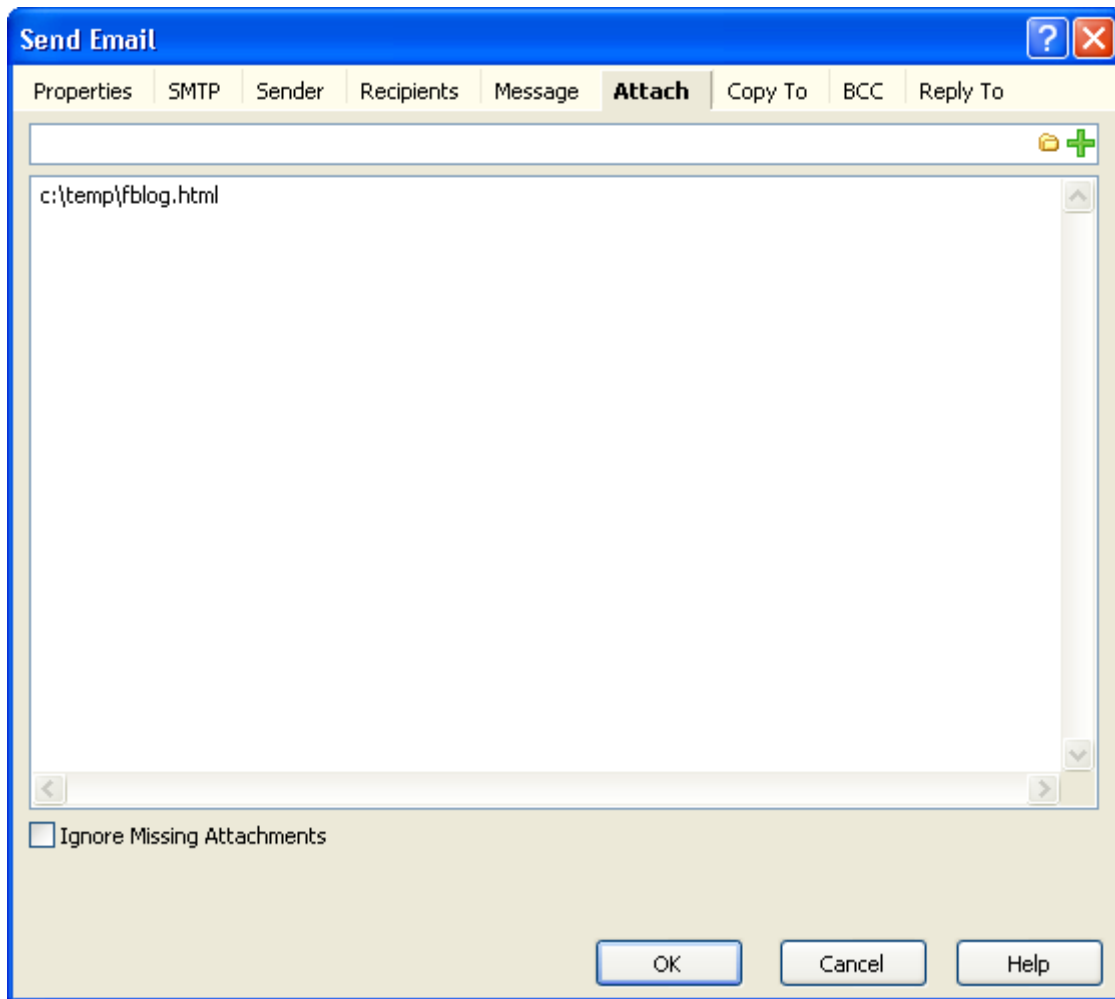




The image shows a 'Send Email' dialog box with a blue title bar and standard Windows window controls. It features a tabbed interface with the following tabs: Properties, SMTP, Sender, Recipients (selected), Message, Attach, Copy To, BCC, and Reply To. The 'Recipients' tab contains a table with three columns: Address, Name, and Enabled. The table has one row with the email address 'support@finalbuilder.com' and the status 'Yes'. Below the table are buttons for '+ Add' and '- Delete'. At the bottom right are buttons for 'OK', 'Cancel', and 'Help'.

| Address                  | Name | Enabled |
|--------------------------|------|---------|
| support@finalbuilder.com |      | Yes     |

The Recipients, Copy To, Blind Copy To and Reply To tabs all have the same grid that allows you to specify multiple email address/names



To attach a file, click on the ... button and select the file(s) using the dialog. Then click on the tick button to add them to the attachments list.

### Scripting Info

The Action properties available are :

```

property Host : WideString;
property Port : integer;
property UserId : WideString;
property Password : WideString;
property Authenticate : WordBool;
property Subject : WideString;

property Recipients : IFBAddressList;
property CopyTo : IFBAddressList;
property BlindCopyTo : IFBAddressList;

property Sender : IFBEmailAddress;
property From : IFBEmailAddress;
property ReplyTo : IFBEmailAddress;
property ReceiptRecipient : IFBEmailAddress;
property Mode : integer ; // valid values are emNormal and emIndividual
  
```

**property** Body : WideString;

IFBAddressList interface (for Recipients, CopyTo and Blind Copy To)

**function** Add(const Name,Address : WideString) : IFBEmailAddress;

**function** Item(Index : integer) : IFBEmailAddress;

**procedure** Clear;

**procedure** RemoveItem(Index : integer);

**property** Count : integer;

IFBEmailAddress interface (for Sender, From,ReplyTo and ReceiptRecipient)

**property** Name : WideString

**property** Address : WideString

**property** Text : WideString; //in the form "My Name" <myaddress@mycompany.com>

This action uses the Open Source Indy Components, for more information see the Indy web site : <http://www.nevrona.com/Indy>

### 5.10.2 FTP Client

This action provides a basic FTP client, which can be used to upload or download files to/from an FTP server.

**FTP Client**

Properties **Server & Authentication** Options Script

☐ Use Global FTP Options

**FTP Server**

Host : ftp.t8software.com Port : 21

**Authentication**

User : t8software.com

Password : \*\*\*\*\*

OK Cancel Help

**Host :** The host name or IP address of the ftp server

**User :** Your user id on the ftp server

**Password :** Your password on the ftp server

**Port :** The port on which the FTP server (or proxy if you are using one) is listening

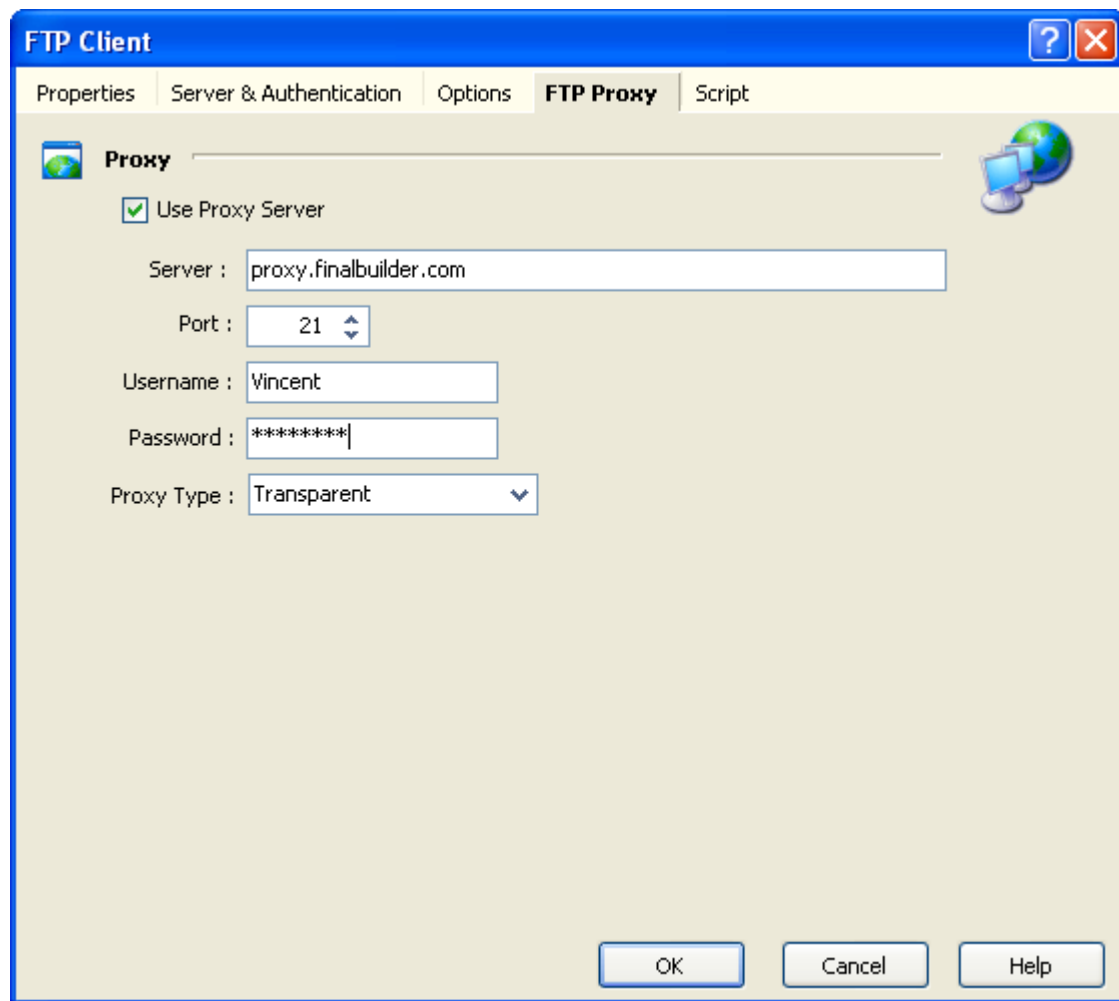
**Passive :** Instructs the FTP client and server to use passive mode transfers, this is needed when going through some firewalls

**Detailed Logging :** Enables Detailed logging.

**Binary Transfer Mode :** Sets the transfer mode to Binary (the default)

**ASCII Transfer Mode :** Sets the transfer mode to ASCII

### Proxy Settings



**Server :** The host name or ip address of the proxy

**Port** : The port number of the proxy

**UserName** : The User name for authentication

**Password** : The password for authentication

**Proxy Type** :

**None** - don't use a proxy

**UserSite** - Send command USER user@hostname

**Site** - Send command SITE (with logon)

**Open** - Send command OPEN

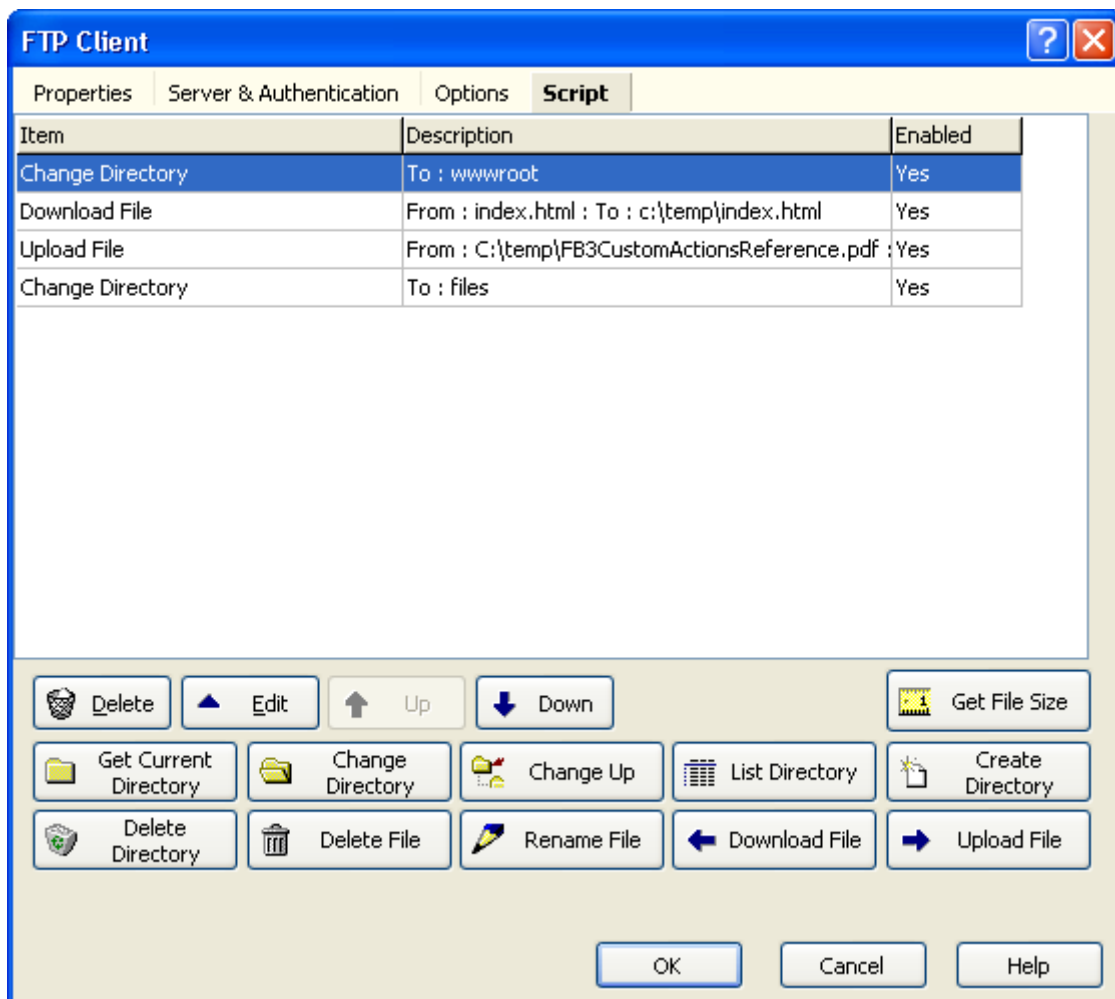
**UserPass** - USER user@firewalluser@hostname / PASS pass@firewallpass

**Transparent** - First use the USER and PASS command with the firewall username and password, and then with the target host username and password.

**HttpProxyWithFtp** - HTTP Proxy with FTP support. Will be supported in Indy 10

**CustomProxy** - use OnCustomFTPProxy to customize the proxy login

### FTP Script



The FTP client works by adding FTP commands to the list. The available commands are :

**Get Current Directory** : This can be retrieved into a Variable

**Change Directory :** Change the remote directory, you can use FinalBuilder Variables with this command.

**Change Up :** Changes the remote directory to its parent directory, ie up one level

**List Directory :** Lists the remote directory into a variable.

**Create Directory :** Create a sub directory in the current remote directory.

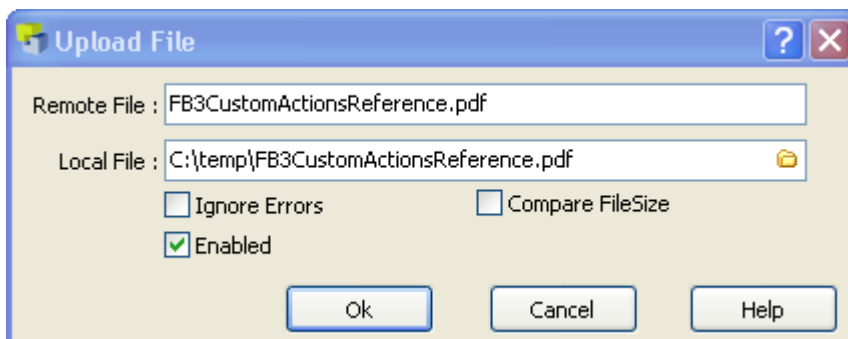
**Delete Directory :** Deletes the specified remote directory.

**Delete File :** Deletes the specified remote File.

**Rename File :** Renames the specified Remote File.

**Download File :** Downloads the specified remote file to the specified local file.

**Upload File :** Uploads the specified local file to the specified remote file.



#### Scripting Info

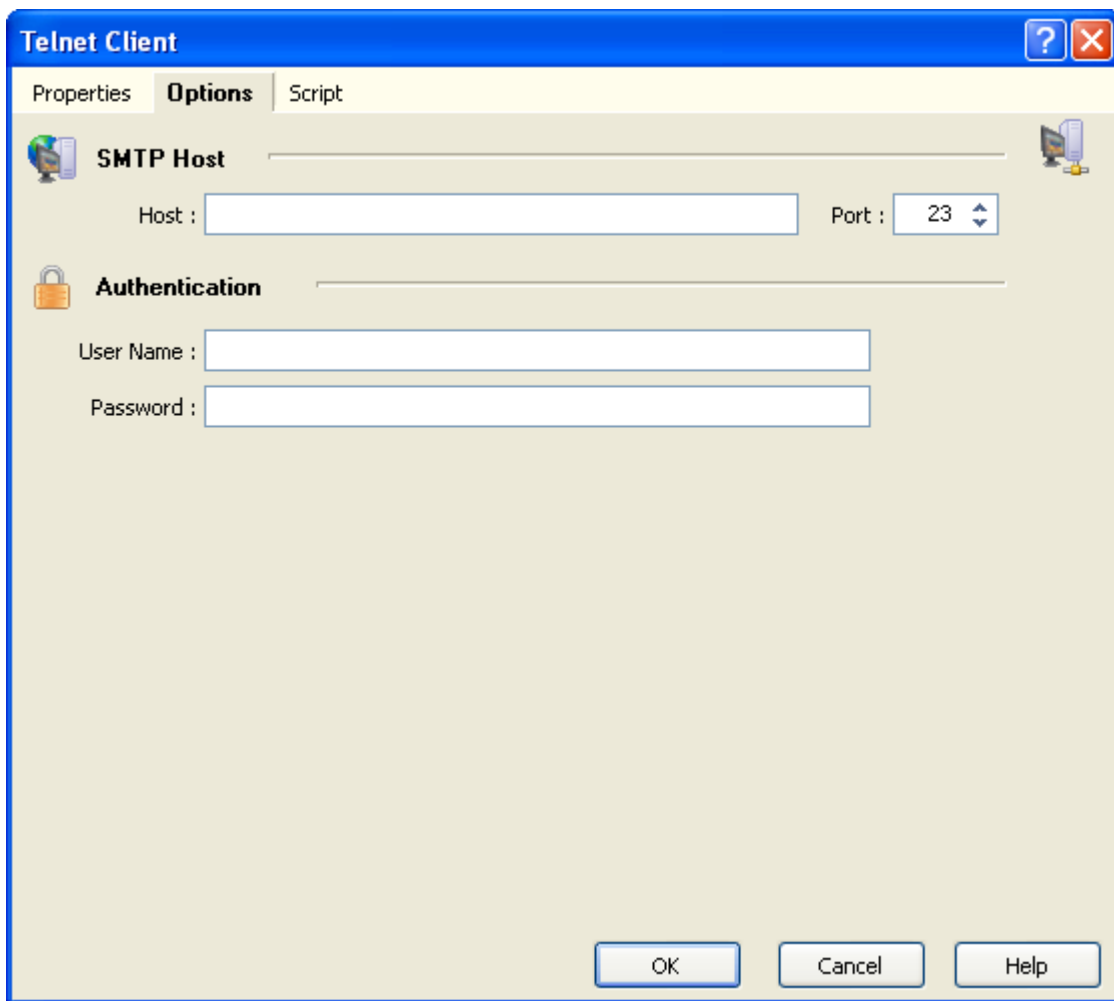
The Action properties available are :

**property** Host : WideString  
**property** Port : integer  
**property** UserID : WideString  
**property** Password : WideString  
**property** Passive : WordBool  
**property** CurrentDir : WideString  
**property** DetailedLogging : WordBool

This action uses the Open Source Indy Components, for more information see the Indy web site : <http://www.nevrona.com/Indy>

### 5.10.3 Telnet Client Action

This action provides a simple scriptable Telnet client.

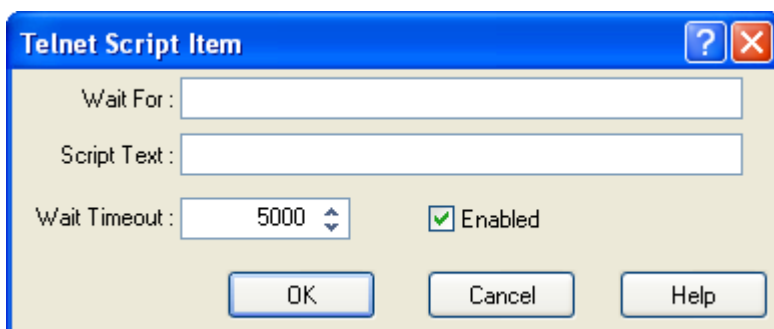


The **Telnet Client** dialog box has three tabs: **Properties**, **Options** (selected), and **Script**. The **Options** tab contains the following fields:

- SMTP Host**: A section header with a globe icon.
- Host**: A text input field.
- Port**: A spin box set to 23.
- Authentication**: A section header with a lock icon.
- User Name**: A text input field.
- Password**: A text input field.

At the bottom are **OK**, **Cancel**, and **Help** buttons.

The script items allow you to wait for a string and (optionally) respond with a string.



The **Telnet Script Item** dialog box contains the following fields:

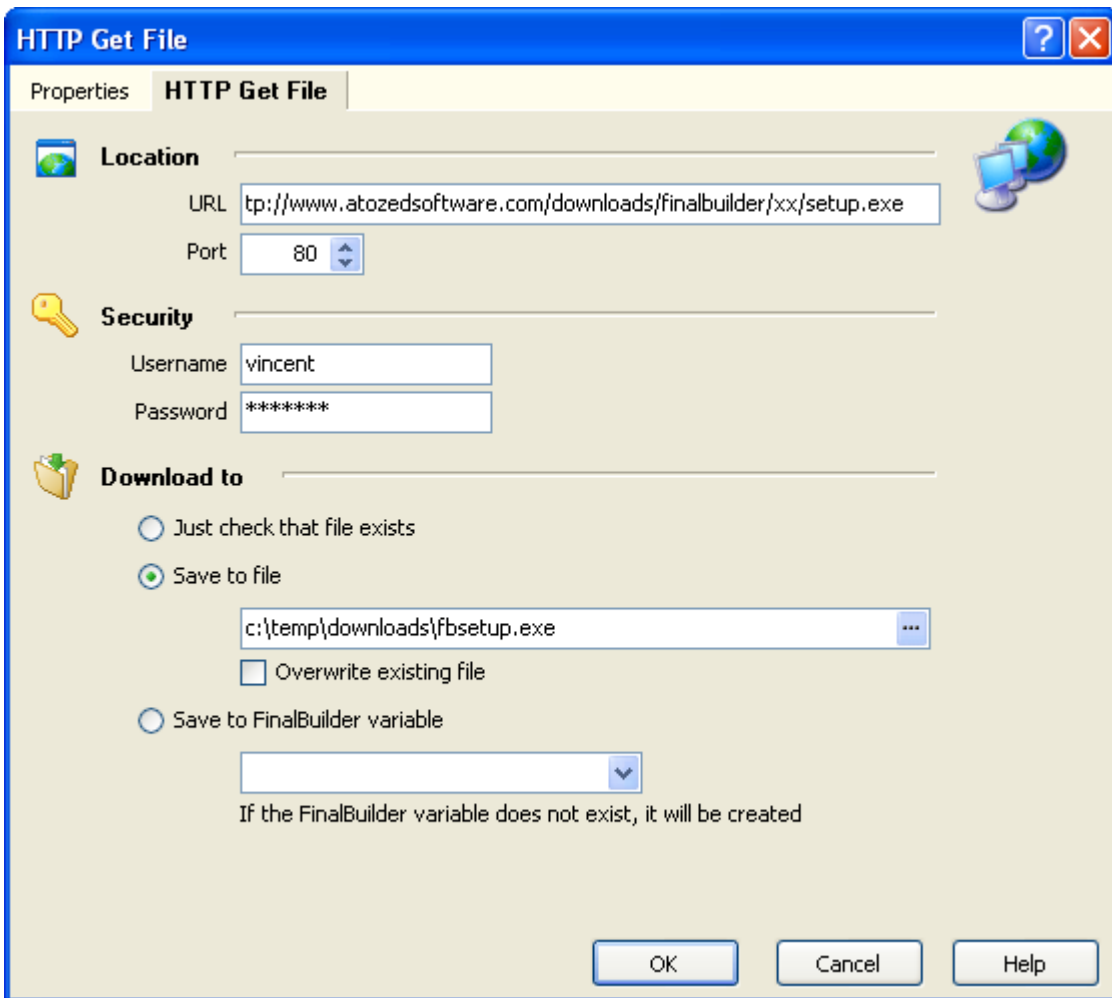
- Wait For**: A text input field.
- Script Text**: A text input field.
- Wait Timeout**: A spin box set to 5000.
- Enabled**: A checked checkbox.

At the bottom are **OK**, **Cancel**, and **Help** buttons.

To send the UserName set the ScriptText field to @USERNAME  
To send the Password set the ScriptText field to @PASSWORD

### 5.10.4 HTTP Get Action

This action enables you to download a file using the http protocol.



The screenshot shows the 'HTTP Get File' dialog box with the following fields and options:

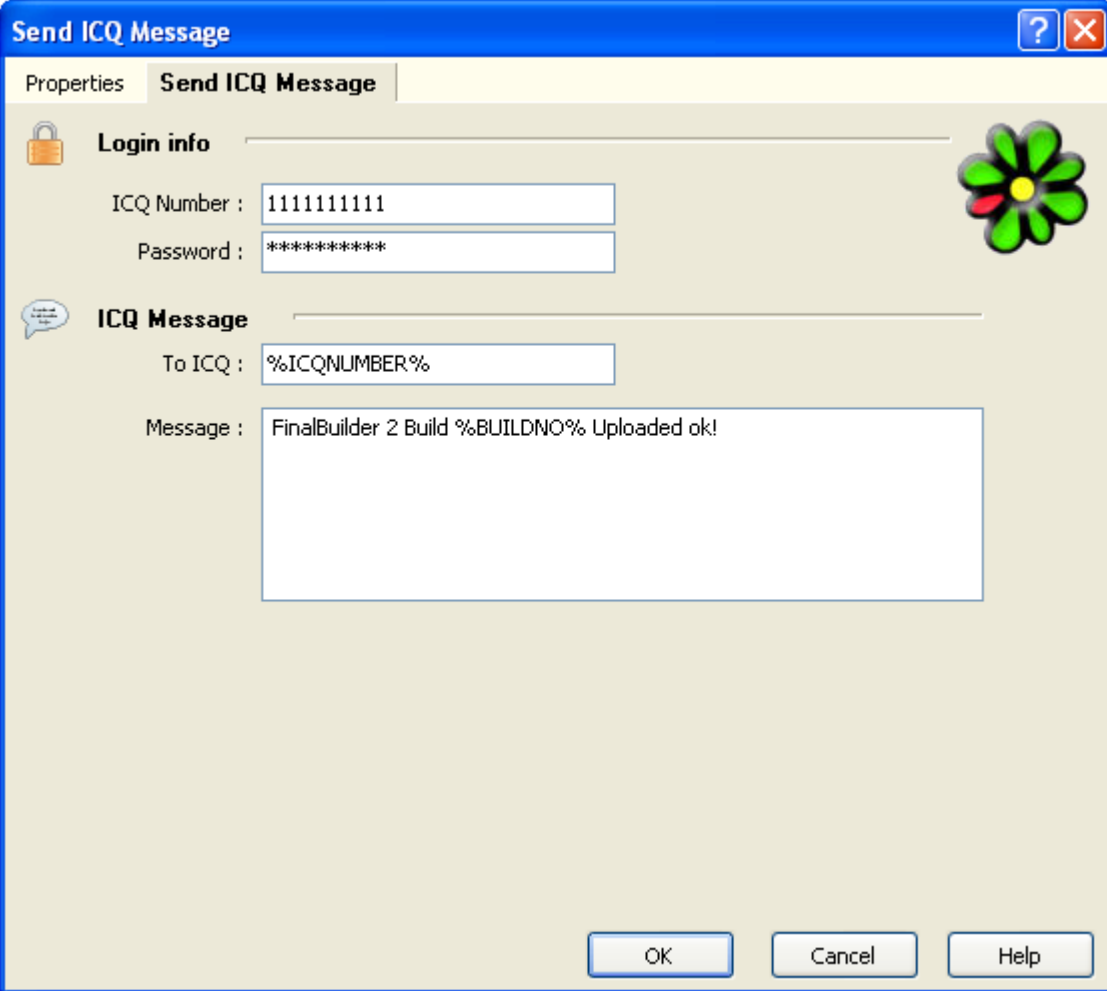
- Location:**
  - URL: `tp://www.atozedsoftware.com/downloads/finalbuilder/xx/setup.exe`
  - Port: `80`
- Security:**
  - Username: `vincent`
  - Password: `*****`
- Download to:**
  - ☐ Just check that file exists
  - ☒ Save to file
    - File path: `c:\temp\downloads\fbsetup.exe`
    - ☐ Overwrite existing file
  - ☐ Save to FinalBuilder variable
    - Variable name: (empty dropdown)
    - Note: If the FinalBuilder variable does not exist, it will be created

Buttons at the bottom: OK, Cancel, Help.

### 5.10.5 ICQ Action

This action enables you to send ICQ messages.





The image shows a Windows-style dialog box titled "Send ICQ Message". It has a blue title bar with a question mark icon and a close button. Below the title bar is a tabbed interface with two tabs: "Properties" and "Send ICQ Message", with the latter being selected. The dialog is divided into two main sections. The first section, "Login info", is marked with a lock icon and contains two text input fields: "ICQ Number" with the value "1111111111" and "Password" with the value "\*\*\*\*\*". The second section, "ICQ Message", is marked with a speech bubble icon and contains a "To ICQ" field with the value "%ICQNUMBER%" and a larger "Message" text area containing the text "FinalBuilder 2 Build %BUILDNO% Uploaded ok!". In the top right corner of the dialog, there is a green flower-like icon with a yellow center. At the bottom right, there are three buttons: "OK", "Cancel", and "Help".

### 5.10.6 NNTP News Post Action

This action allows you to post a message on one or more news servers.

**NNTP News Post**

Properties **NNTP** Message Attach

**NNTP Host**

Host : news.atozedsoftware.com Port : 119

Timeout : 10 seconds ( 0 = no timeout)

**Sender**

Sender Name : FinalBuilder News

Email Address : dontreply@atozedsoftware.com

Reply To :

**Newsgroups**

atozedsoftware.announce,atozedsoftware.finalbuilder

Separate multiple newsgroups with comma

**Authentication**

☐ Login

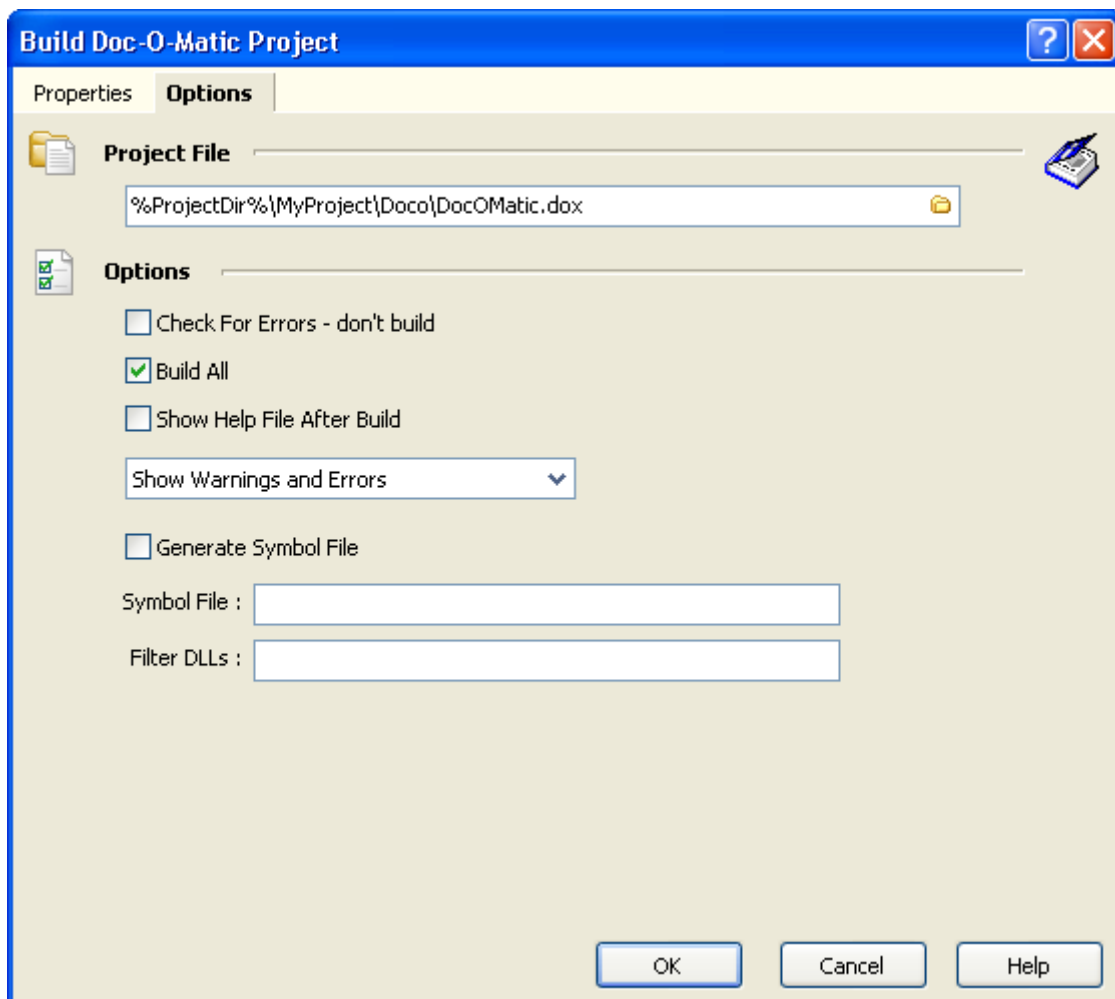
User Name : Password :

OK Cancel Help

## 5.11 Help Compilers

### 5.11.1 Build Doc-O-Matic Project

This action provides an interface with the Doc-O-Matic command line compiler.



**Project File :** The Doc-O-Matic project file.

**Check for Errors - Don't Build :** just that!

**Show Help File After Build :** Show the help file after build, not recommended

**Reporting Level :** Show All messages : Show Errors only : Show Hints, Warnings and Errors : Show Warnings and Errors.

**Generate Symbol File :** Alternate usage, generates a symbol file.

**Symbol File :** The fully qualified path to the generated symbol file.

### Scripting Info

The Action properties available are :

**property** CheckForErrors : WordBool

**property** BuildAll : WordBool

**property** WarningLevel : integer // valid values are : wShowAll, wShowHintsWarningsAndErrors, wShowWarningsAndErrors , wShowErrorsOnly

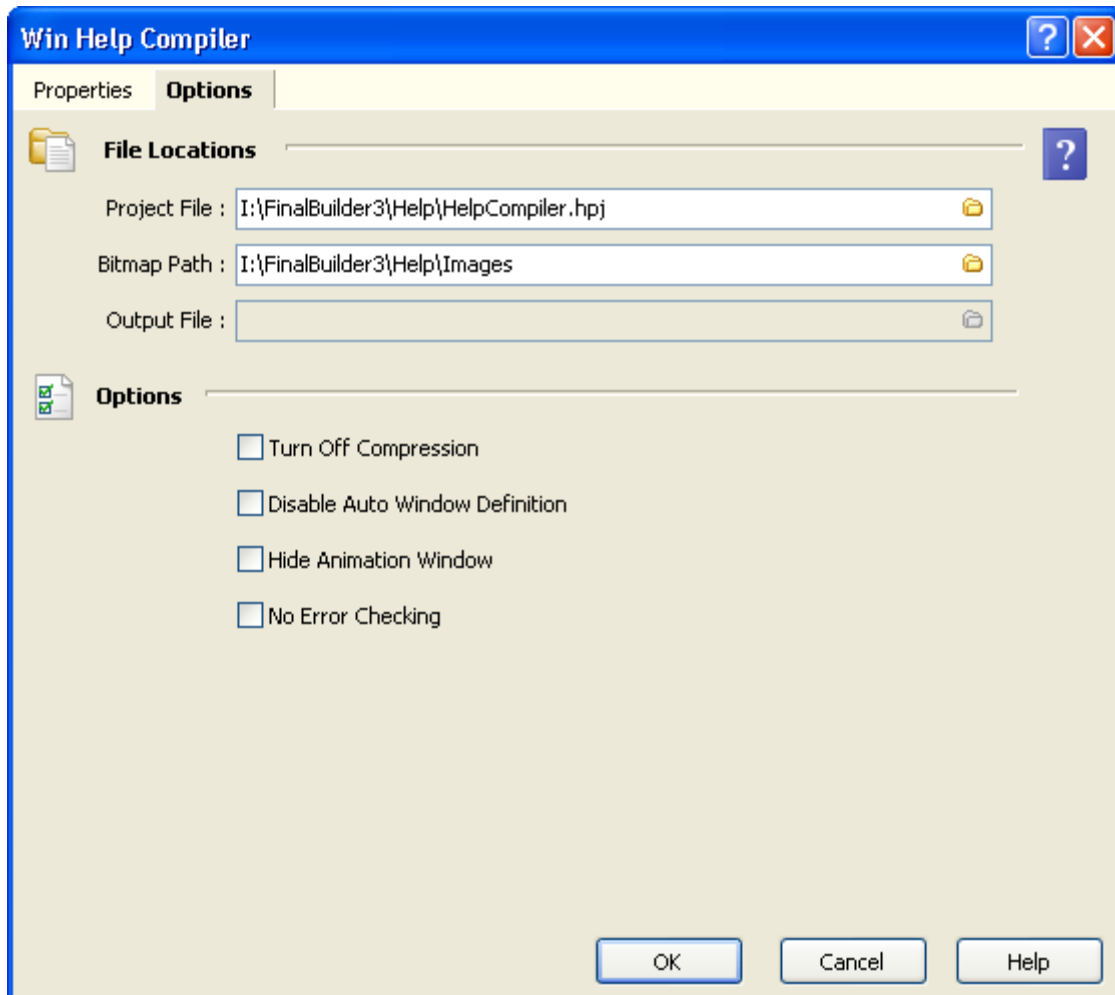
**property** ProjectFile : WideString

**property** ShowAfterBuild : WordBool  
**property** SymbolFile : WideString  
**property** GenerateSymbolFile : WordBool

Doc-O-Matic can be found at <http://www.toolsfactory.com/>

### 5.11.2 WinHelp Compiler

This action provides an interface to the Windows Help Compiler. Note that to use this action, you should first set the WinHelp compiler location in the Options dialog.



**Project File :** The fully qualified path to the winhelp project file (\*.hpj) or Rich Text File (\*.rtf)

**Bitmap Path :** The search path for bitmaps referenced in the help project. Separate entries with a semi-colon.

**Turn Off Compression :** Instructs the compiler not to compress the help file.

**Hide Animation Window :** Hides the animated window while compiling

**Disable Auto Window Definition :** Prevents the automatic Creation of window definition, if a window specified in a hotspot or macro was not defined in the project file.

**No Error Checking** : Specifies that there are no errors in the rtf files. Some error checking will be turned off to speed up the compilation process. This option should only be used where compilation speed is crucial.

**Output File** : Specifies the name of the hlp file to create. Use this only when not compiling an hpj file.

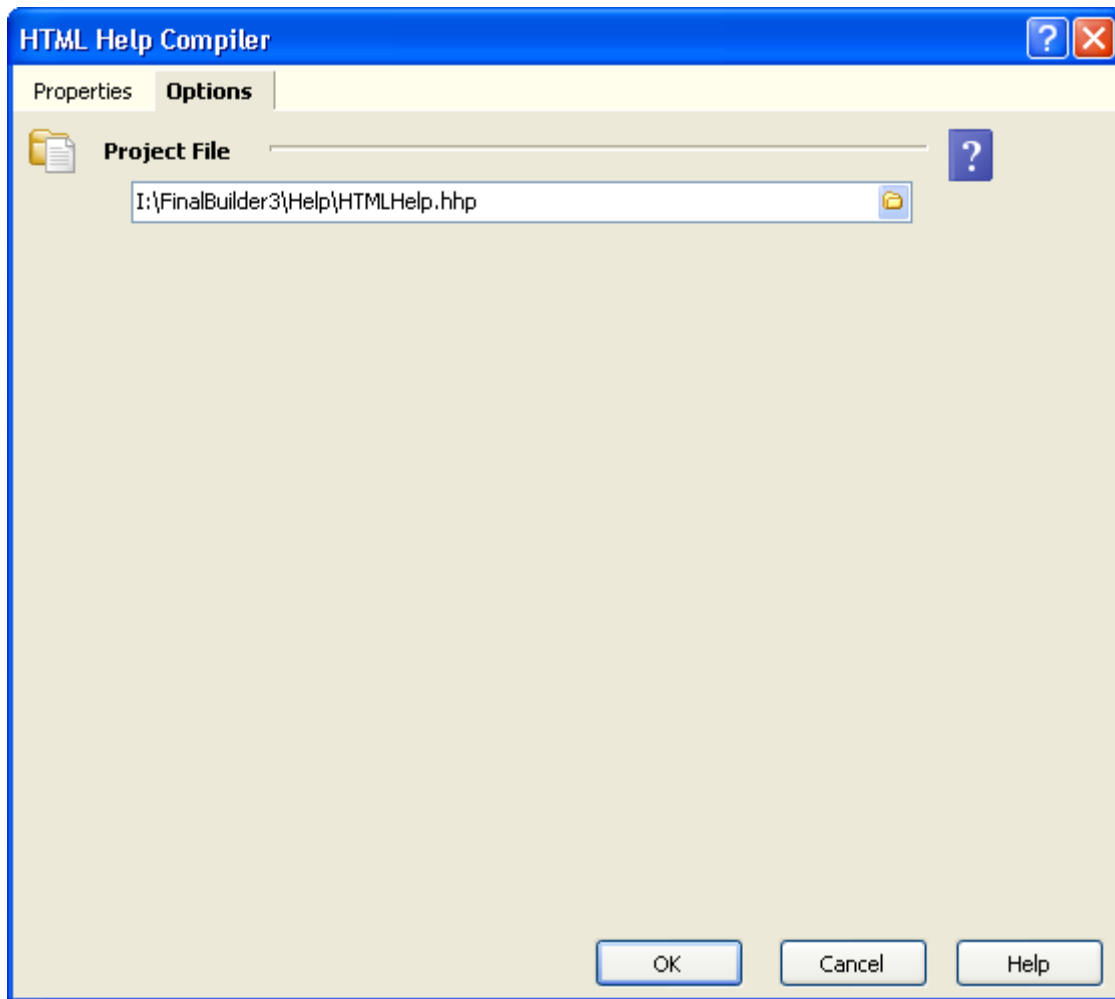
### Scripting Info

The Action properties available are :

**property** ProjectFile : WideString  
**property** BitmapPath : WideString  
**property** TurnOffCompression : WordBool  
**property** HideAnimationWin : WordBool  
**property** DisableAutoWinDef : WordBool  
**property** NoErrorCheck : WordBool  
**property** OutputFile : WideString

### 5.11.3 HTML Help Compiler

This action provides an interface to the HTML Help Compiler. Note that to use this action, you should first set the HTML Help Compiler location in the Options dialog.



**Project File** : The fully qualified path to the html help project file (hhp).

### Scripting Info

The Action properties available are :

**property** ProjectFile : WideString

## 5.11.4 Help & Manual 3 Action

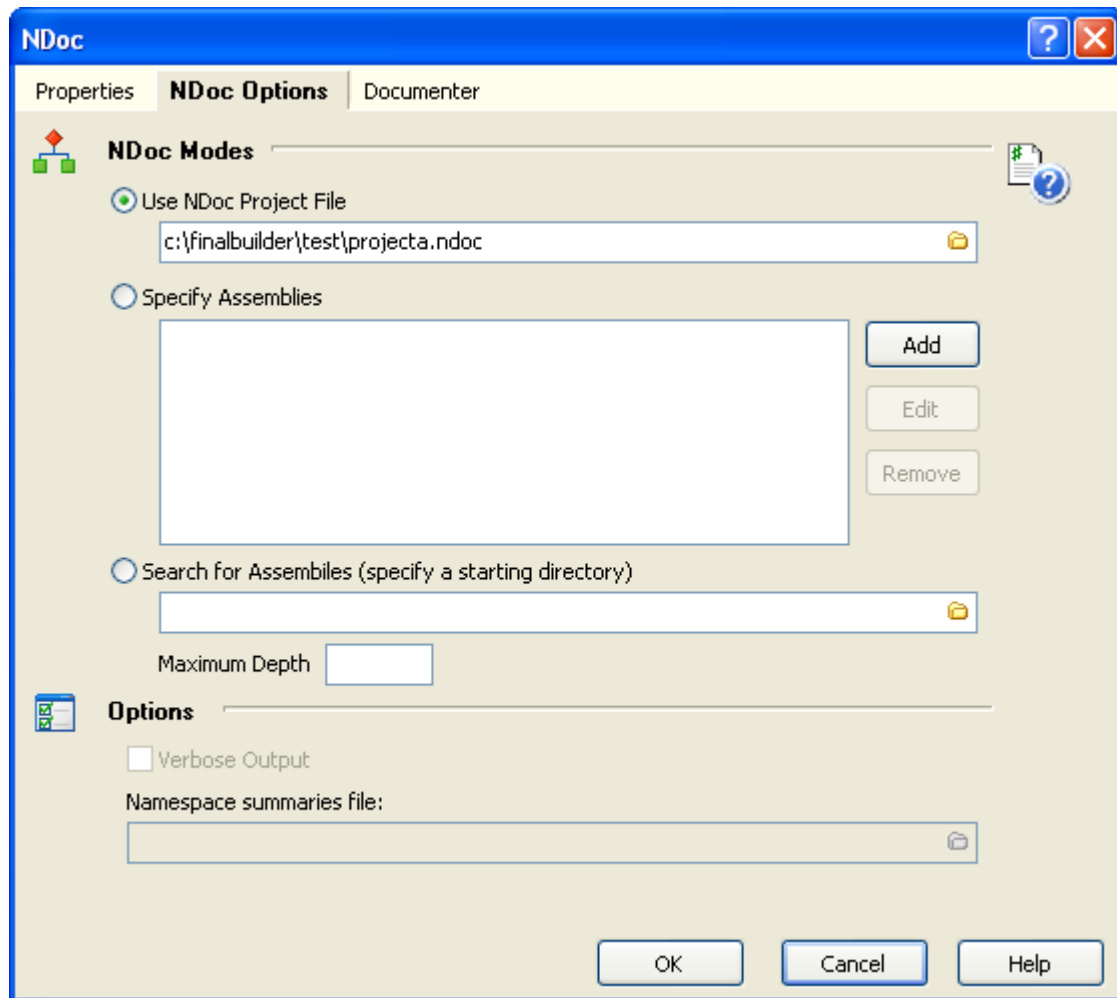
This Action enables the automated builds of Help&Manual 3.0 projects - you can build multiple output formats in one action. Help & Manual 3.0 is a product of [ECSoftware](#) and is the tool we use and recommend for help file and manual generation.

## 5.11.5 NDoc Action

NDoc generates class libraries documentation from .NET assemblies and the XML

documentation files generated by the C# compiler (or an add-on tool for VB.NET).

NDoc uses add-on documenters to generate documentation in several different formats, including the MSDN-style HTML Help format (.chm), the Visual Studio .NET Help format (HTML Help 2), and MSDN-online style web pages.



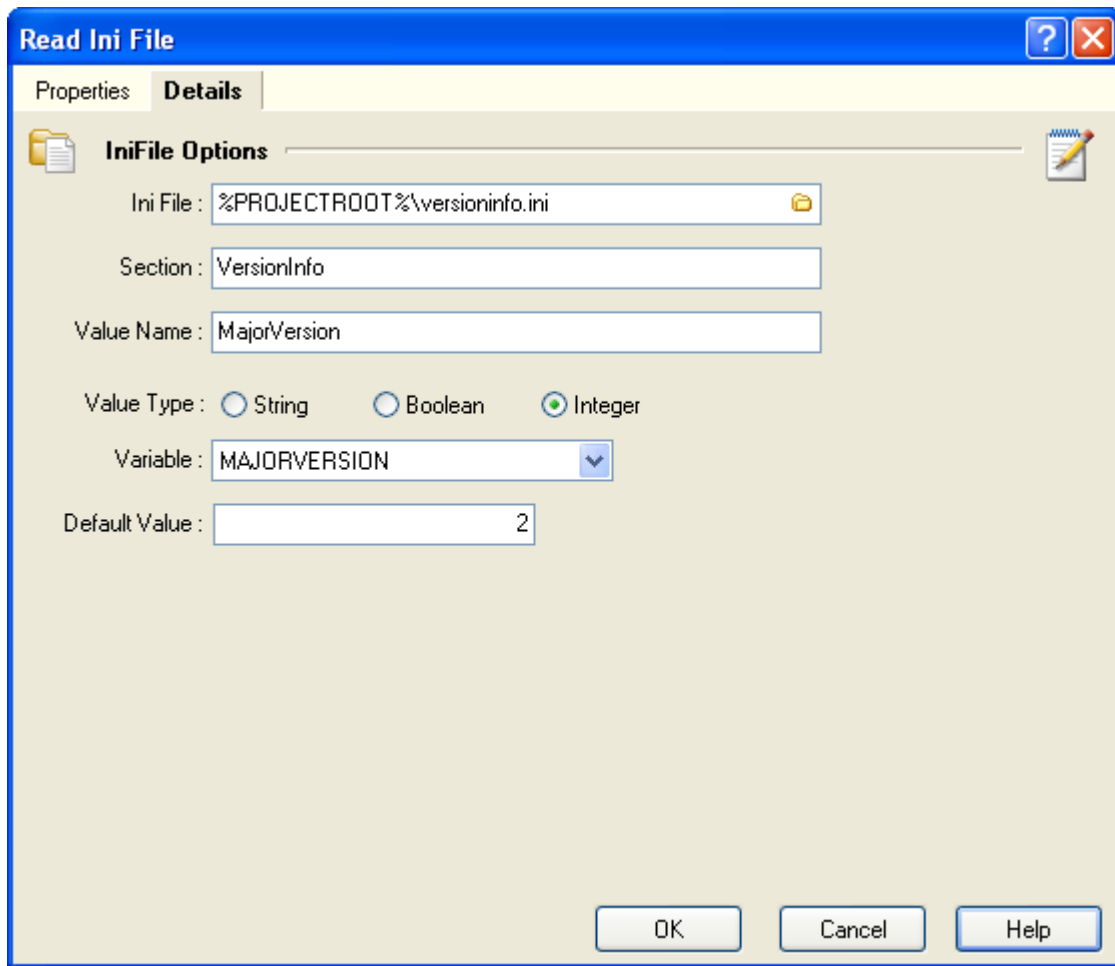
For more information see the NDoc homepage:

<http://ndoc.sourceforge.net/wiki>

## 5.12 Ini Files & Registry

### 5.12.1 Read Ini File

This action allows you to read values from an ini file into FinalBuilder variables.



Ini File : The Fully qualified path to the ini file.

Section : The ini File section where the value name is found.

Value Name : The name of the value to read

Variable : The name of the FinalBuilder Variable to read the value into.

Default Value : The default value to use if the value name is not found in the ini file.

### Scripting Info

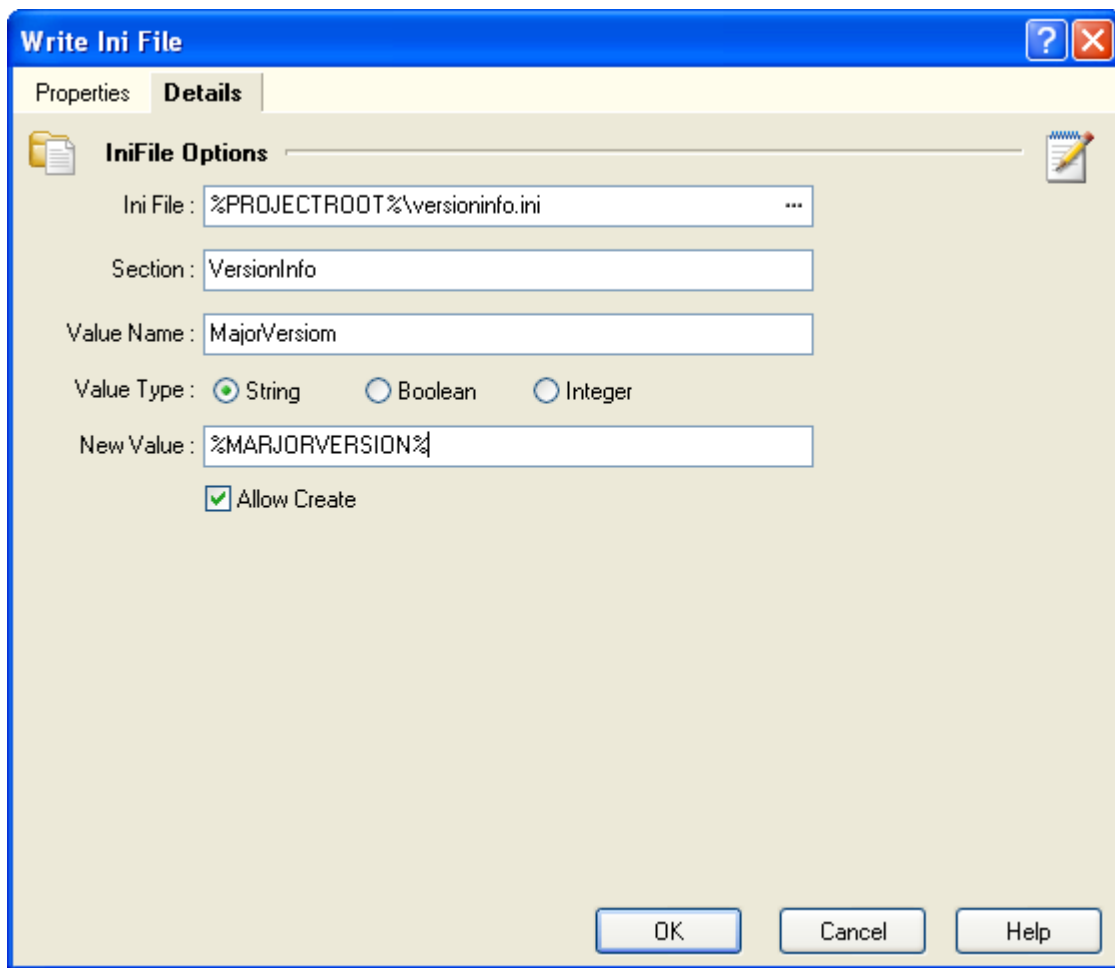
The Action properties available are :

**property** IniFile : WideString  
**property** Section : WideString  
**property** ValueName : WideString  
**property** VariableName : WideString  
**property** DefaultValue : WideString

## 5.12.2 Write Ini File

This action allows you to write values to an ini file.





**Ini File :** The Fully qualified path to the ini file.

**Section :** The ini File section where the value name is found.

**Value Name :** The name of the value to write

**New Value :** The value to write to the ini file. You can use FinalBuilder variables here.

**Allow Create :** Allow the ini file to be created if it does not already exist.

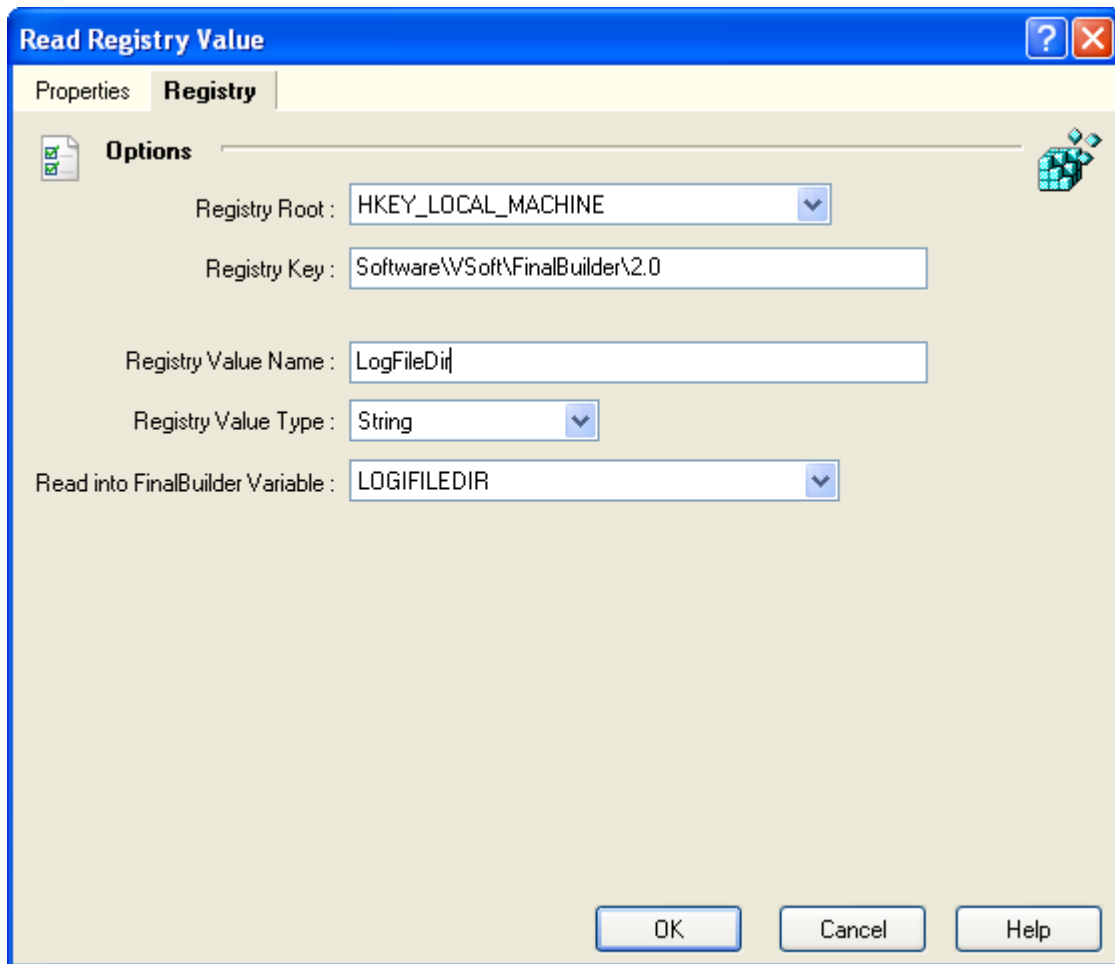
### Scripting Info

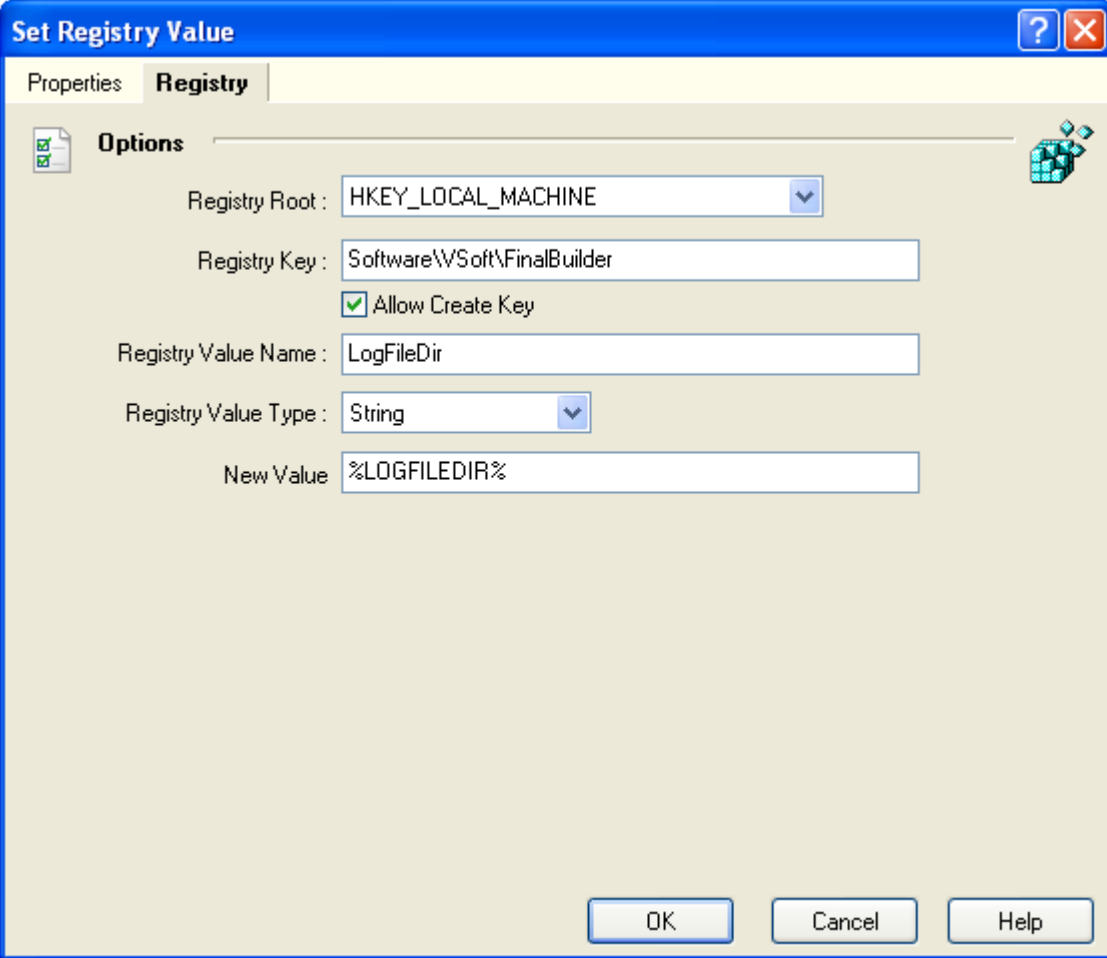
The Action properties available are :

**property** IniFile : WideString  
**property** Section : WideString  
**property** ValueName : WideString  
**property** NewValue : WideString  
**property** AllowCreate : WordBool

### 5.12.3 Read/Set/Delete Registry Value

These actions allow you to Read a Registry Value into a FinalBuilder variable, Set a Registry Value or Delete a Registry Value.





The image shows a Windows-style dialog box titled "Set Registry Value". It has a blue title bar with a question mark icon and a close button. Below the title bar is a tabbed interface with "Properties" and "Registry" tabs. The "Registry" tab is active. Under the "Options" section, there are several input fields and a checkbox. The "Registry Root" is set to "HKEY\_LOCAL\_MACHINE". The "Registry Key" is "Software\VSof\FinalBuilder". The "Allow Create Key" checkbox is checked. The "Registry Value Name" is "LogFileDir". The "Registry Value Type" is set to "String". The "New Value" is "%LOGFILEDIR%". At the bottom right, there are three buttons: "OK", "Cancel", and "Help".

**Set Registry Value**

Properties **Registry**

**Options**

Registry Root : HKEY\_LOCAL\_MACHINE

Registry Key : Software\VSof\FinalBuilder

☒ Allow Create Key

Registry Value Name : LogFileDir

Registry Value Type : String

New Value : %LOGFILEDIR%

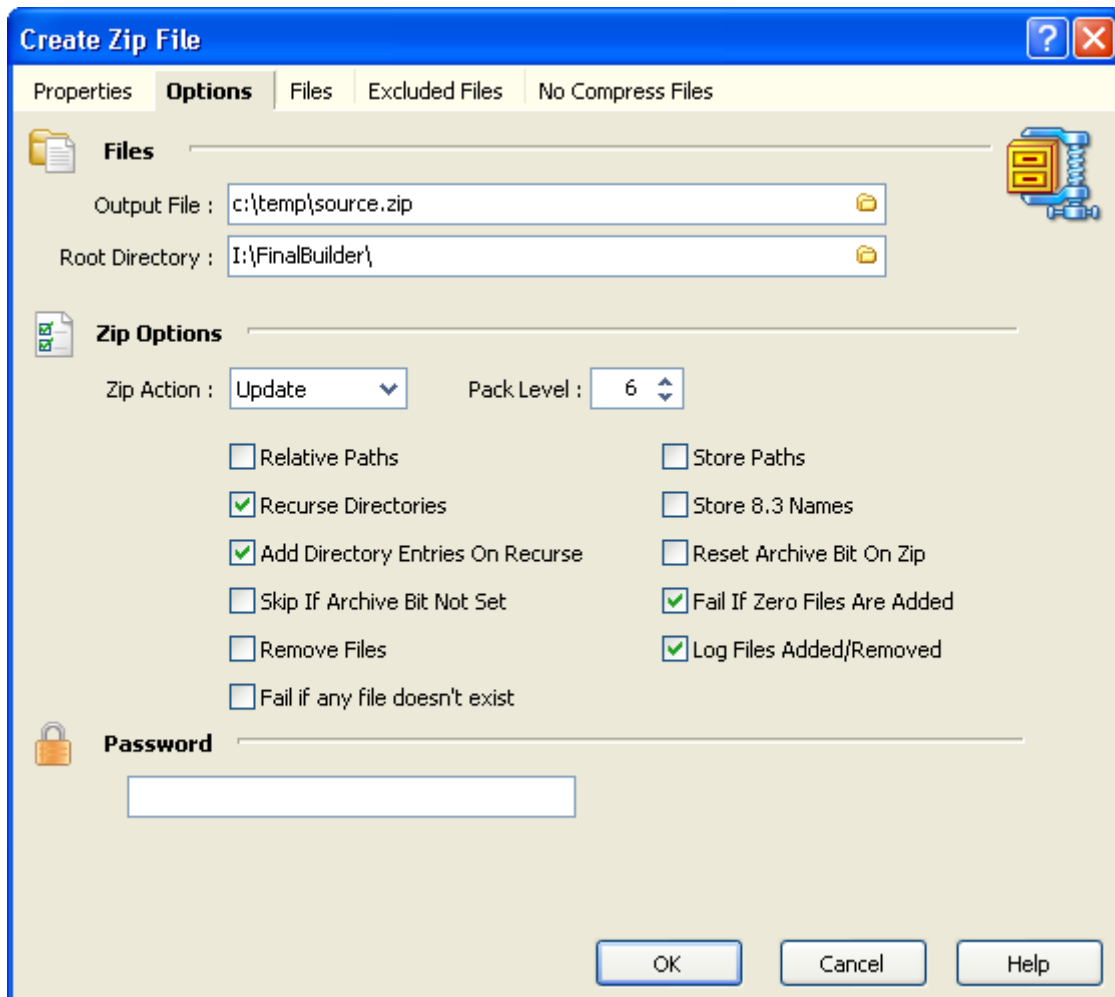
OK Cancel Help



## 5.13 Archiving

### 5.13.1 Create Zip File

This action provides the ability to create Zip files.



**Output File :** Output FileName

**Zip Action :** The ZipAction property determines whether files will be replaced in a zip or not. If ZipAction is set to zaUpdate then a zip entry will only be replaced if the disk file is newer than the zip entry. If ZipAction is set to zaReplace, then the zip entry will be replaced by the disk file regardless of the file dates. A ZipAction of zaFreshen is the same as zaUpdate, except that filenames that do not match any entries already in the zip file will be ignored and not added to the zip.

**Pack Level :** The PackLevel property determines how hard the compression algorithm will try to compress files. This property can be given a value from 0 through 9. A value of 0 is no compression at all (STORED) which is useful for adding things like other zip files (which will compress very little if any) to an archive. A value of 1 will compress the fastest, but the compression ratio will be the lowest. A value of 9 will compress the slowest, but the compression ratio will be the highest. You may specify particular files that you do not want to try to compress by adding their filespecs or wildcards to the NoCompressList.

**Relative Paths :** The Relative Paths property should be set to True if you wish to save path information but only wish to save path information relative to a specified directory. The only path information that is saved is for subdirectories below the specified directory. This is similar to the Relative Path option of PKZip for Windows. Whenever you set Relative Paths property to True, the Recurse Property and the Store Paths Property are automatically set to True also. Likewise, if the Store Paths Property is set to False, then the Relative Paths Property is automatically set to False also.

**Store Paths :** The StorePaths property, if set to True, will cause path information to be stored with the zip entry. If the StoreVolumes Property is set to True then the entire path will be stored. If the StoreVolumes Property is False, then only the path information will be stored. If StorePaths is False, then only the filename itself is stored. Note that if the RelativePaths Property is set to True, then this StorePaths property will automatically be set to True also. Likewise, if the StorePaths property is ever set to False, then both the RelativePaths and the StoreVolumes Property will automatically be set to False.

**Recurse Directories :** The Recurse property determines whether subdirectories will be recursed to look for files to be compressed when zipping with a wildcard mask. Set to true if you wish subdirectories to be traversed. If this value is set to True, and a wildcard mask is specified in the FilesList without any path information, then the value of the RootDir Property will determine which directory zipping will start in. Results may be unexpected or even bad if you do not supply path information either in the FilesList or the RootDir Property.

**Store 8.3 Names :** When set to True, this will force any long file and pathnames to be stored in DOS 8.3 format. This is useful if you plan to unzip the files onto a WIN3.X system where long filenames are not valid.

**Add Directory Entries On Recurse :** If Add Directory Entries On Recurse is True, then when you do a recursive search through subdirectories (Recurse = True) a separate entry will be made in the archive for each directory. This will allow even empty directories to be restored. If Add Dir Entries On Recurse is False, path information will still be stored with each file that is compressed, but a separate entry will not be inserted for the directories.

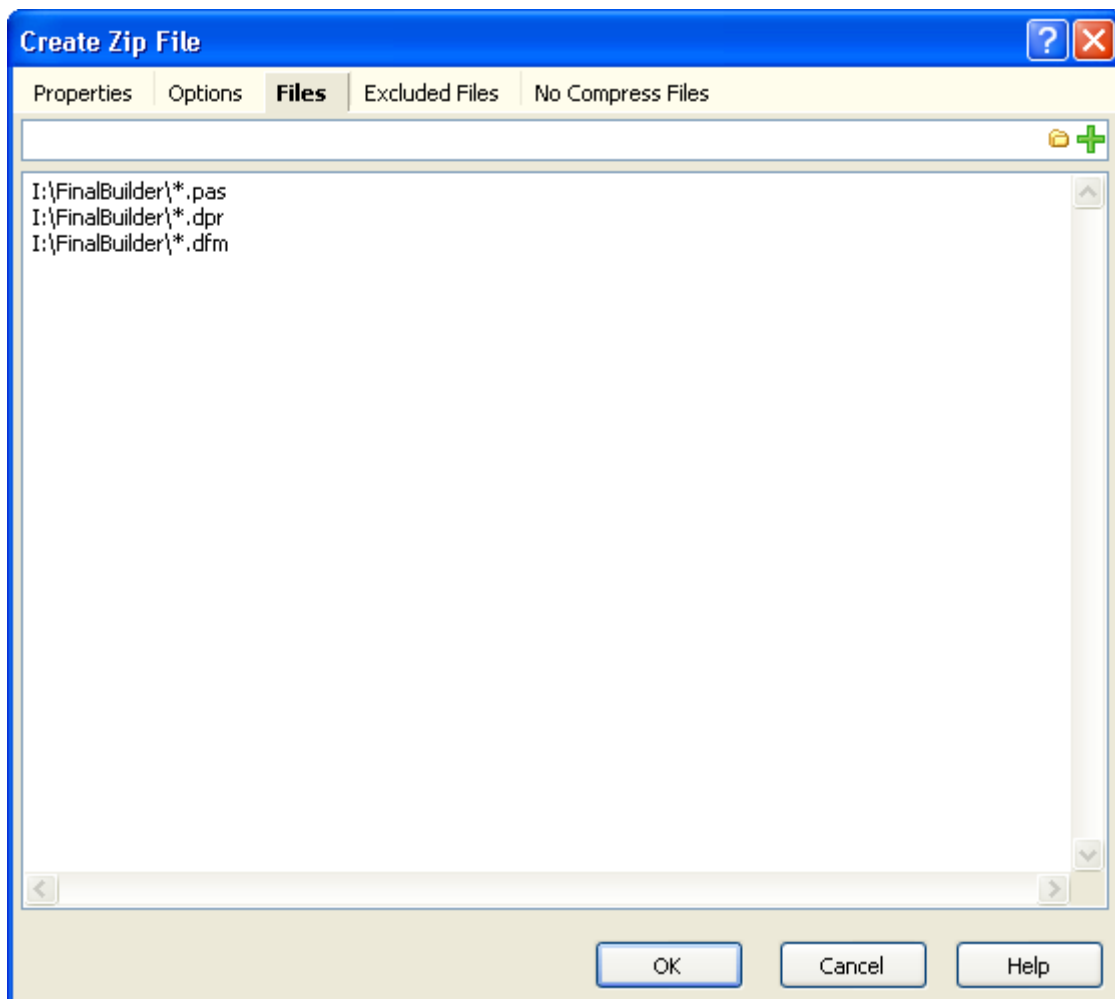
**Root directory :** The RootDir property determines where zipping will start for any wildcard entries or filenames in the FilesList Property that do not already include path information. Essentially, the value of RootDir will be prepended to anything in the FilesList that does not have any path information when zipping. Also, when storing relative path information using the RelativePaths Property, you must use this RootDir property to specify the directory from where path information will begin being saved.

**Skip if Archive bit Not Set :** Setting this to True will cause files that do not have their Archive Bit set (turned on) to be skipped during zip operations. Therefore, while this is set to True, only files with their Archive attribute turned on will be zipped.

**Reset Archive bit On zip :** Setting this to True will cause the Archive Bit for each file to be reset (turned off) after being zipped.

**Remove Files :** The Remove Files property, if set to True, will cause the original disk files that were added to the zip file to be deleted from the disk, in effect, moving the files into the zip file. USE THIS OPTION WITH EXTREME CAUTION. If an exception occurs during processing, files will not be Deleted.

**Fail If Zero Files are Added :** Fail If Zero Files are added to zip file. This will cause the run to stop unless the ignore failure property is set.



The Files Section allows you to specify the files that will be added to the zip file. You can use Wildcards (\*, ?) and FinalBuilder variables when specifying the files.

The Excluded Files section allows you specify file that should not be included in the resulting zip file. You can use Wildcards (\*, ?) and FinalBuilder variables when specifying the files.

The No Compress Files allows you to specify files that should be added to the zip file but should not be compressed. This is useful for adding other zip files. You can use Wildcards (\*, ?) and FinalBuilder variables when specifying the files.

### Scripting Info

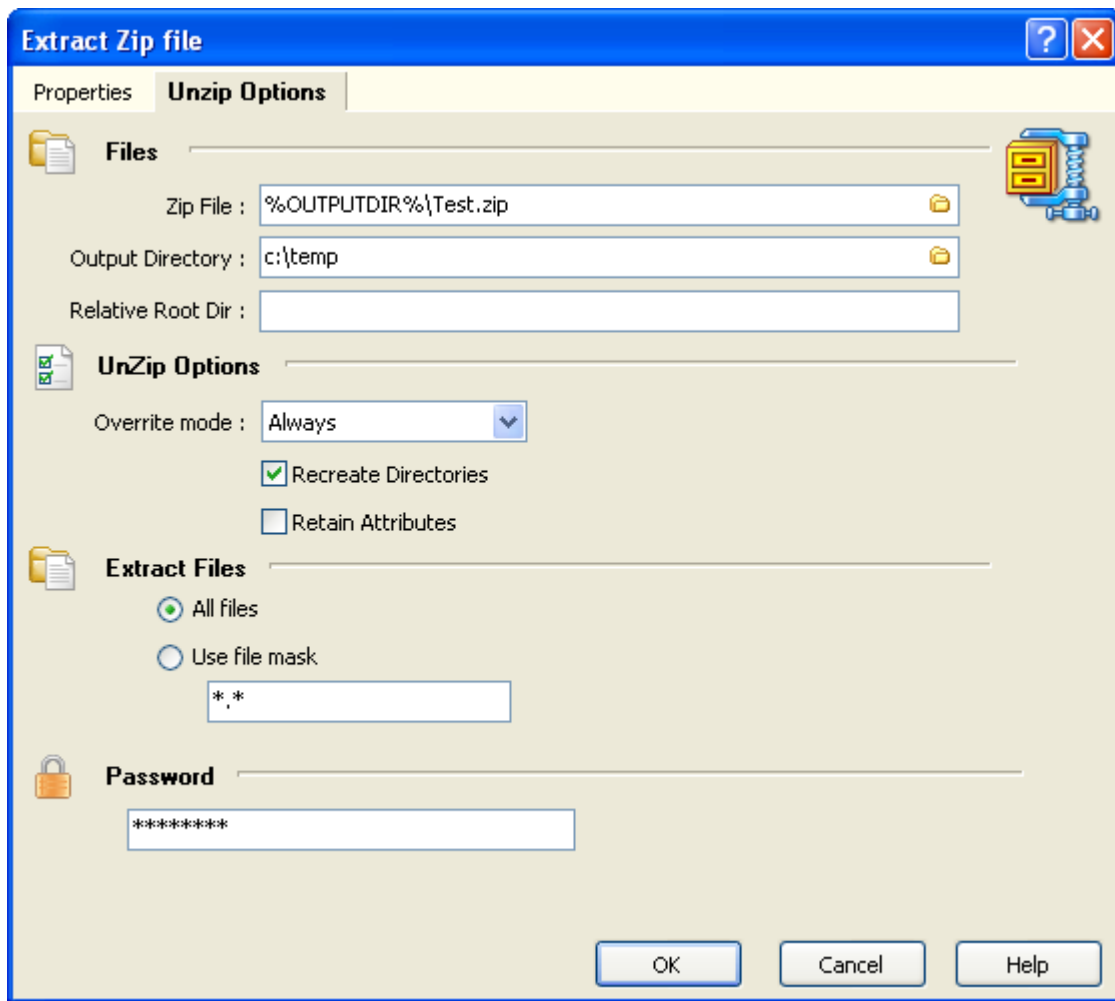
The Action properties available are :

- property** OutputFileName : WideString
- property** PackLevel: Integer // 0 - 9
- property** Recurse: WordBool
- property** Dispose: WordBool
- property** StorePaths: WordBool
- property** RelativePaths: WordBool
- property** Store83Names: WordBool
- property** SkipIfArchiveBitNotSet : WordBool
- property** ResetArchiveBitOnZip: WordBool

**property** AddDirEntriesOnRecurse: WordBool  
**property** FailIfZeroFiles : WordBool  
**property** RootDir : WideString

### 5.13.2 Extract Zip File Action

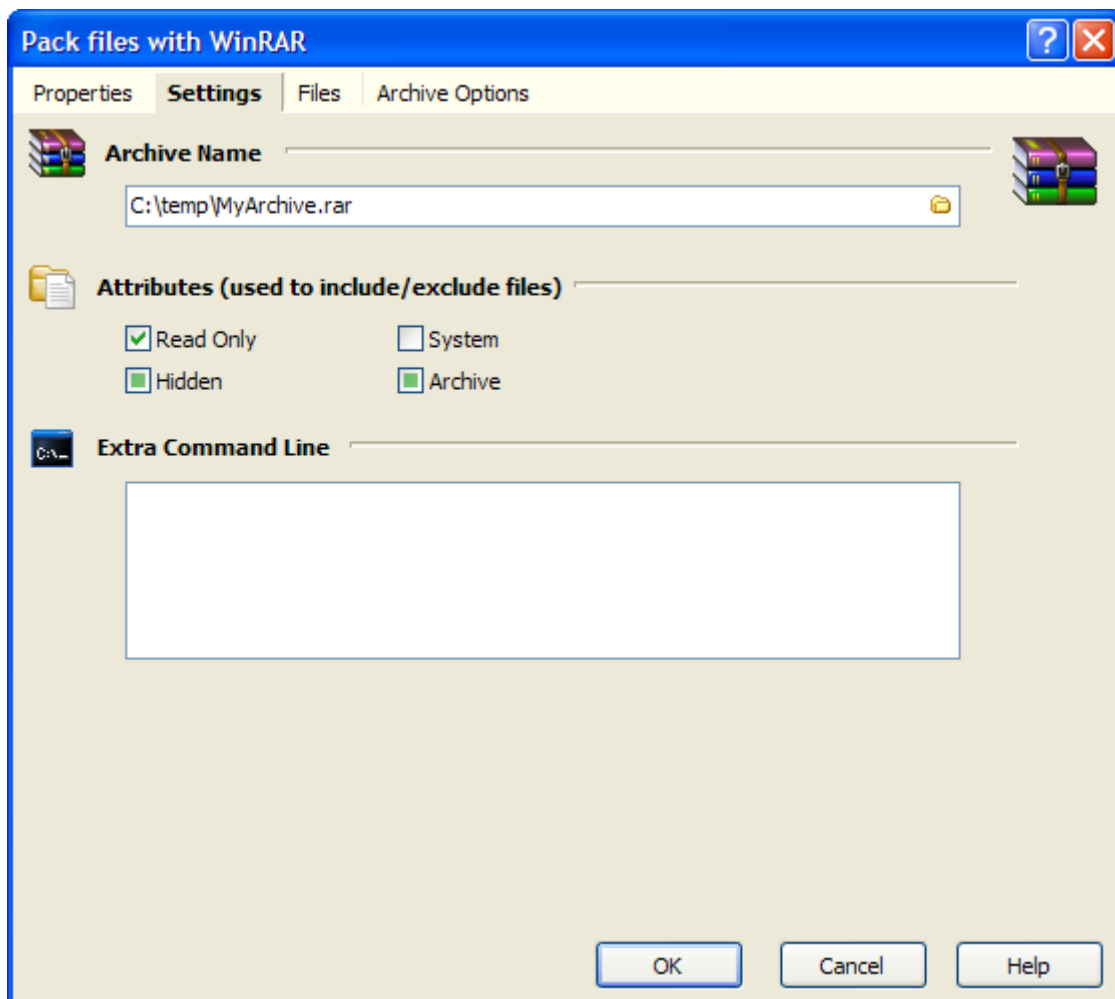
This action allows you to extract files from a zip archive file. You can choose which files to extract using file masks or just extract all files.



### 5.13.3 WinRAR Action

The WinRAR action enables you create RAR archives using WinRAR.





**Archive Name** - specify the name of the archive to create

**Attributes** - In the above screenshot, only files with the Read Only attribute, and not system files will be added to the archive.

**Extra Command Line** - If there are any WinRAR options which the FB action doesn't surface, then you can manually specify them

#### 5.13.4 7Zip

The 7Zip actions enable you to automate archive operations as part of your build process.

The archive operations available are:

Create Archive (supports Zip, 7z, GZip, BZip2, TAR)

Test Archive (supports Zip, 7z, GZip, BZip2, TAR)

List files in Archive (supports Zip, 7z, GZip, BZip2, TAR, RAR, ARJ, CAB, CPIO, RPM, DEB, SPLIT)

Extract Archive (supports Zip, 7z, GZip, BZip2, TAR, RAR, ARJ, CAB, CPIO, RPM, DEB, SPLIT)

Update Archive (supports Zip, 7z, GZip, BZip2, TAR)

Delete file from Archive (supports Zip, 7z, GZip, BZip2, TAR)

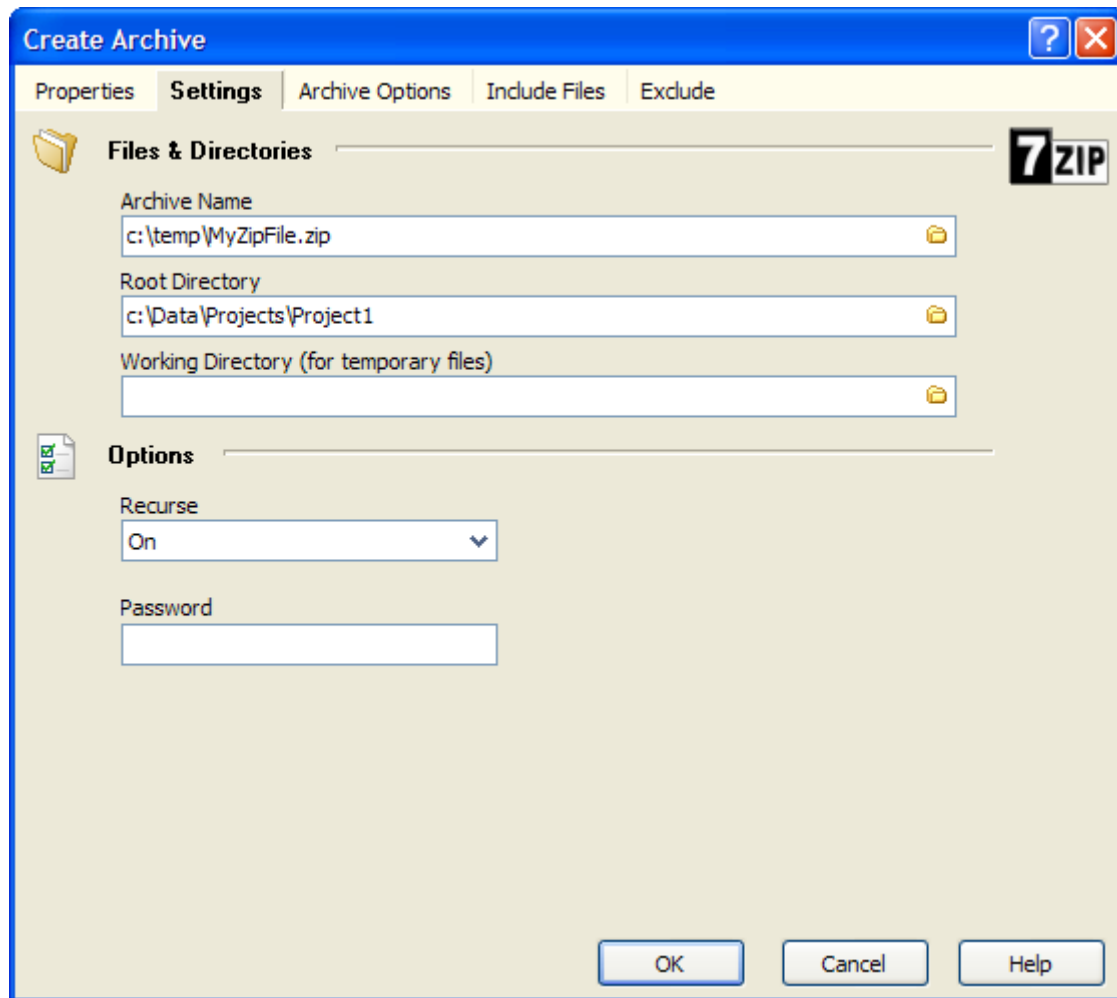
These actions require the 7Zip tool, available at <http://www.7-zip.org/>

#### 5.13.4.1 Create Archive

The Create Archive actions allows you to create archives using any of the following formats:

Zip, 7z, GZip, BZip2, TAR

More Info on the 7Zip based actions



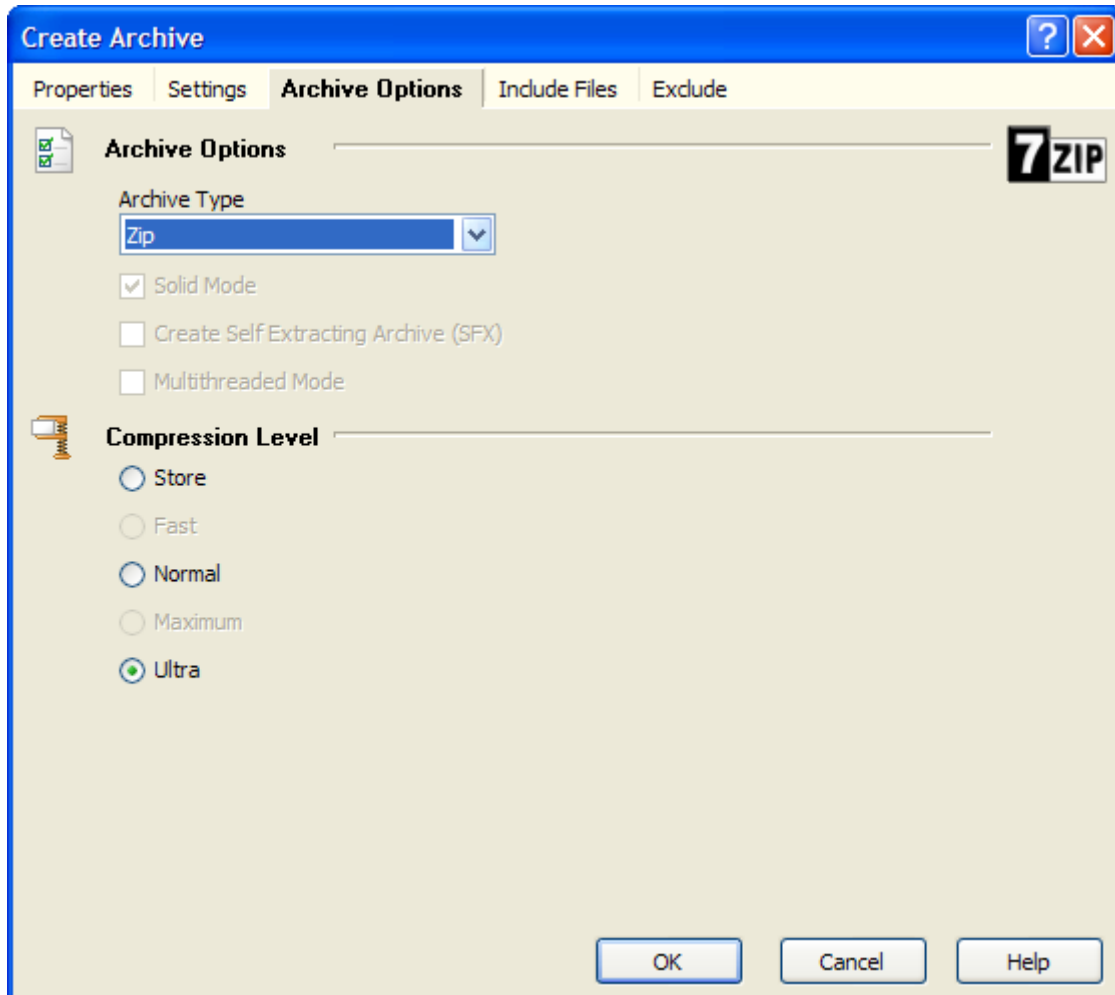
**Archive Name** - specify the name of the new archive. The file extension should match the type of archive you are creating (eg. ZIP)

**Root Directory** - you can optionally set this to a directory so that the included files can be relative to this directory

**Working Directory** - any temporary files will be placed in this directory

**Recurse** - specify how it should deal with sub-folders

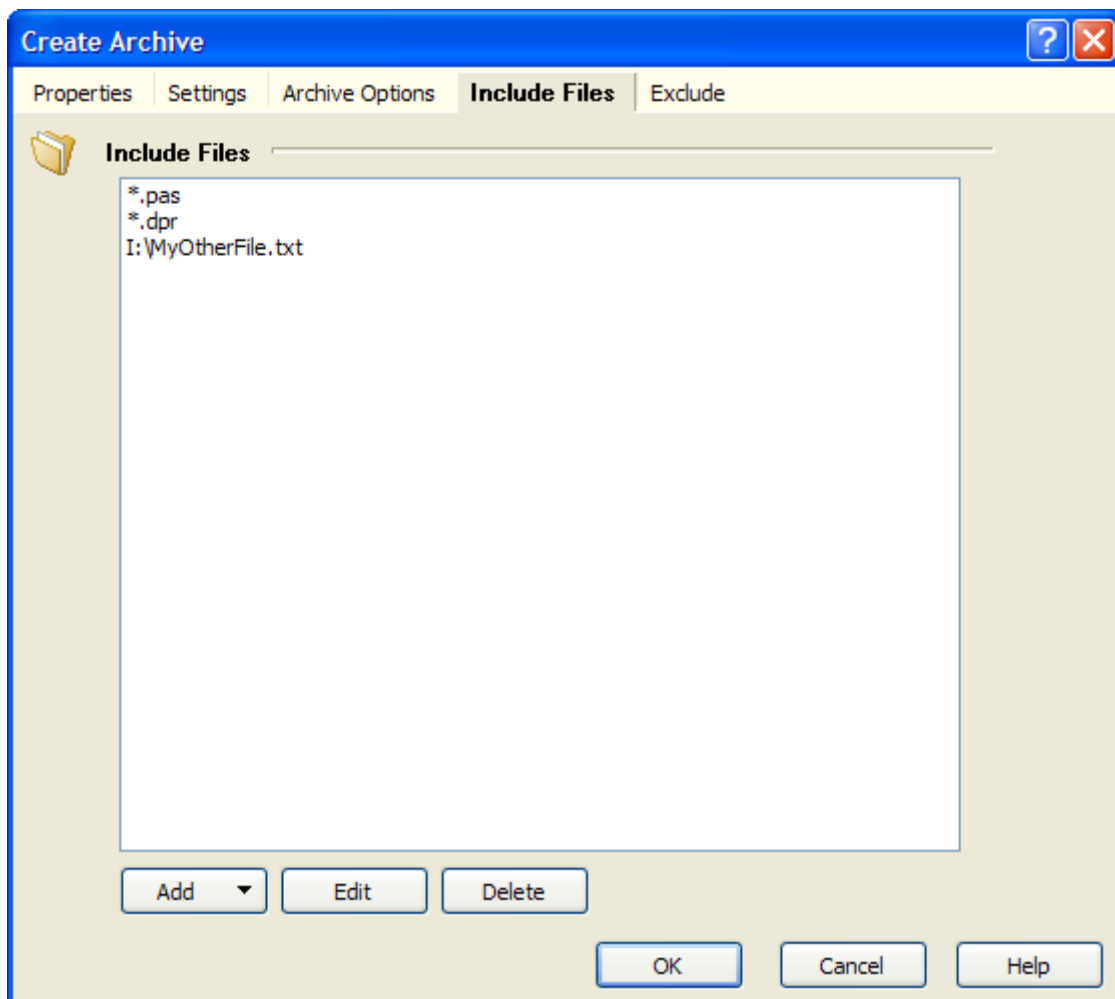
**Password** - specify a password to protect the archive. You'll need to supply this password to decrypt the archive



**Archive Type** - specify the archive type you want to create. It should match the file extension of the archive name.

**Solid Mode, SFX, and Multithreaded** are modes available if you choose the 7zip format

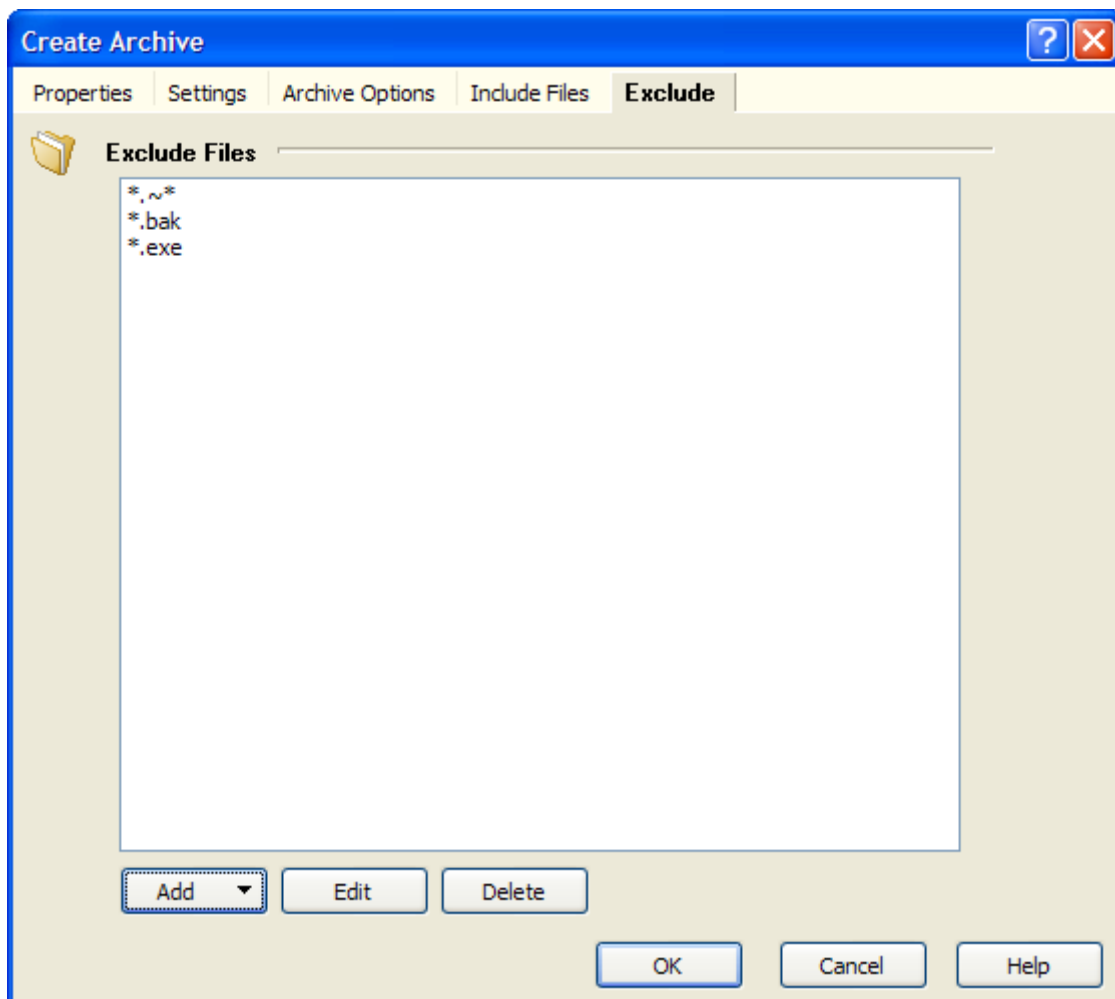
**Compression Level** - specify which compression level you require. Higher compression levels require more CPU and memory. Some compression formats only support a subset of the available compression levels.



Specify the files to include in the new archive.

If you don't specify a fully qualified filename, then the working directory must be set on the Settings tab.

The add button allows you to add a file, folder, or Other. Other is typically used to enter a wildcard filespec, such as \*.txt



Specify any files or filespecs to exclude from the archive.

#### 5.13.4.2 Test Archive

The Test Archive actions allows you to test the integrity of an archive in any of the following formats:

Zip, 7z, GZip, BZip2, TAR

More Info on the 7Zip based actions

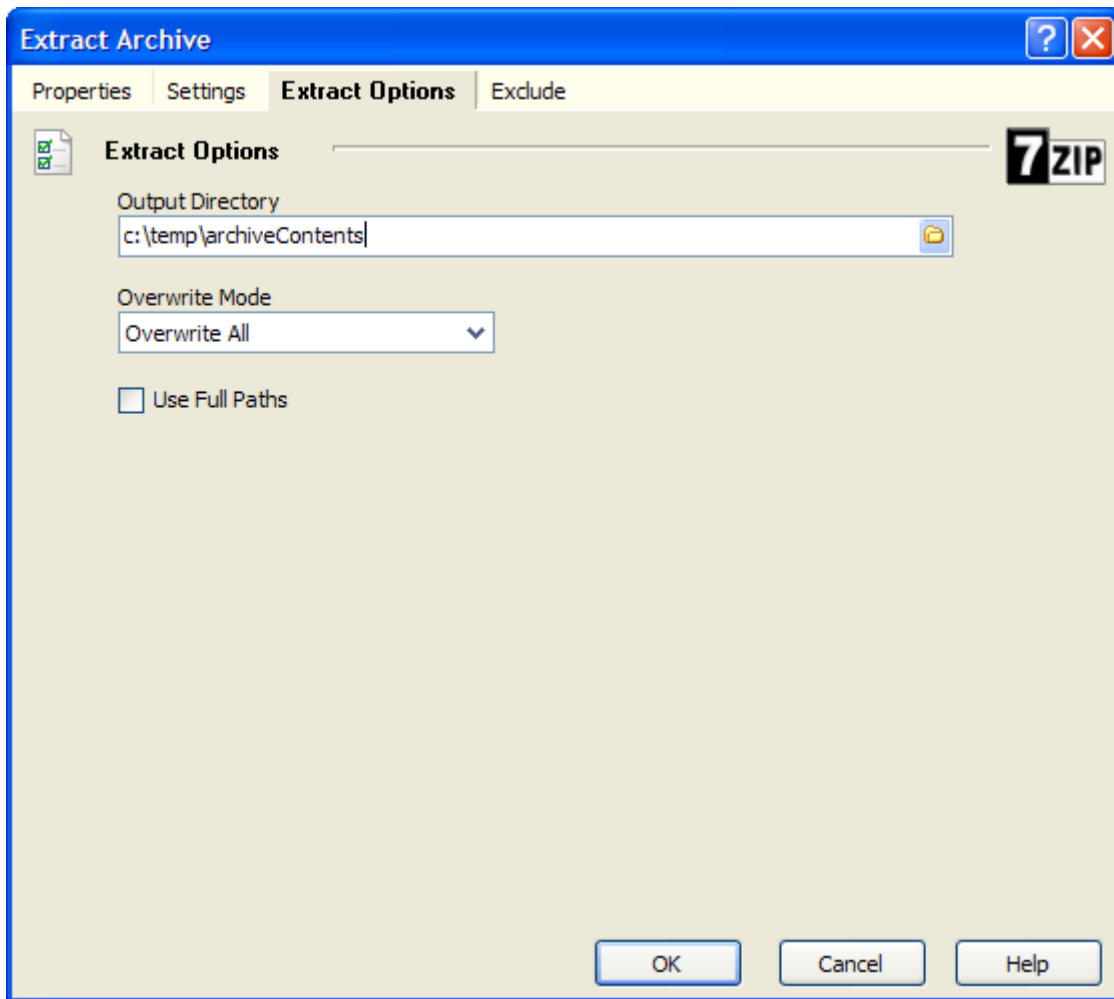
For more detailed descriptions of the options, see the Create Archive action.

#### 5.13.4.3 Extract Archive

The Extract Archive actions allows you to extract the files from an archive in any of the following formats:

Zip, 7z, GZip, BZip2, TAR, RAR, ARJ, CAB, CPIO, RPM, DEB, SPLIT

More Info on the 7Zip based actions



**Output Directory** - specify the directory where the extracted files will be saved to

**Overwrite Mode** - specify how existing files with the same name will be dealt with

**Use Full Paths** - Files will be extracted with their paths as they are in the archive file. eg. if a file in the archive is in a directory called "dir" then the files in that directory will be extracted to <outputdirectory>\dir\<files>

For more detailed descriptions of the other options, see the Create Archive action.

#### 5.13.4.4 Update Archive

The Update Archive actions allows you to update (or freshen) the files within an existing archive in any of the following formats:  
Zip, 7z, GZip, BZip2, TAR

More Info on the 7Zip based actions

For more detailed descriptions of the options, see the Create Archive action.

#### **5.13.4.5 List Archive**

The List Archive actions allows you to list the files stored within an archive in any of the following formats:

Zip, 7z, GZip, BZip2, TAR

More Info on the 7Zip based actions

For more detailed descriptions of the options, see the Create Archive action.

#### **5.13.4.6 Delete from Archive**

The Delete from Archive actions allows you to delete files within an archive in any of the following formats:

Zip, 7z, GZip, BZip2, TAR

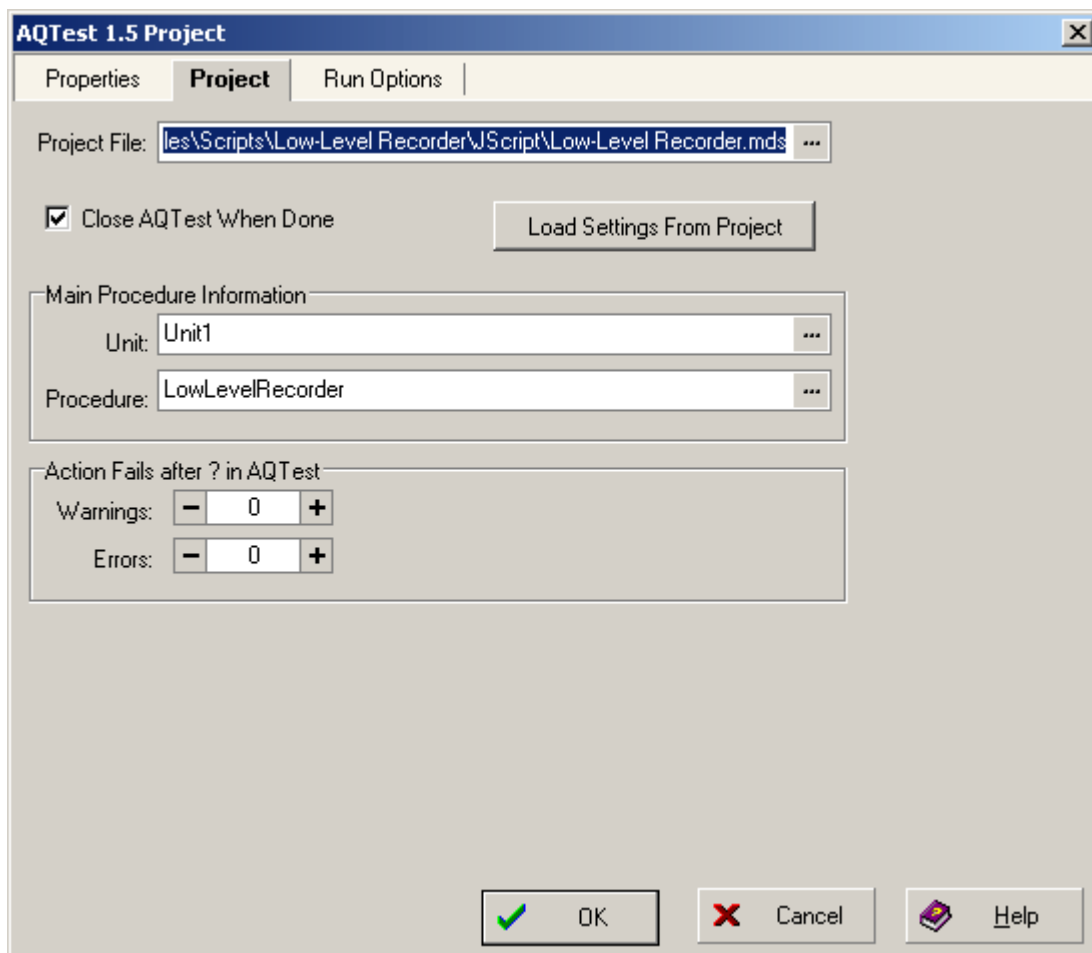
More Info on the 7Zip based actions

For more detailed descriptions of the options, see the Create Archive action.

### **5.14 Testing Tools**

#### **5.14.1 AQTest**

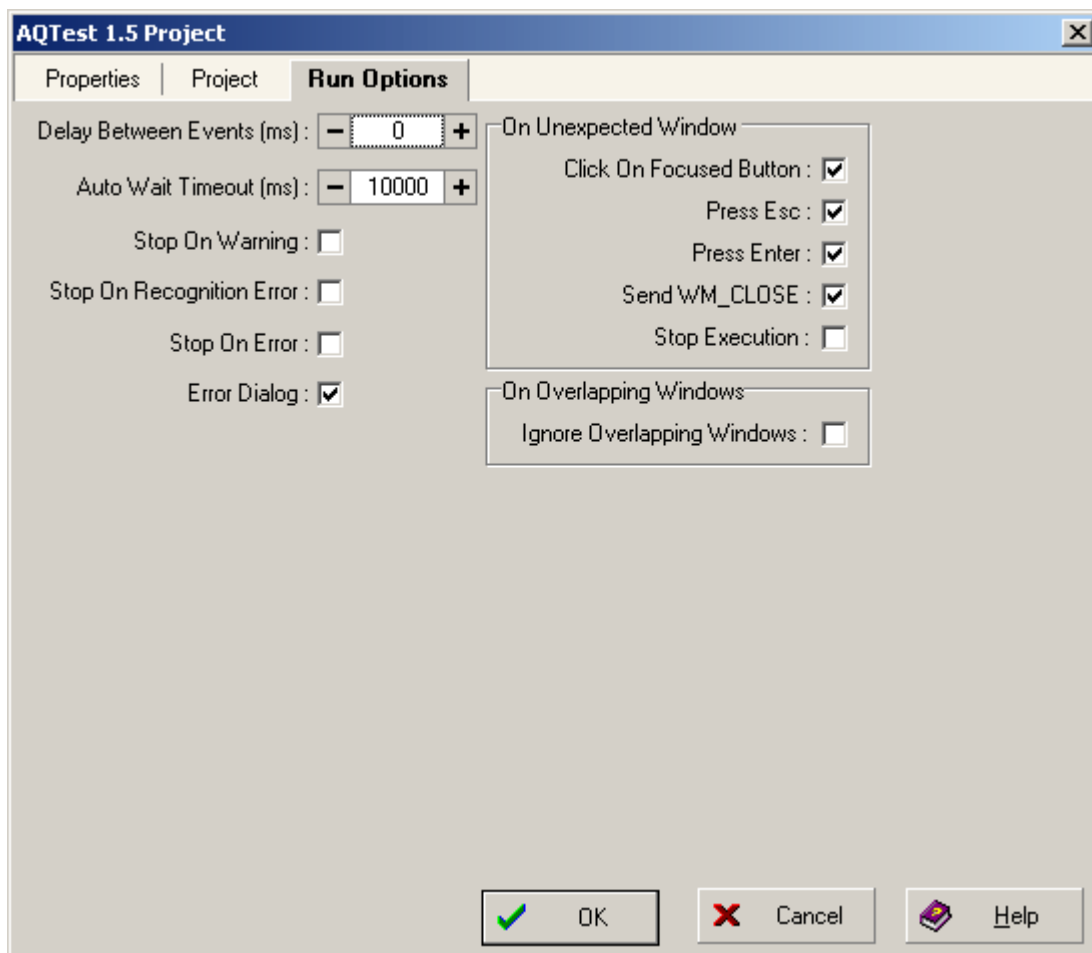
This action integrates Automated QA's AQTest automated testing tool with FinalBuilder. This makes it possible to perform Regression testing as part of the build process.



**Project File :** The fully qualified path to the AQTest project file (.mds).

**Close AQTest When Done :** This action will start AQTest if it is not already loaded. If you are only doing one test, you can check this option to have FinalBuilder close AQTest when it is done testing.





For a full description of the options, please consult the AQTest 1.5 manual.

### Scripting Info

The Action properties available are :

**property** ProjectFile : WideString  
**property** CloseWhenDone : WordBool;

This action was written by Eric Holton and VSoft Technologies. AQTest can be found at <http://www.automatedqa.com>

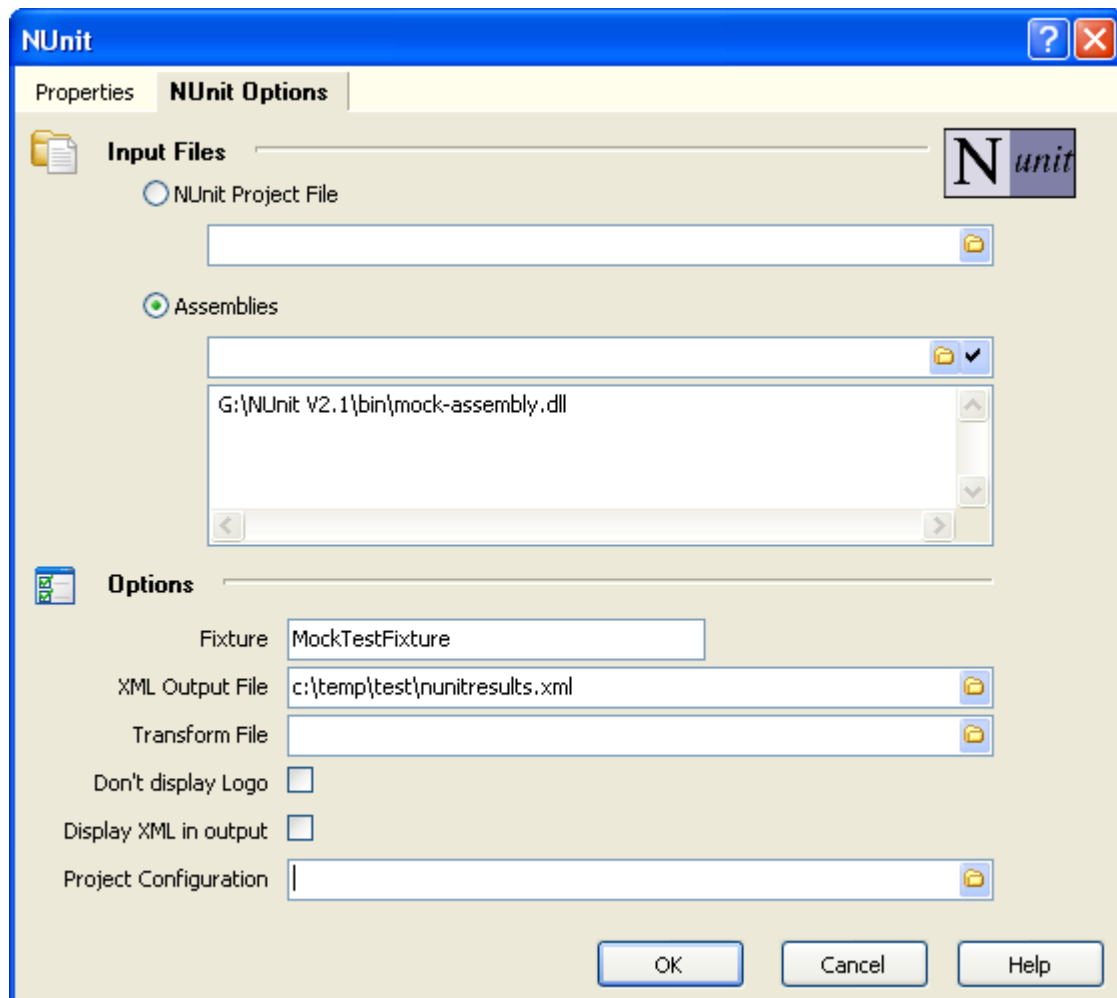
### 5.14.2 AutomatedQA Test Complete Actions

These Actions provide integration with AutomatedQA's TestComplete 2.x & 3.x products. They were kindly provided by Eric Holton - <http://www.holtonsystems.com>

### 5.14.3 NUnit Action

NUnit is a unit-testing framework for all .Net languages. Initially ported from JUnit, the current version, 2.1 is the third major release of this xUnit based unit testing tool for Microsoft .NET. It is written entirely in C# and has been completely redesigned to take advantage of many .NET language features, for example custom attributes and other

reflection related capabilities. NUnit brings xUnit to all .NET languages.



For more information see the NUnit homepage:

<http://www.nunit.org/>

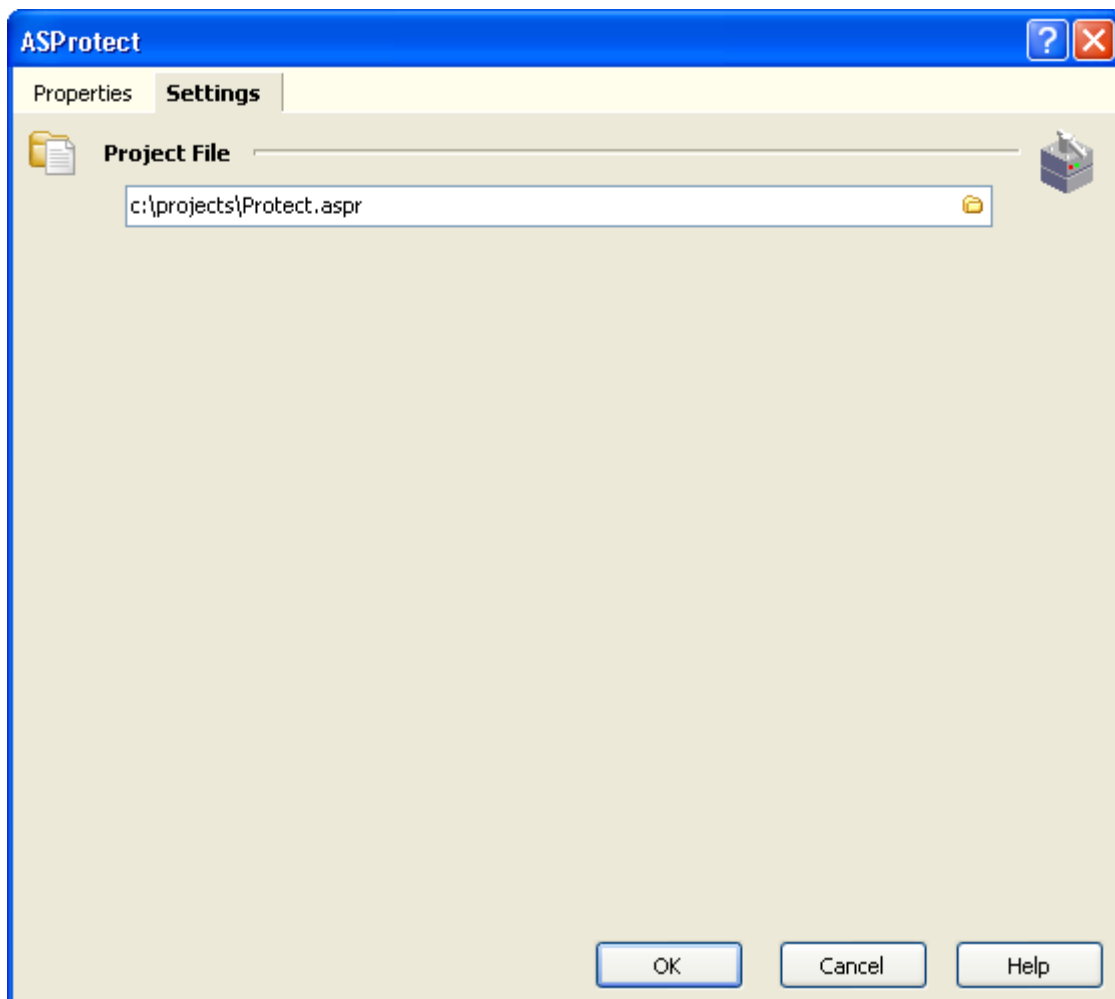
#### 5.14.4 MSTest

The MS Test action enables you to automate the testing of your .Net assemblies using MSTest.

### 5.15 Licensing Tools

#### 5.15.1 ASProtect Action

This Action provides support for the ASProtect Software protection product. For more information on ASProtect, please visit the ASPack website <http://www.aspack.com>



### Scripting Info

The Action properties available are :

**property** ProjectFile : String;

The Options Object is exposed as :

**function** ASPProtectOptions : IFBASProtectOptions

It has one property:

**property** ASPProtectLocation: string

### 5.15.2 ProActivate Action

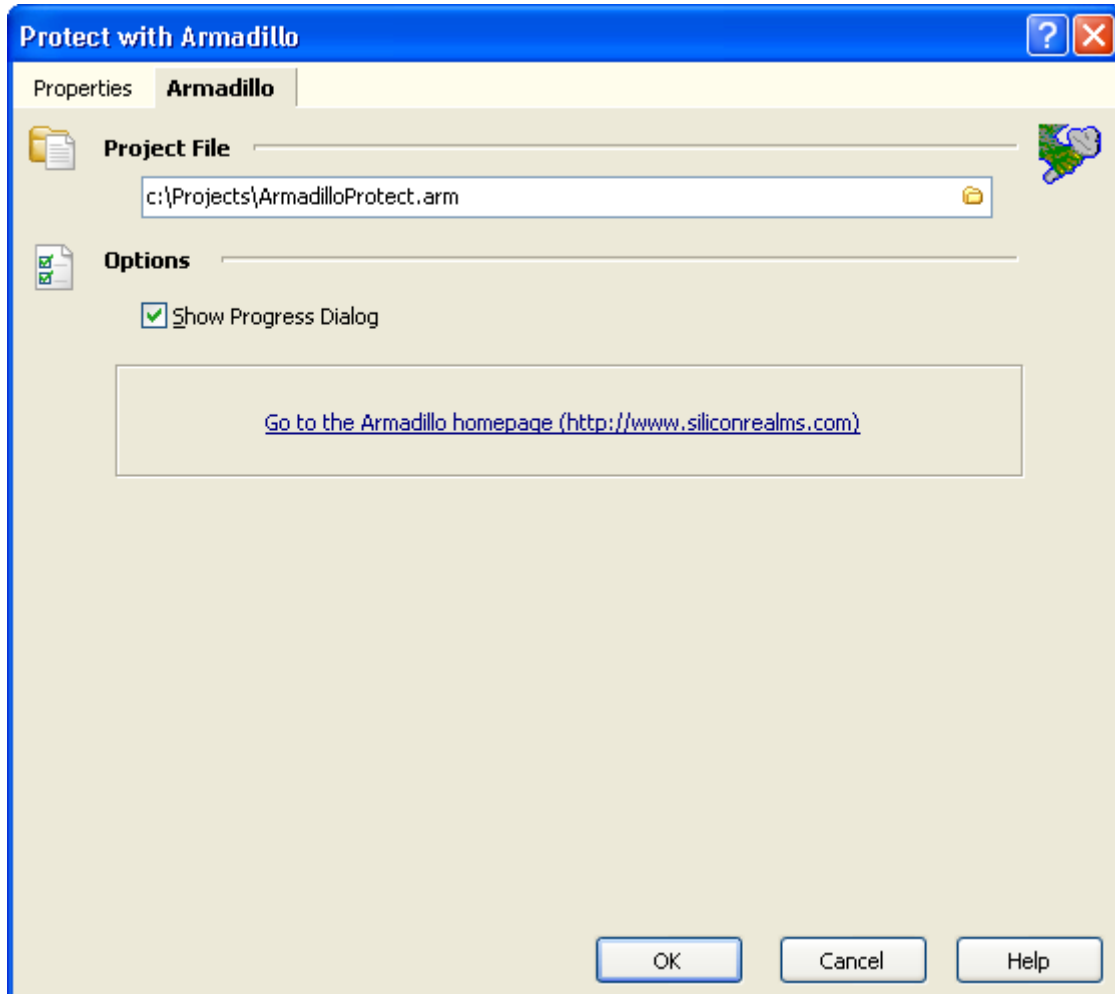
This Action provides support for TurboPower's ProActivate software Licensing product. For more information on ProActivate please visit the TurboPower website

<http://www.turbopower.com>

### 5.15.3 Armadillo Action

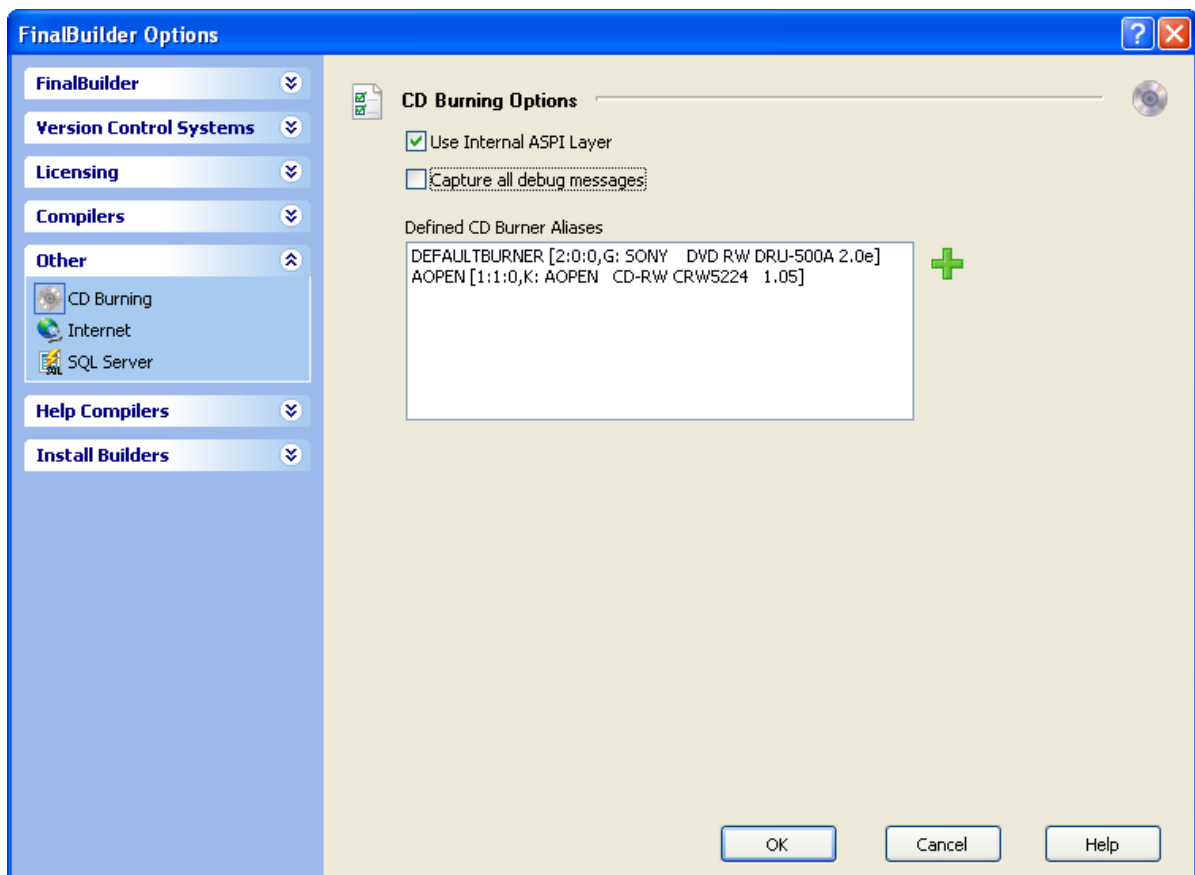
This action provides support for the Armadillo Software protection system. For more info on Armadillo, please visit the Silicon Realms website - <http://www.siliconrealms.com>. Note that to use this action you must set the Armadillo location in the options dialog.

This action was written and kindly provided by Peter Thörnqvist.



### 5.16 CD/DVD Burner Actions

In order to use the CD/DVD Burner actions, you need to define Aliases for the burner(s) available on your machine. Aliases are what you use to determine which burner an action will use. This allows you to define the same alias for different burners on other machines, enabling the project to run even though other machines have different burners.

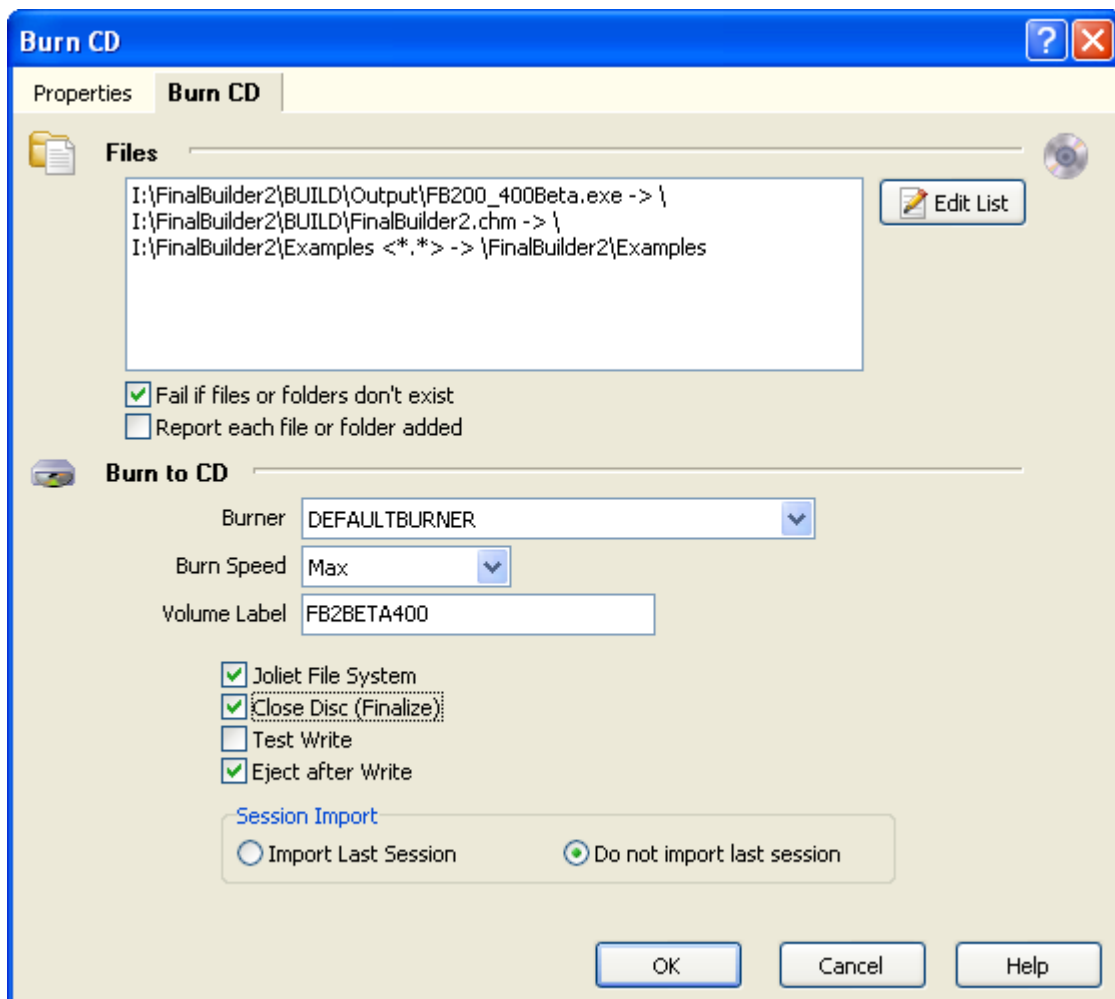


By default, FinalBuilder uses its own internal ASPI layer (CD/DVD Burner API), which supports most types of drives including IDE, SCSI, USB & Firewire. If you find that FinalBuilder does not find your drive, or it has problems with it, try turning off this option and it will use an external aspi layer in WNASPI32.DLL. Note that the Adaptec layer commonly found on many machines does not support Firewire or USB devices, we recommend using the WNASPI32.DLL from Nero into your system32 directory if you have it and want to try an external aspi layer.

### 5.16.1 Burn CD/DVD Action

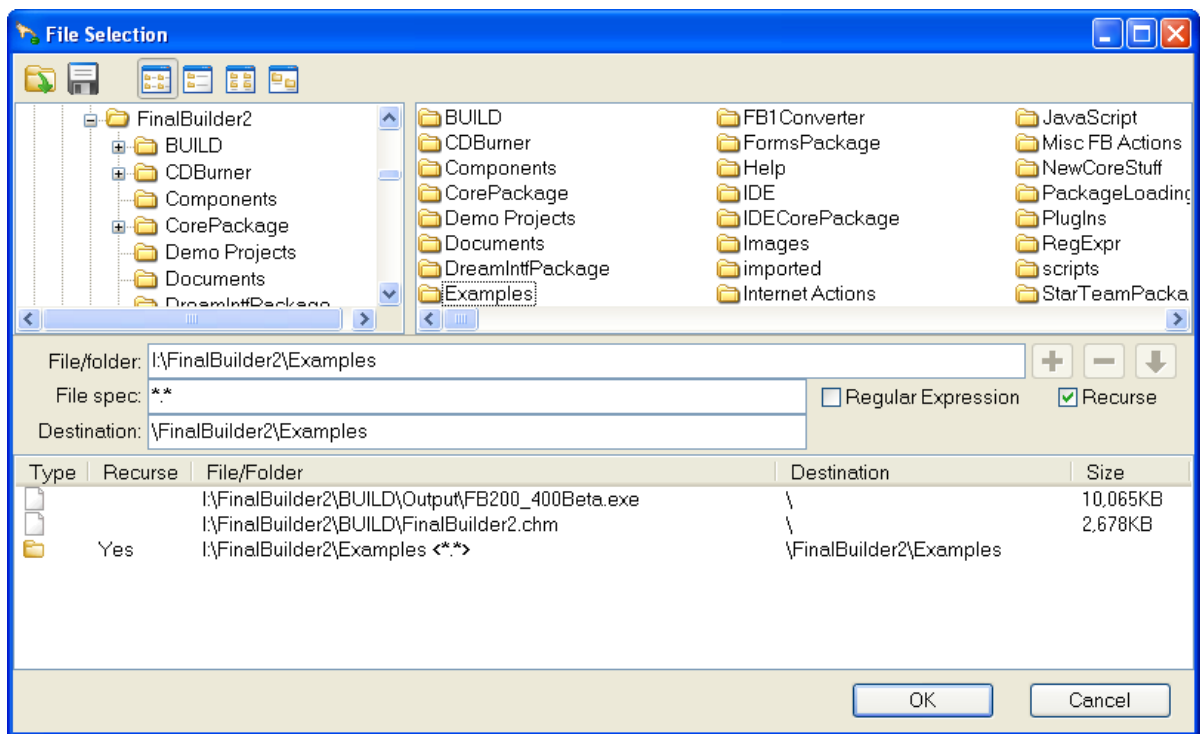
#### [Professional Edition]

This action supports burning files direct to CDR/CDRW/DVDR & DVDRW Media (depending on your burner hardware). The files burned directly to the drive, ie an image is not created first. Note that to use this action you need to define an alias for your burner first (see here for more info).



If you are having problems seeing the files on your DVD or CD after burning, then you might have to enable the "Eject after Write" option so that the operating system refreshes the contents of the media.

The File Selection editor allows you to select files & folders and use wildcards or regular expressions to add files to the selection.



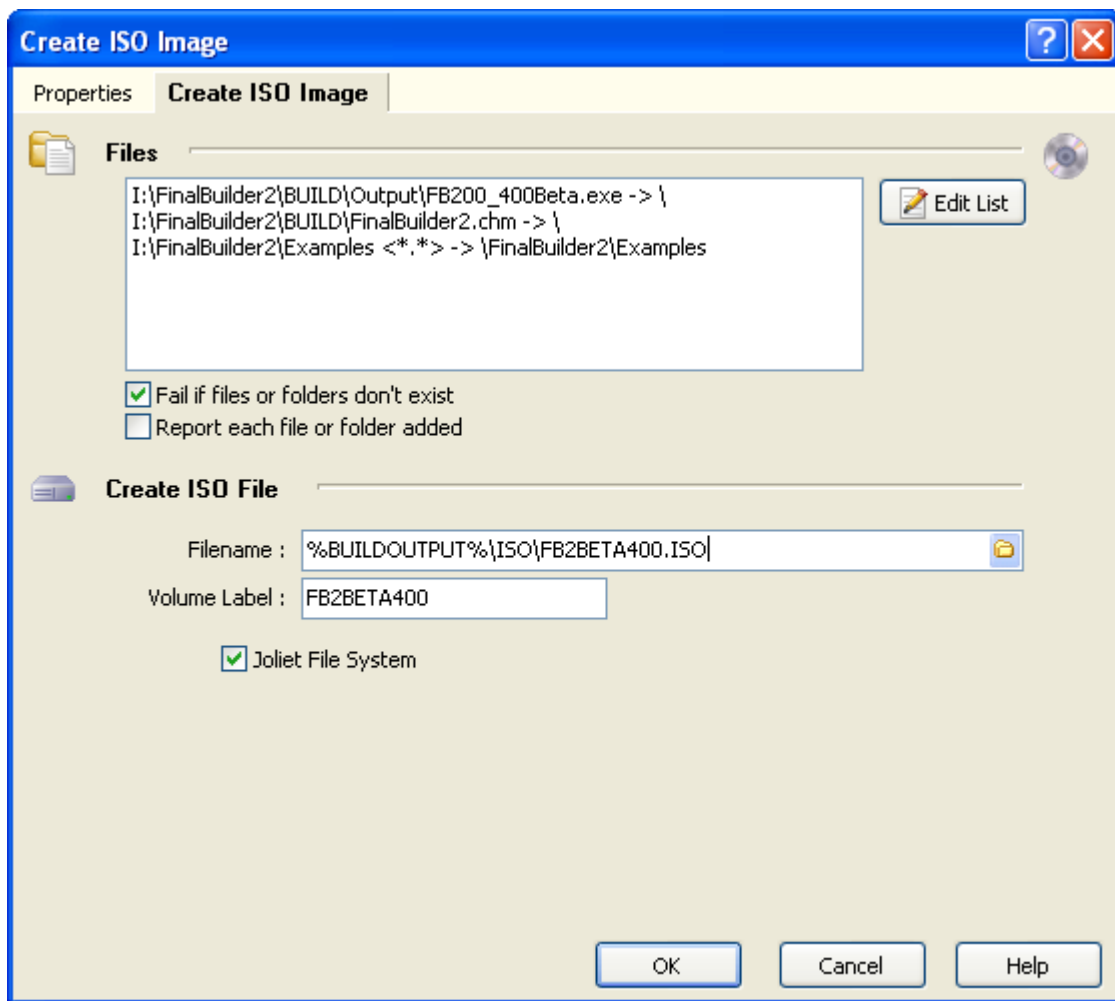
In the File spec you can enter a normal DOS type file spec using wild cards \* and ?

You can also specify a regular expression, but you need to select the "Regular Expression" checkbox. If you are specifying \*.\* you should not select regular expression as it is an invalid regular expression.

### 5.16.2 Create ISO Action

**[Professional Edition]**

This action allows you to create an ISO CD/DVD Image File. The File Selection is exactly the same as in the Burn CD Action. Note that FinalBuilder does not impose any limits on the size of the resulting ISO File.

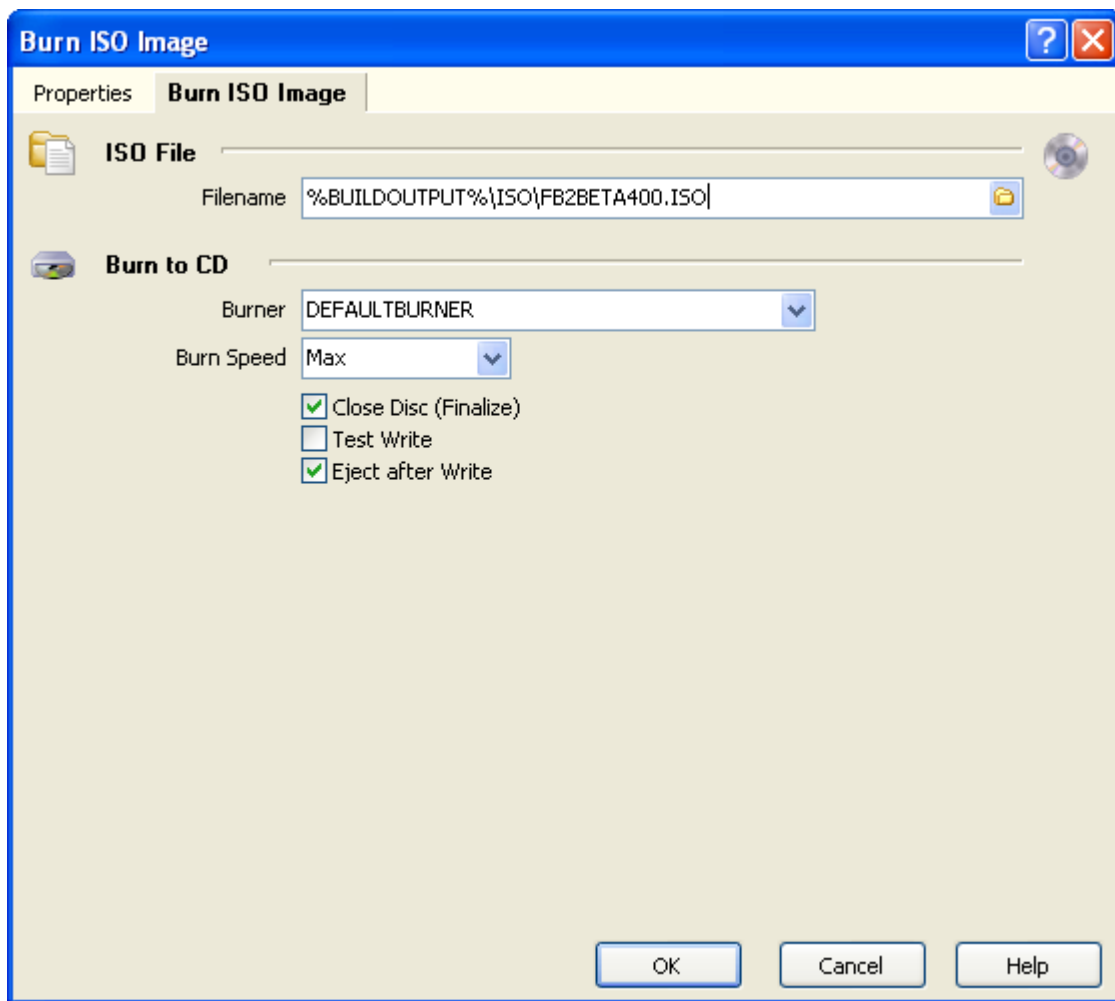


### 5.16.3 Burn ISO Action

#### [Professional Edition]

This action supports burning an ISO file generated by FinalBuilder (or any other tool that generates valid ISO Files) to CDR/CDRW/DVDR & DVDRW Media (depending on your burner hardware). Note that to use this action you need to define an alias for your burner first (see here for more info).

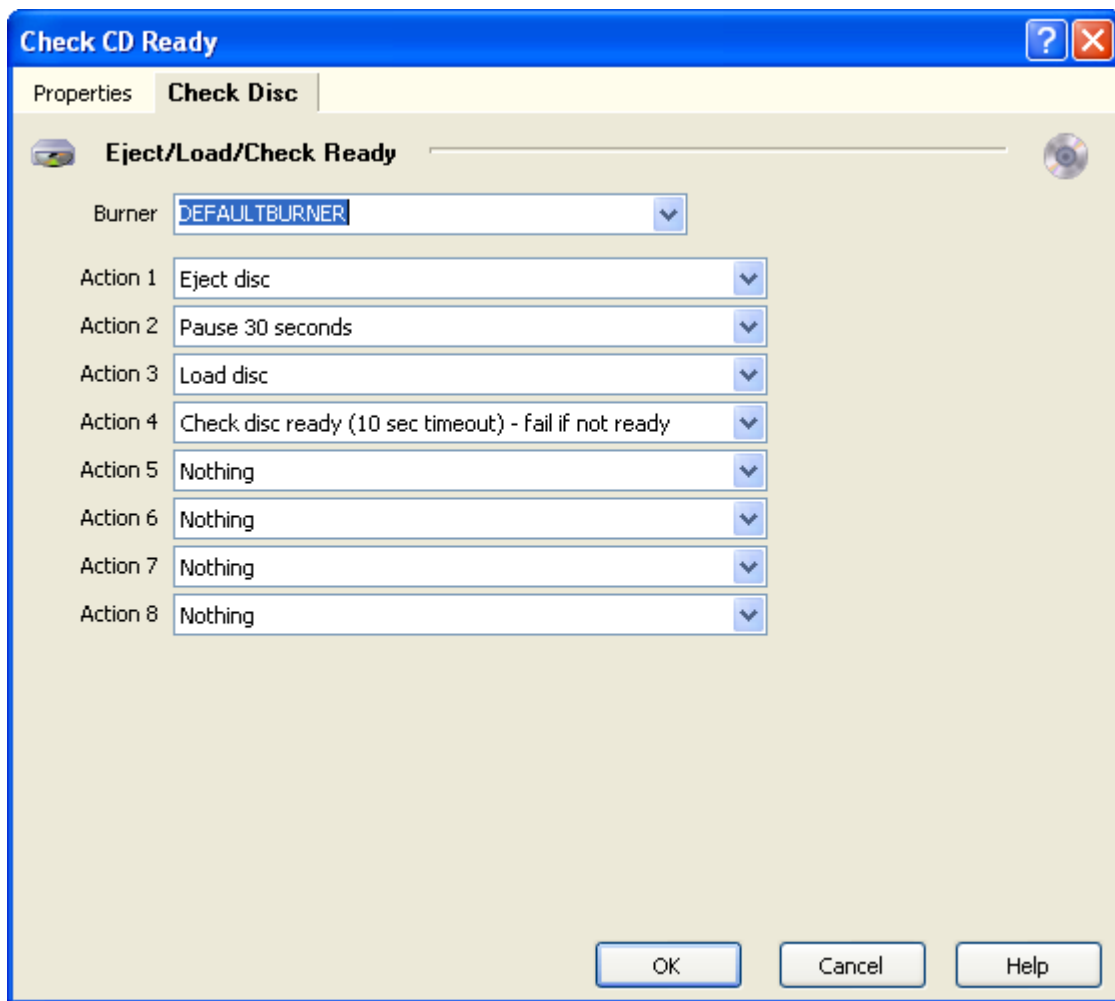




#### 5.16.4 Check Ready Action

##### [Professional Edition]

This action allows you to automate to some degree the process of loading a CD into the burner, and making sure the burner is ready to burn before attempting to burn to CD. An example which shows the use of this action is installed in the FinalBuilder\Examples\CDBurner directory. Note that to use this action you need to define an alias for your burner first (see here for more info).

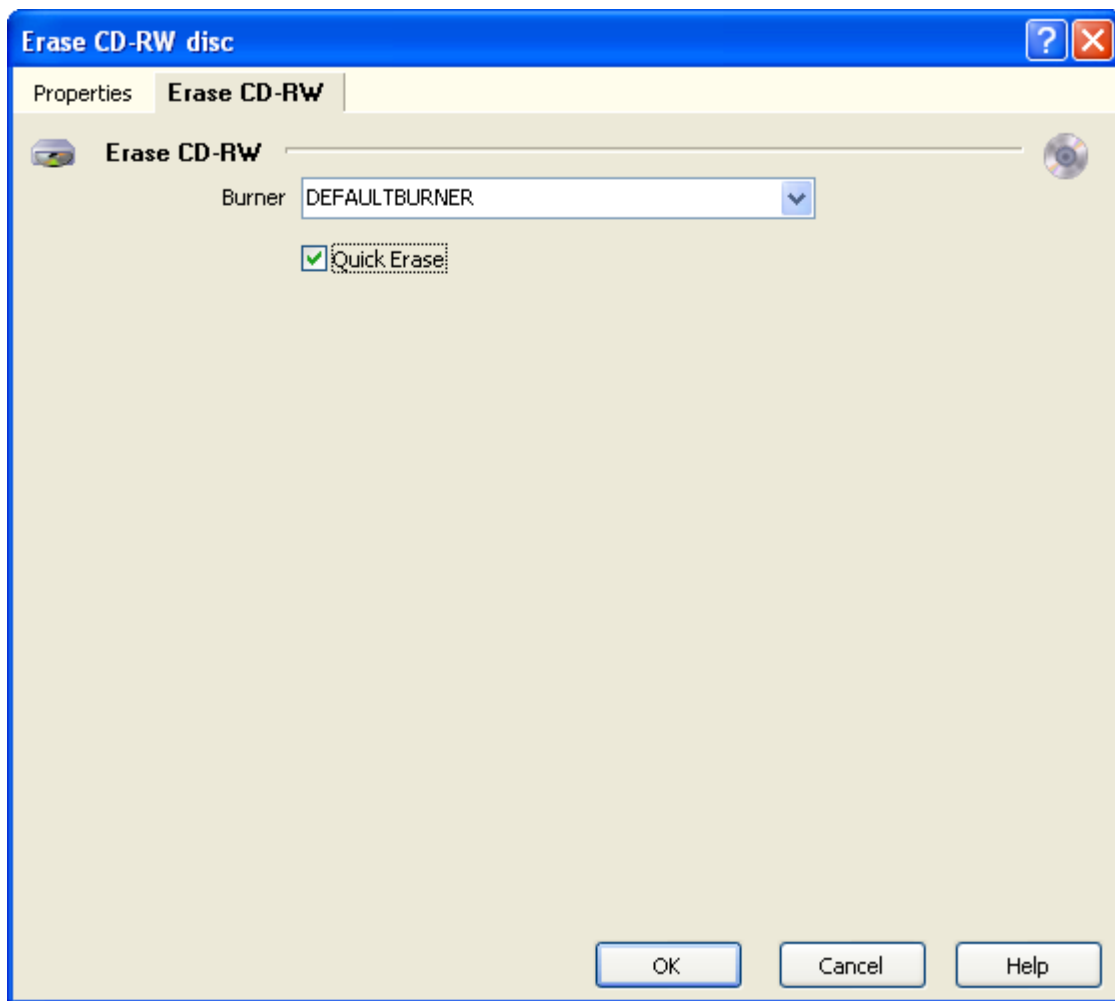


### 5.16.5 Erase CD/DVD RW

[Professional Edition]

This action allows you to Erase rewritable CD/DVD media (if your burner supports this). Note that to use this action you need to define an alias for your burner first (see here for more info). If your burner does not support rewritable media then it will not appear in the list of burner aliases.

Quick Erase is recommended for most media and is substantially faster (full erase can take a long time on a DVDRW!).



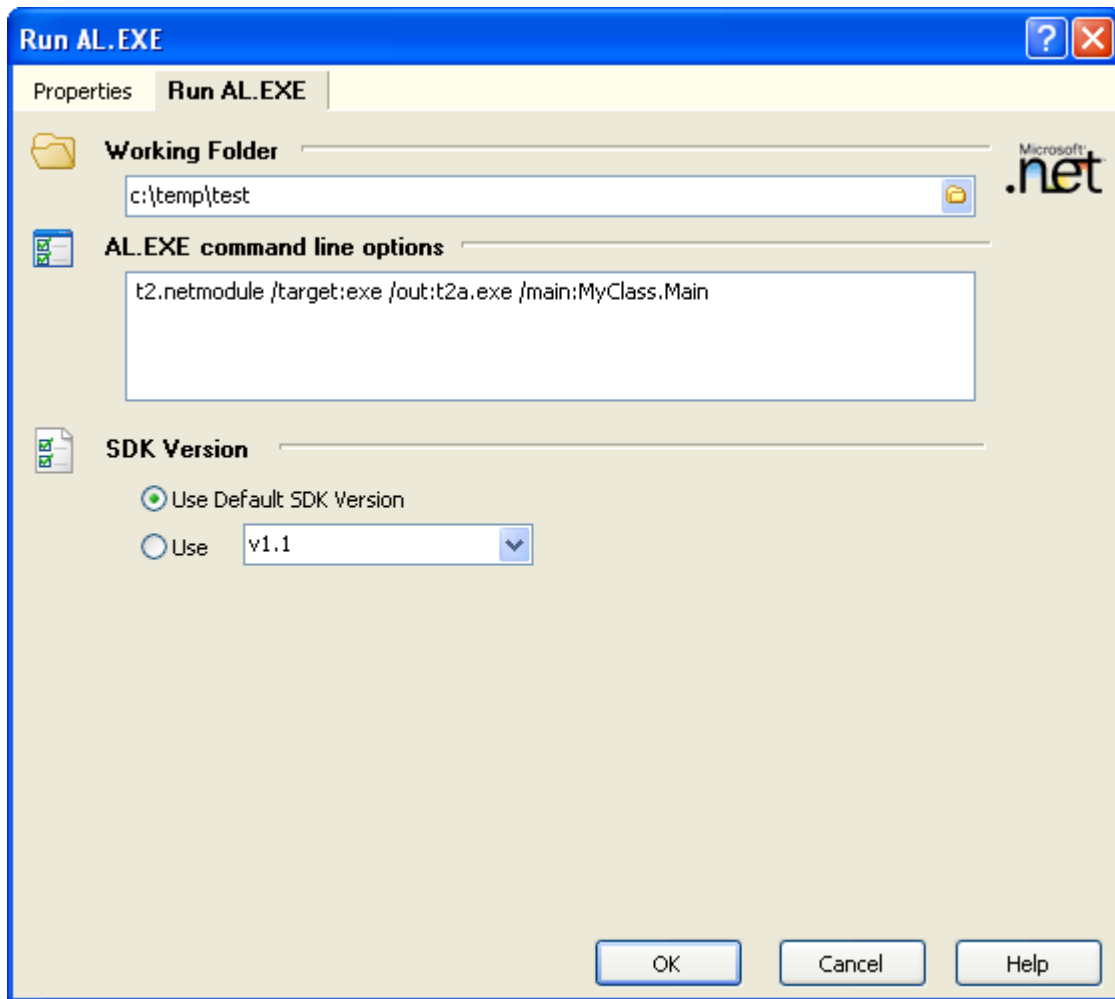
## 5.17 .NET Actions

### 5.17.1 .Net Framework Tools

The .Net Framework tools are installed as part of the .Net framework (ie. if you have .Net installed, you have the framework tools)

#### 5.17.1.1 Run AL.EXE

The Assembly Linker generates a file with an assembly manifest from one or more files that are either modules or resource files. A module is a Microsoft intermediate language (MSIL) file that does not have an assembly manifest.

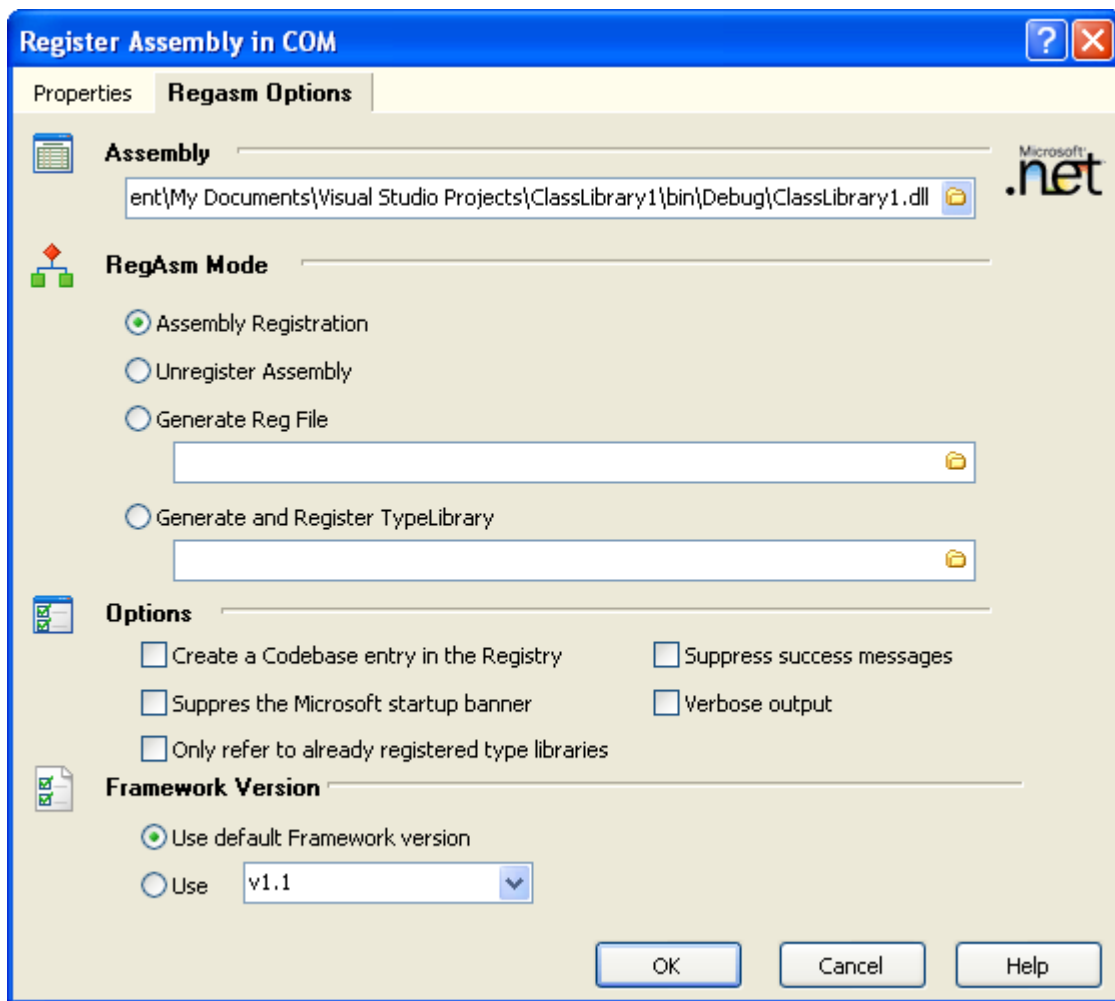


For more information see:

<http://msdn.microsoft.com/library/en-us/cptools/html/cpgrfAssemblyGenerationUtilityAlexe.asp>

#### 5.17.1.2 Register Assembly in COM [REGASM]

The Assembly Registration tool reads the metadata within an assembly and adds the necessary entries to the registry, which allows COM clients to create .NET Framework classes transparently. Once a class is registered, any COM client can use it as though the class were a COM class. The class is registered only once, when the assembly is installed. Instances of classes within the assembly cannot be created from COM until they are actually registered.

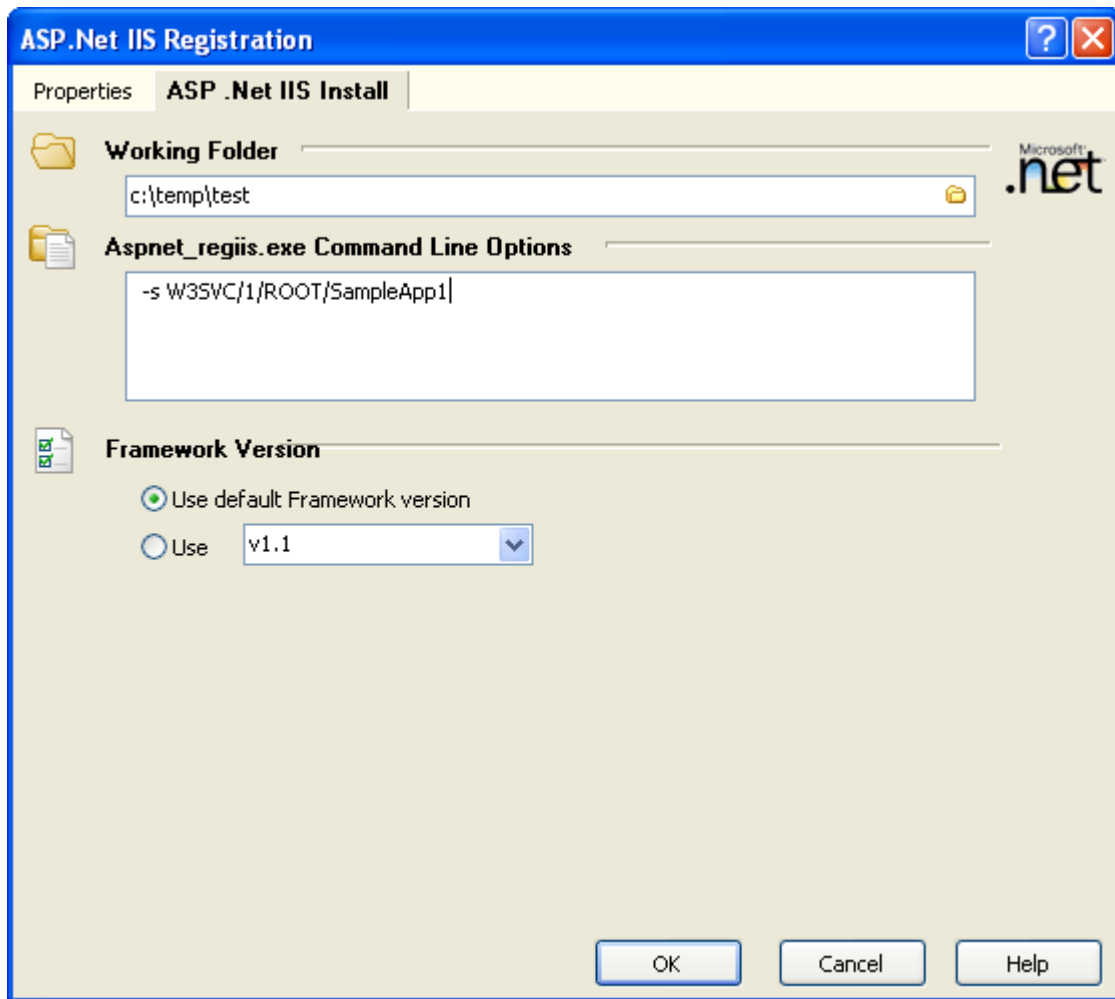


For more information see:

<http://msdn.microsoft.com/library/en-us/cptools/html/cpgrfAssemblyRegistrationToolRegasmexe.asp>

### 5.17.1.3 Run ASPNET\_REGIIS.EXE

Allows an administrator or installation program to update the scriptmaps for an ASP.NET application to point to the ASP.NET ISAPI version associated with the tool. You can also use the tool to perform other ASP.NET configuration operations.



For more information see:

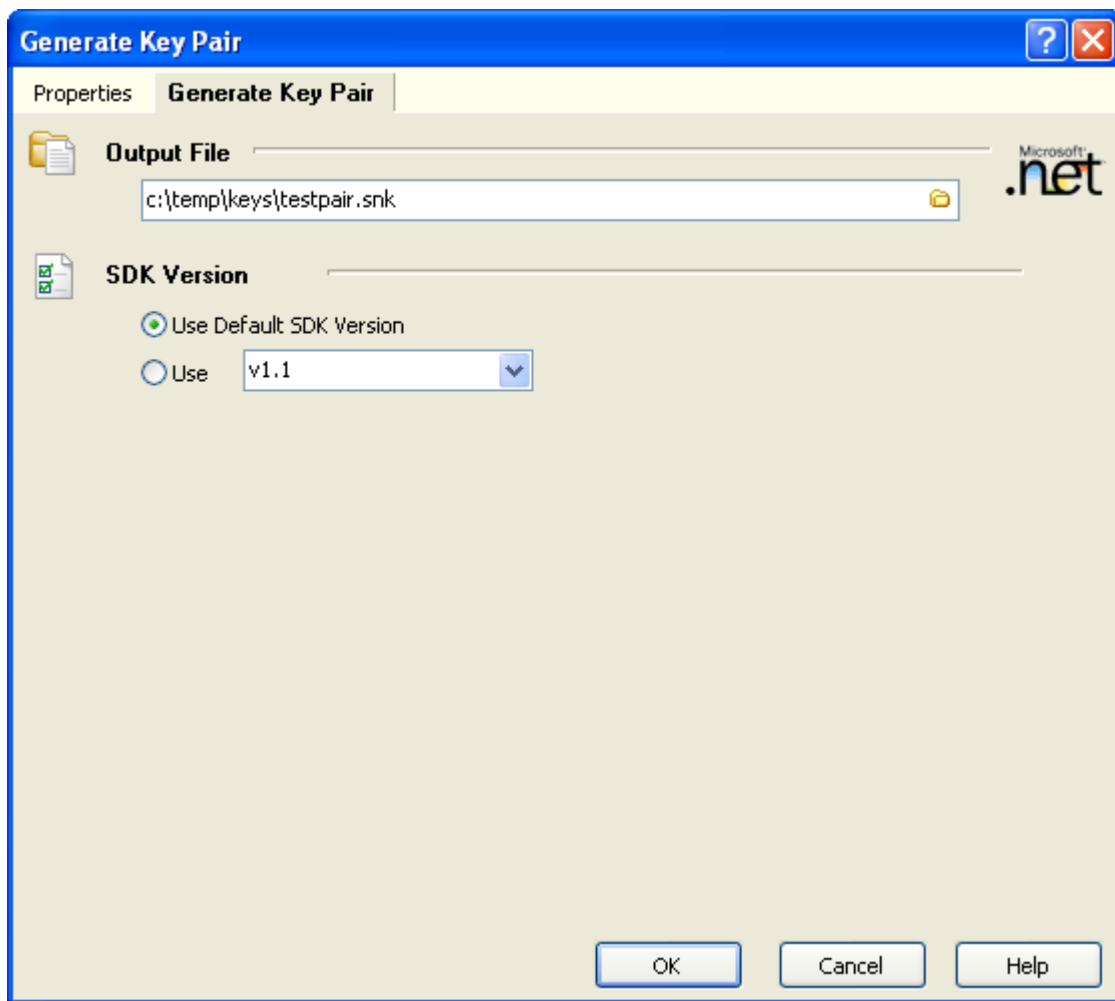
[http://msdn.microsoft.com/library/en-us/cptools/html/cpgrfASPNETIISRegistrationToolAspnet\\_regiisexe.asp](http://msdn.microsoft.com/library/en-us/cptools/html/cpgrfASPNETIISRegistrationToolAspnet_regiisexe.asp)

## 5.17.2 .Net SDK Tools

The .Net SDK tools are installed as part of the .Net SDK, please check if you have the SDK installed, the standard place is C:\Program Files\Microsoft.NET\SDK

### 5.17.2.1 Generate Key Pair [SN]

Generates a new key pair and writes it to the specified file.

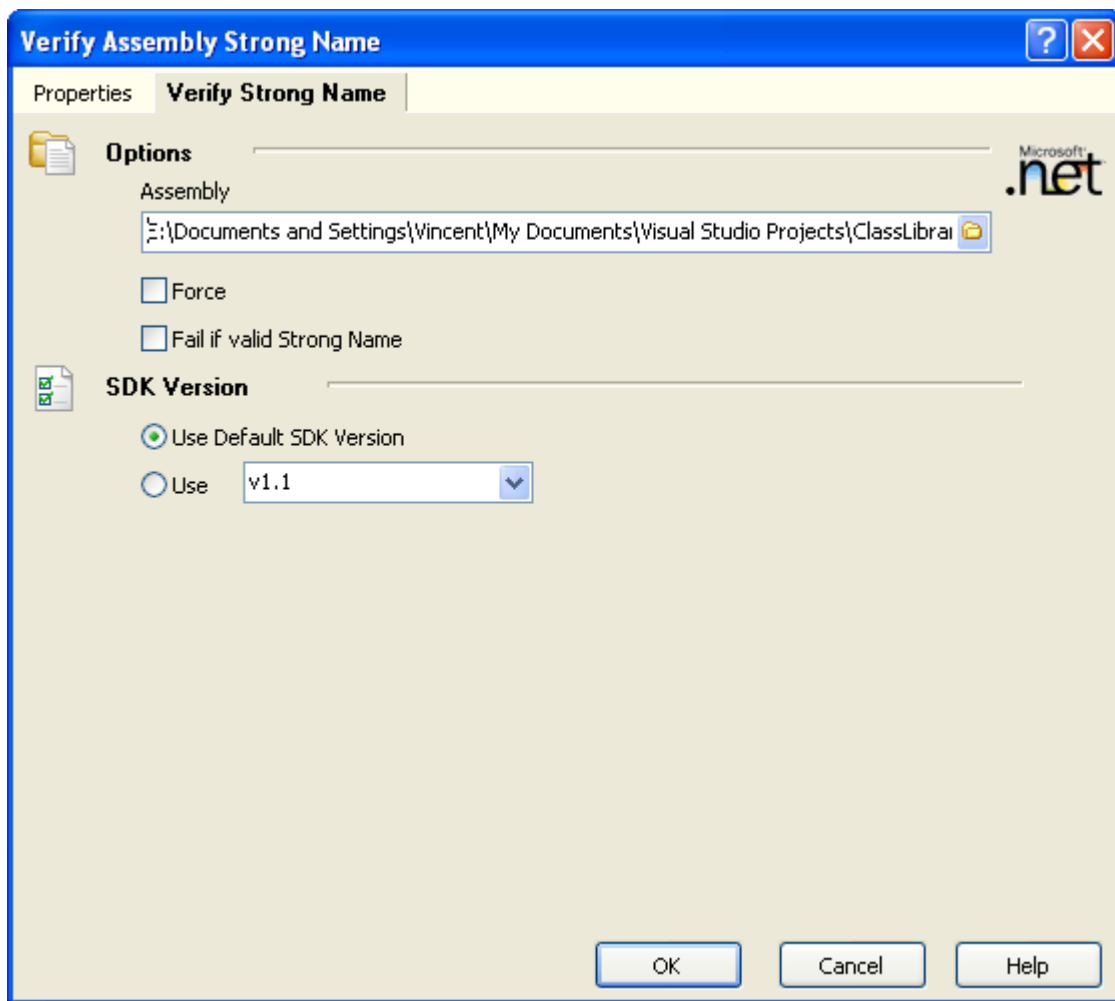


For more information see:

<http://msdn.microsoft.com/library/en-us/cptools/html/cpgrfStrongNameUtilitySNexe.asp>

#### 5.17.2.2 Verify Strong Name [SN]

Verifies the strong name in assembly, where assembly is the name of a file that contains an assembly manifest.



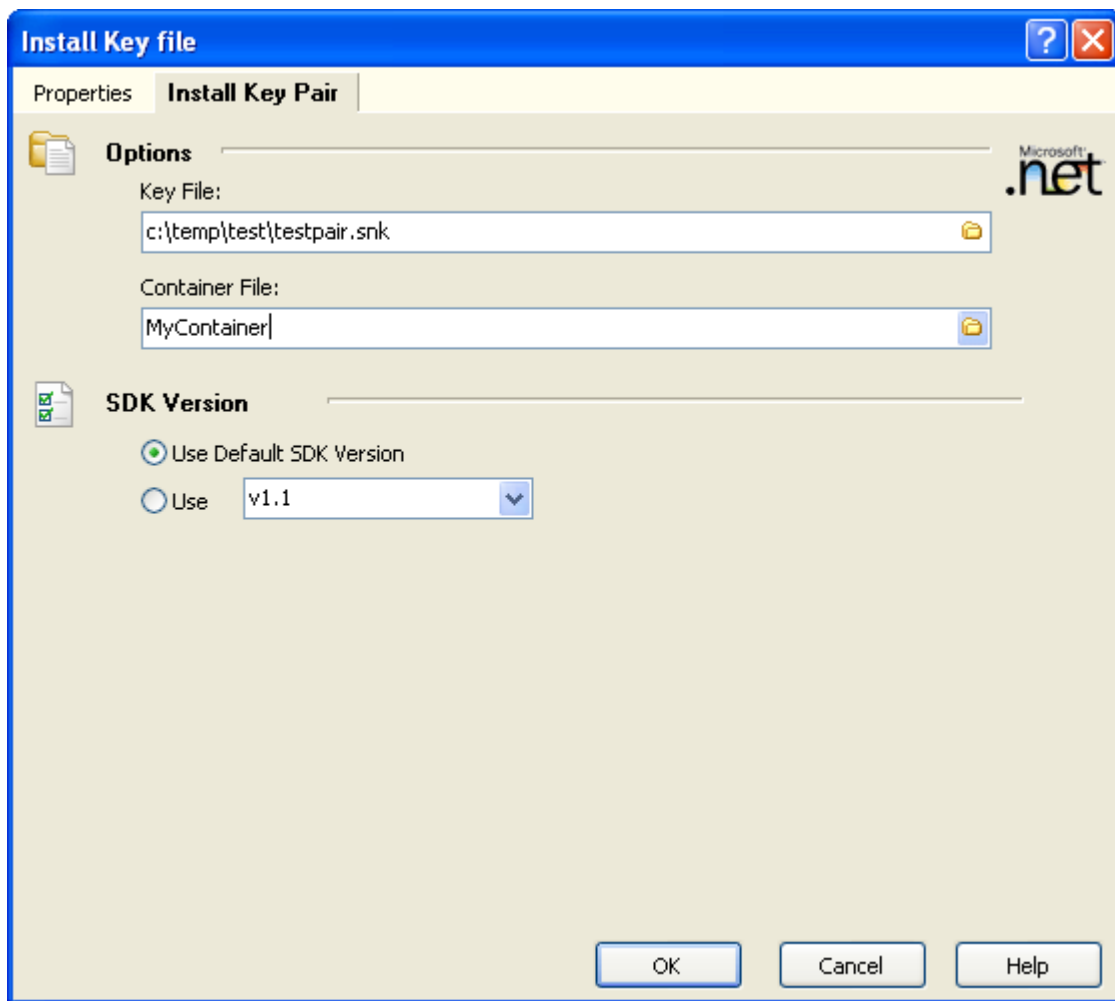
For more information see:

<http://msdn.microsoft.com/library/en-us/cptools/html/cpgfrStrongNameUtilitySNexe.asp>

#### 5.17.2.3 Install Key in Container [SN]

Installs the key pair from infile in the specified key container. The key container resides in the strong name CSP.



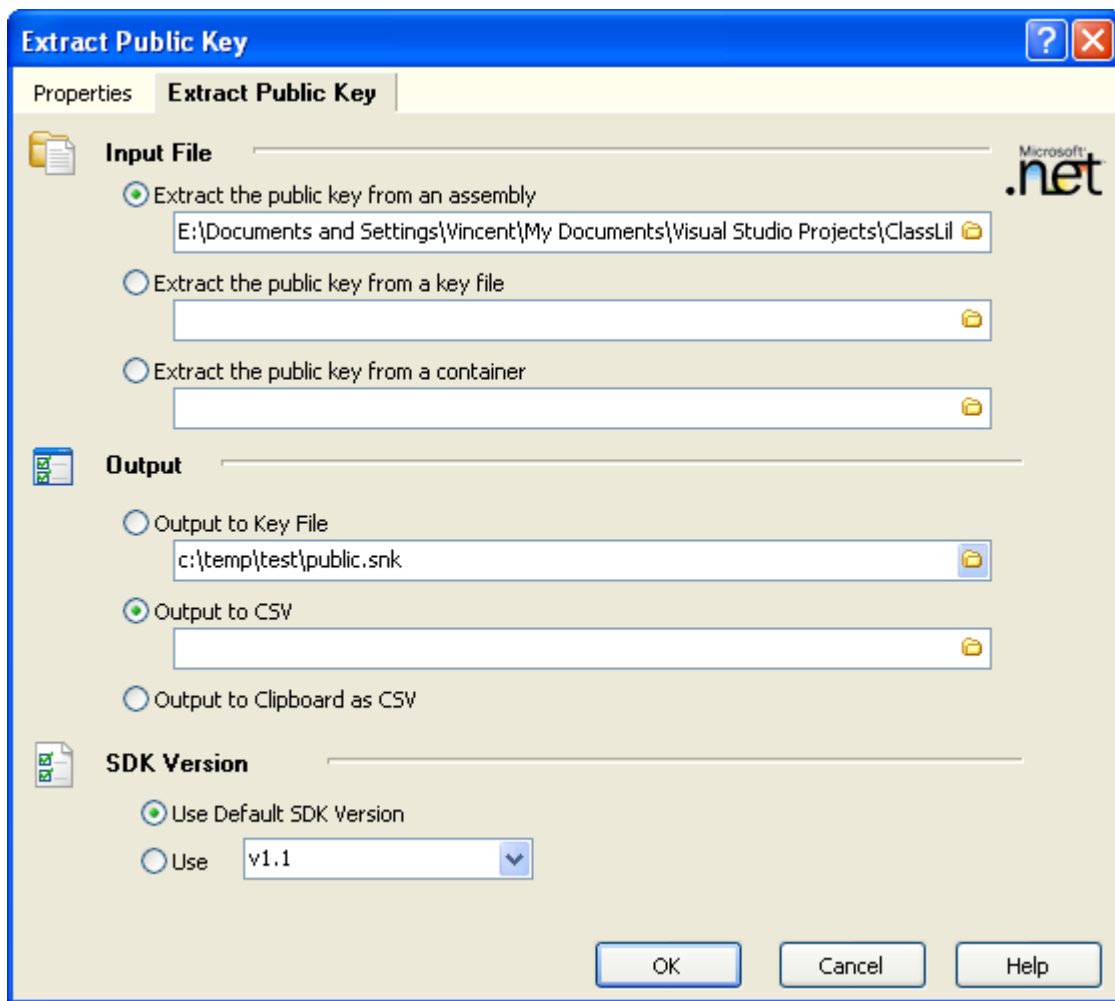


For more information see:

<http://msdn.microsoft.com/library/en-us/cptools/html/cpgrfStrongNameUtilitySNexe.asp>

#### 5.17.2.4 Extract Public Key [SN]

Extracts the public key from assembly and stores it in outfile.

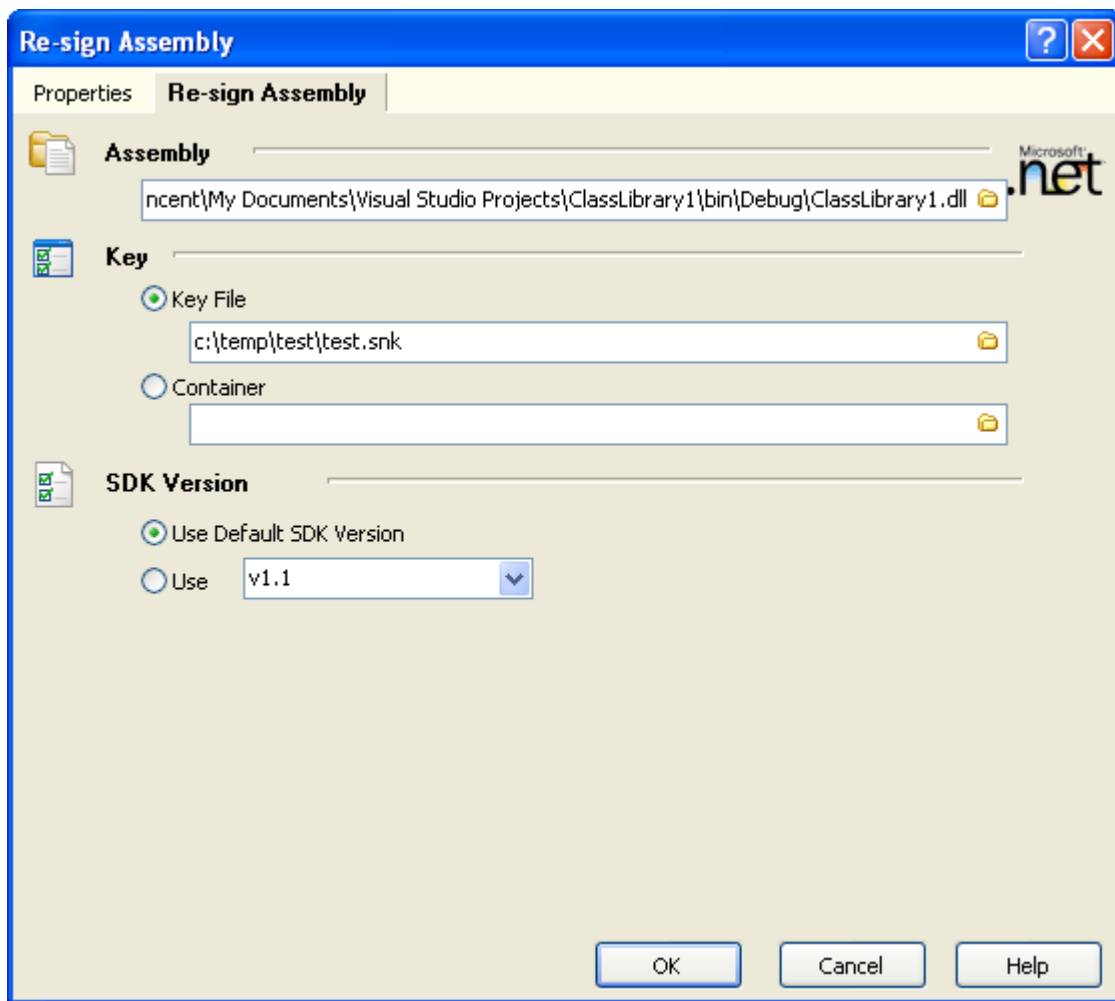


For more information see:

<http://msdn.microsoft.com/library/en-us/cptools/html/cpgrfStrongNameUtilitySNexe.asp>

#### 5.17.2.5 Re-sign Assembly [SN]

Re-signs a previously signed or delay signed assembly.

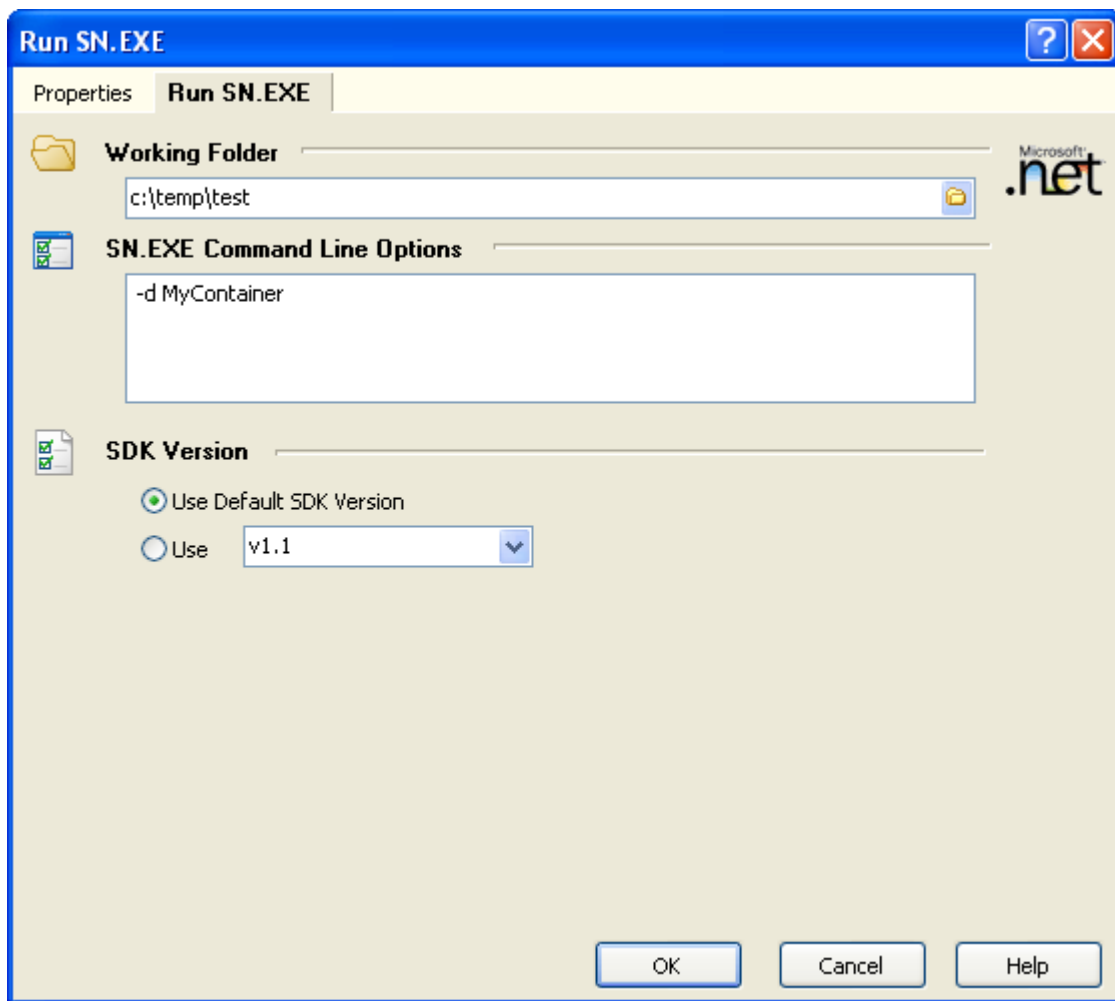


For more information see:

<http://msdn.microsoft.com/library/en-us/cptools/html/cpgrfStrongNameUtilitySNexe.asp>

#### 5.17.2.6 Run SN.EXE

The Strong Name tool helps sign assemblies with strong names. Sn.exe provides options for key management, signature generation, and signature verification. You should only need to use this action if the Generate Key Pair, Verify Strong Name, Install Key in Container, Extract Public Key, and Re-sign Assembly can't do what you require.



For more information see:

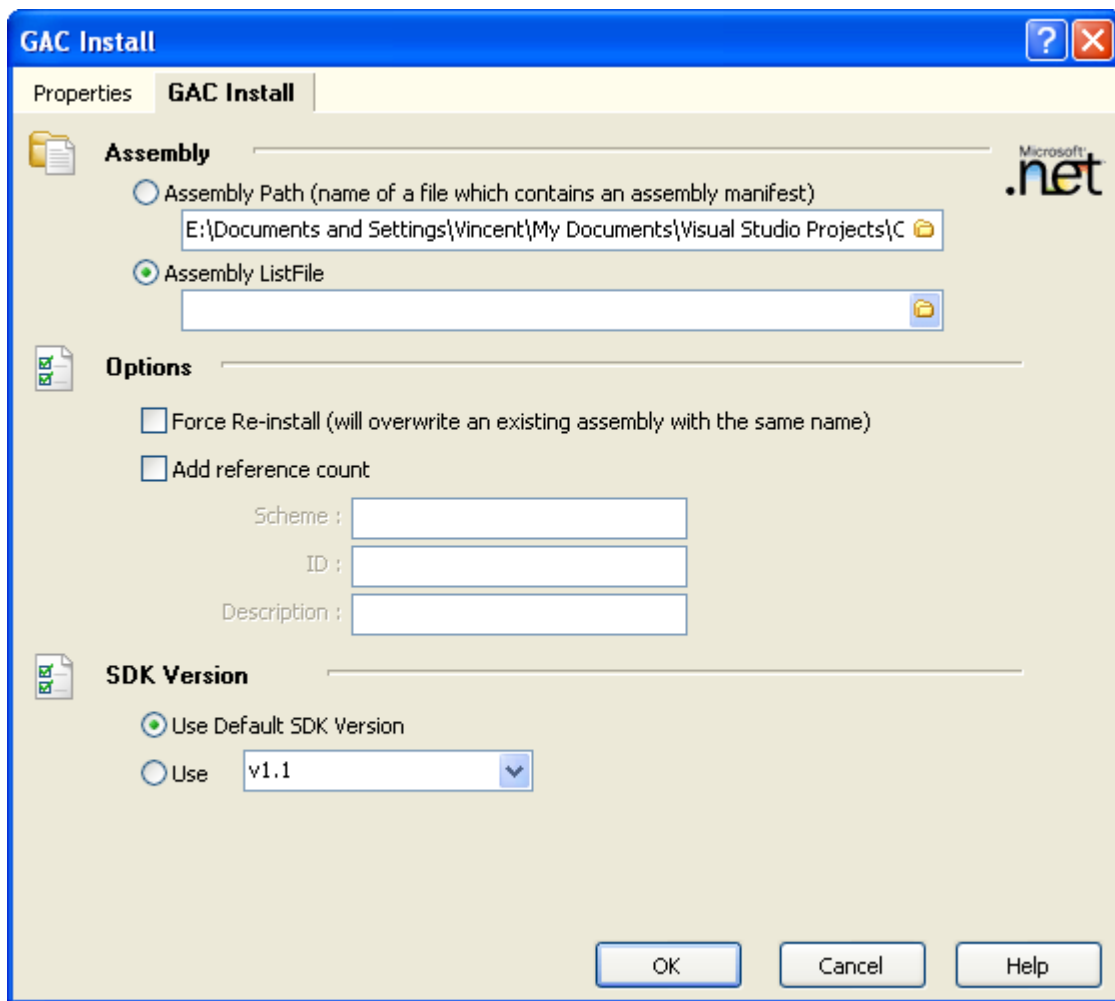
<http://msdn.microsoft.com/library/en-us/cptools/html/cpgrfStrongNameUtilitySNexe.asp>

#### 5.17.2.7 GAC Install [GACUTIL]

Installs an assembly into the global assembly cache.

This action automates the following options:

/i  
/if  
/il  
/ir



For more information see:

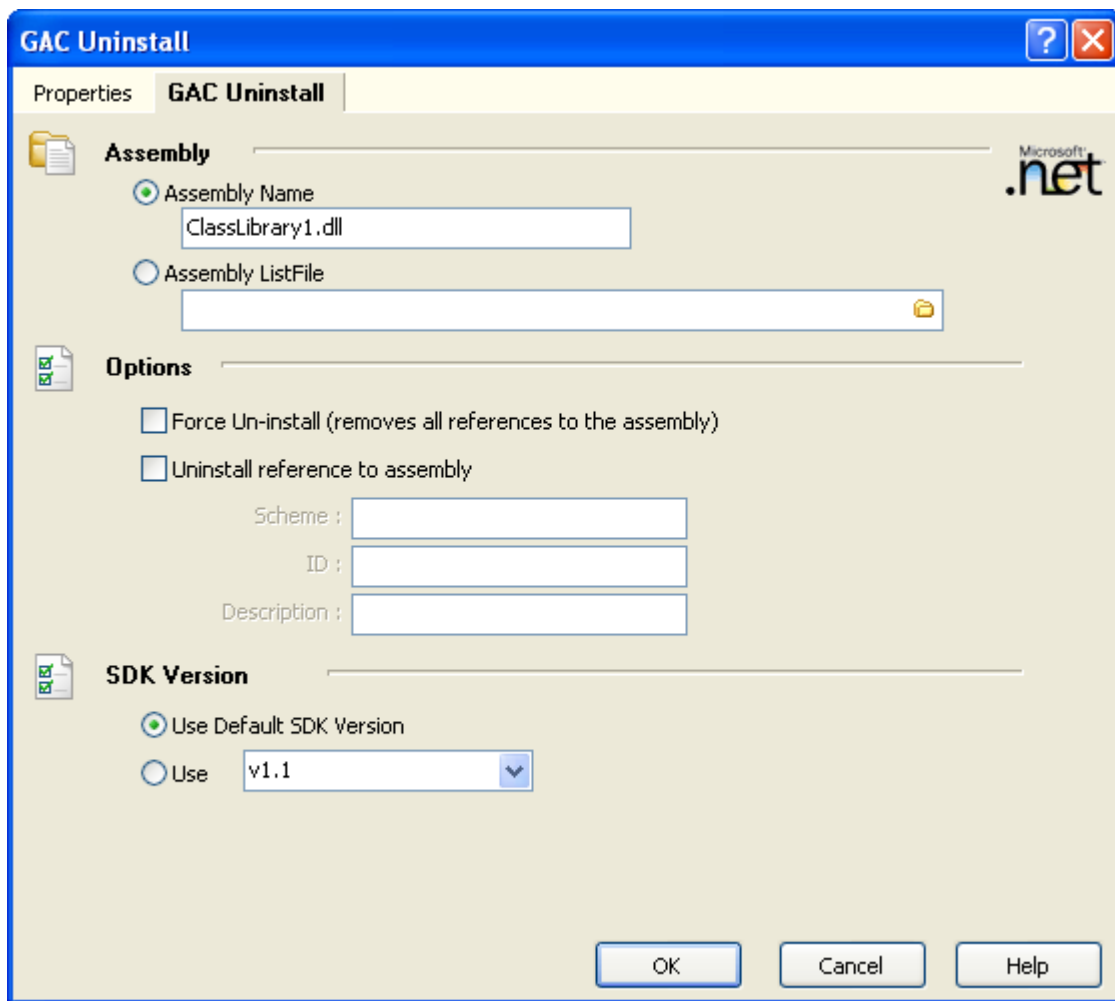
<http://msdn.microsoft.com/library/en-us/cptools/html/cpgrfGlobalAssemblyCacheUtilityGacutil.exe.asp>

#### 5.17.2.8 GAC Uninstall [GACUTIL]

Uninstalls an assembly from the global assembly cache.

This action automates the following options:

/u  
/uf  
/ul  
/ur

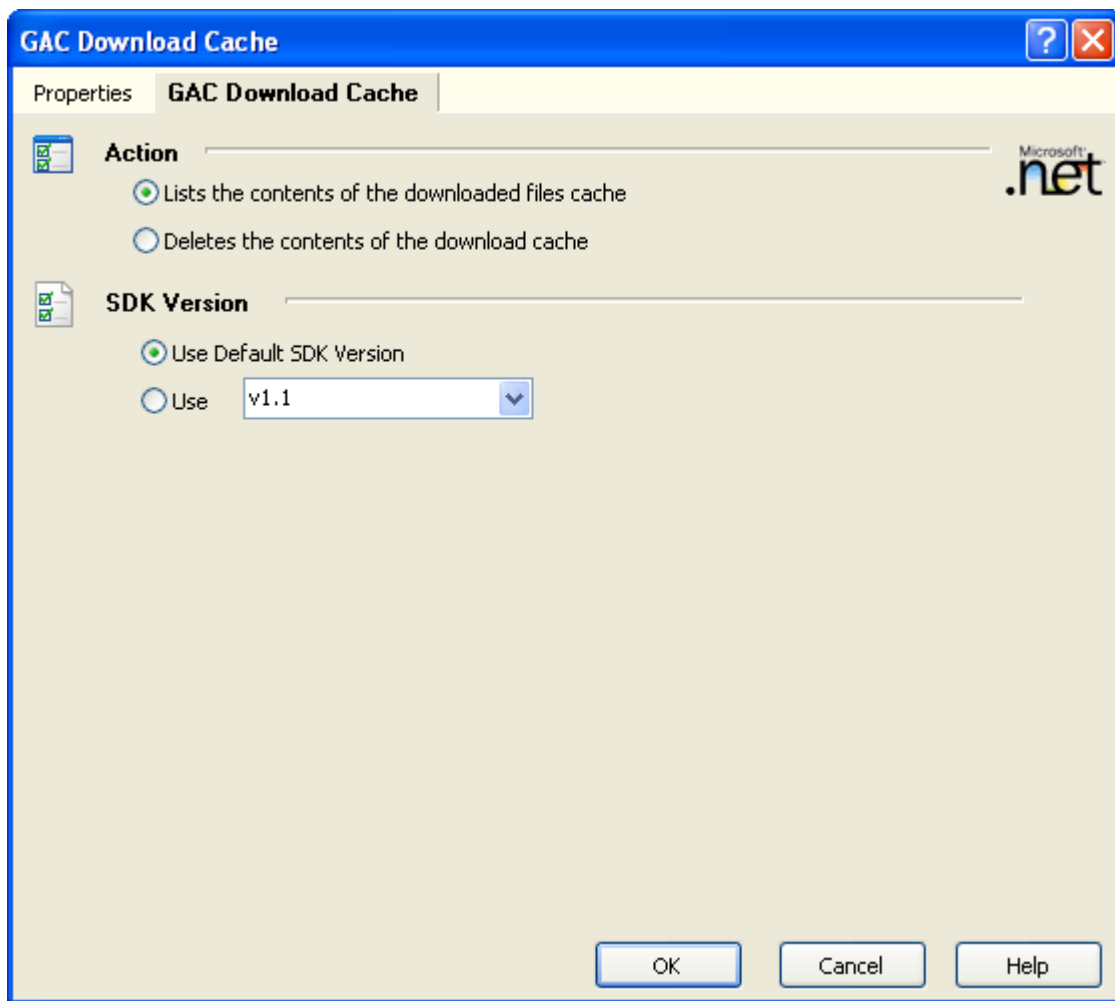


For more information see:

<http://msdn.microsoft.com/library/en-us/cptools/html/cpgrfGlobalAssemblyCacheUtilityGacutilexe.asp>

#### 5.17.2.9 GAC Download Cache [GACUTIL]

Lists or deletes the contents of the downloaded files cache.

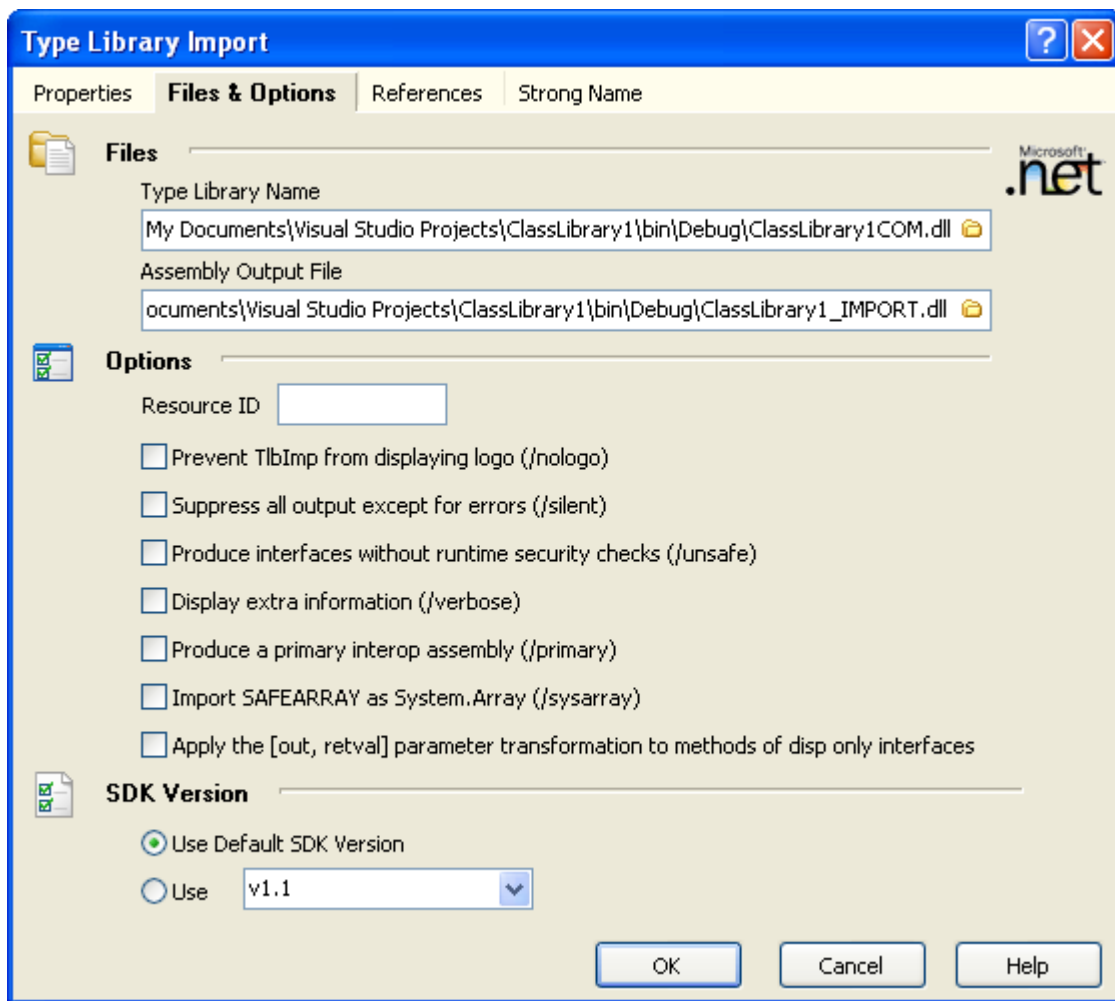


For more information see:

<http://msdn.microsoft.com/library/en-us/cptools/html/cpgrfGlobalAssemblyCacheUtilityGacutil.exe.asp>

#### 5.17.2.10 Type Library Import [TLBIMP]

The Type Library Importer converts the type definitions found within a COM type library into equivalent definitions in a common language runtime assembly. The output of Tlbimp.exe is a binary file (an assembly) that contains runtime metadata for the types defined within the original type library. You can examine this file with tools such as Ildasm.exe.



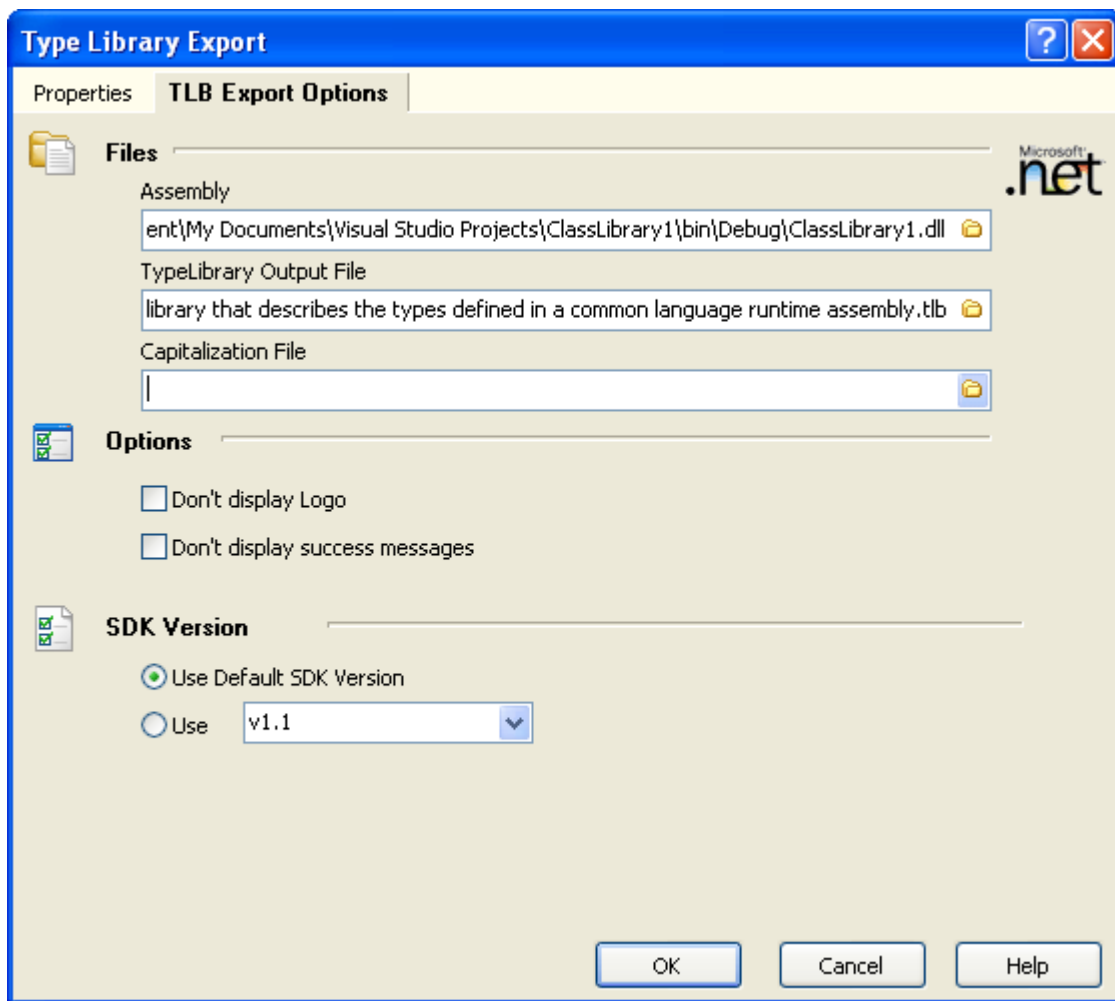
For more information see:

<http://msdn.microsoft.com/library/en-us/cptools/html/cpgrfTypeLibraryImporterTlbimpexe.asp>

#### 5.17.2.11 Type Library Export [TLBEXP]

The Type Library Exporter generates a type library that describes the types defined in a common language runtime assembly.



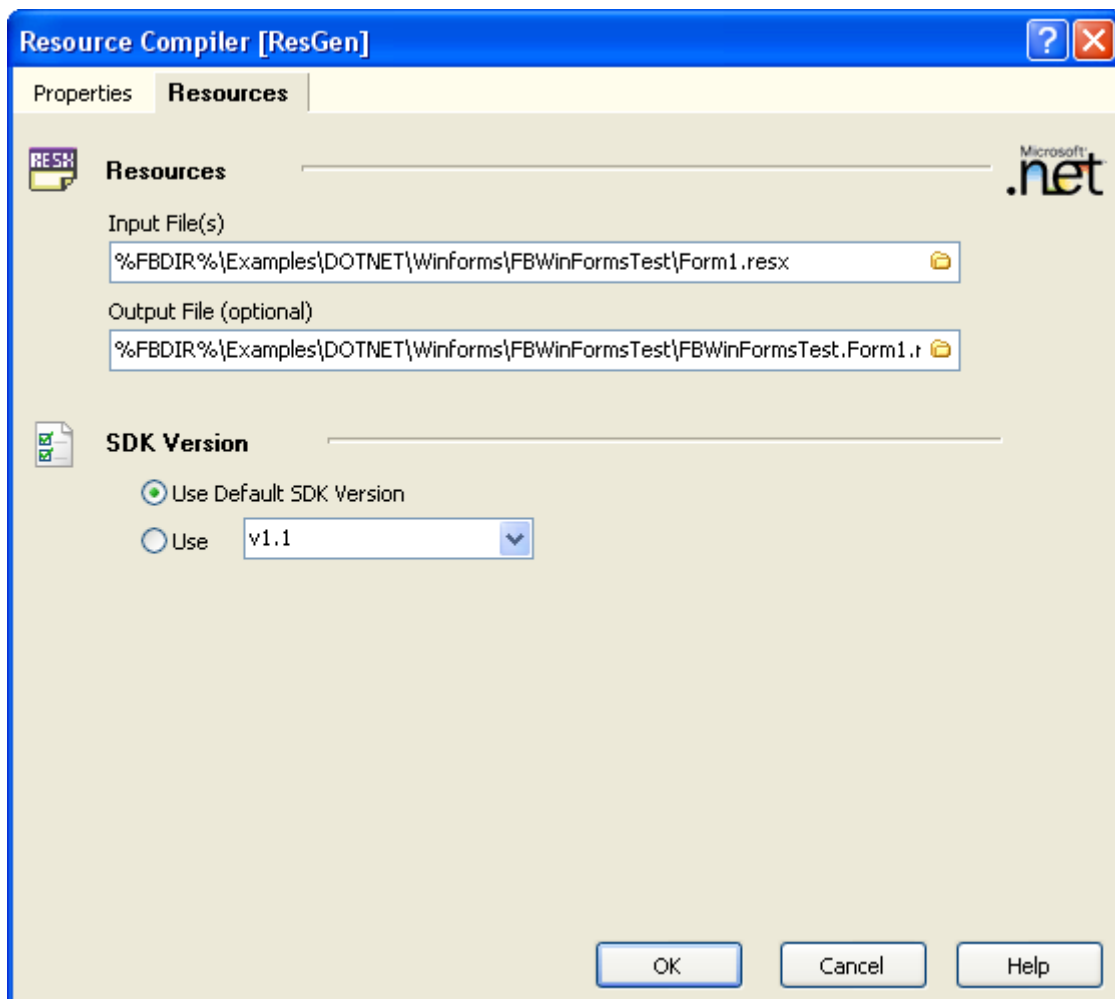


For more information see:

<http://msdn.microsoft.com/library/en-us/cptools/html/cpgrfTypeLibraryExporterTlbExpexe.asp>

#### 5.17.2.12 Run ResGen.exe

Resource File Generator converts .txt files and .resx (XML-based resource format) files to common language runtime binary .resources files that can be embedded in a runtime binary executable or compiled into satellite assemblies.



For more information see :

<http://msdn.microsoft.com/library/default.asp?url=/library/en-us/cptools/html/cpgrfresourcefilegeneratorutilityresgenexe.asp>

### 5.17.3 3rd Party Tools

Some .Net tools have been placed in other categories:

NDoc Action

NUnit Action

MSTest

Nant Project Action

MSBuild Project Action

### 5.17.3.1 FxCop

The FxCop action enables you to automate compliance testing of your .Net assemblies using FxCop

FxCop is a code analysis tool that checks .NET managed code assemblies for conformance to the Microsoft .NET Framework Design Guidelines. It uses reflection, MSIL parsing, and callgraph analysis to inspect assemblies for more than 200 defects in the following areas:

- Library design
- Localization
- Naming conventions
- Performance
- Security

See the FxCop homepage for more information, <http://www.gotdotnet.com/team/fxcop/>

### 5.17.3.2 Dotfuscator

The Dotfuscator action enables you to automate obfuscation of your .Net assemblies using Dotfuscator

The .NET environment provides unprecedented flexibility and power in developing windows applications fast. Once that application is done, you better protect it. That's where Dotfuscator comes in. Dotfuscator provides powerful protection for your .NET code to protect your valuable intellectual property. After all, if your code was worth writing, isn't it worth protecting?

**NOTE :** The Community Edition of Dotfuscator displays a Register dialog when unregistered and an upgrade dialog when registered. This will cause the action to hang indefinitely until aborted. This action will only work correctly with the full commercial version.

See the Dotfuscator homepage for more information, <http://www.preemptive.com/products/dotfuscator/>

### 5.17.3.3 Demeanor

The Demeanor action enables you to automate obfuscation of your .Net assemblies using Wise Owl Demeanor

Demeanor for.NET protects your intellectual property by making it extremely difficult to reverse engineer your .NET applications. Unprotected .NET applications can be easily reverse engineered via decompilation and inspection by many decompiler products.

To counter this threat, WiseOwl has developed Demeanor for.NET -- the best .NET obfuscator available. Demeanor for.NET applies many transformations to your .NET applications that makes them much more difficult to reverse engineer.

Demeanor for.NET obfuscates the names of your types, fields, methods, properties and events by changing their names to meaningless symbols. Demeanor for.NET also obfuscates the metadata of your application, discarding all types and members that aren't needed during runtime. Demeanor for.NET also alters the control flow of your methods so that the resulting code is much harder to understand.

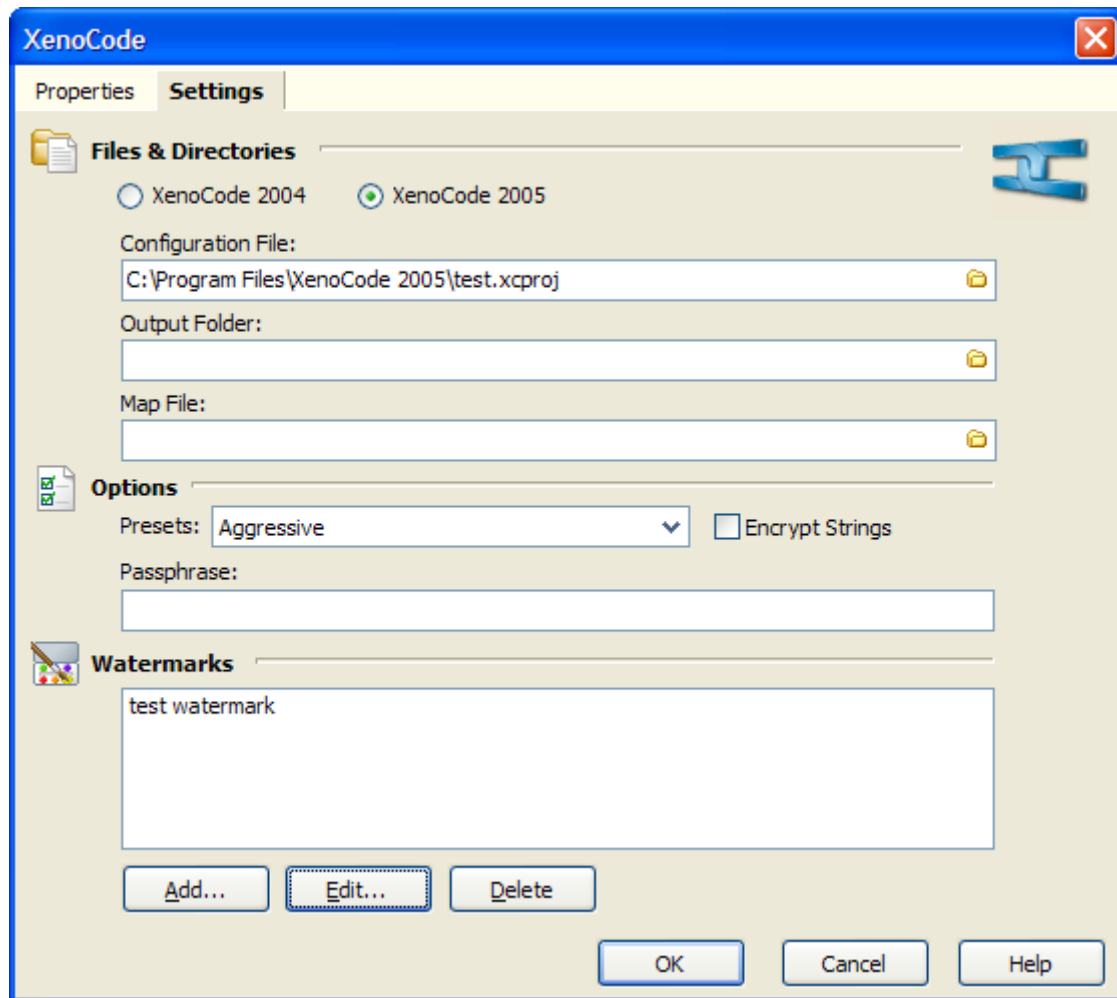
See the Wise Owl Demeanor homepage for more information,  
<http://www.wiseowl.com/products/products.aspx>

#### 5.17.3.4 XenoCode

The XenoCode action enables you to automate obfuscation and optimization of your .Net assemblies using XenoCode

XenoCode is the powerful, flexible, and easy-to-use code protection and deployment solution for .NET developers.

See the XenoCode homepage for more information, <http://www.xenocode.com/>



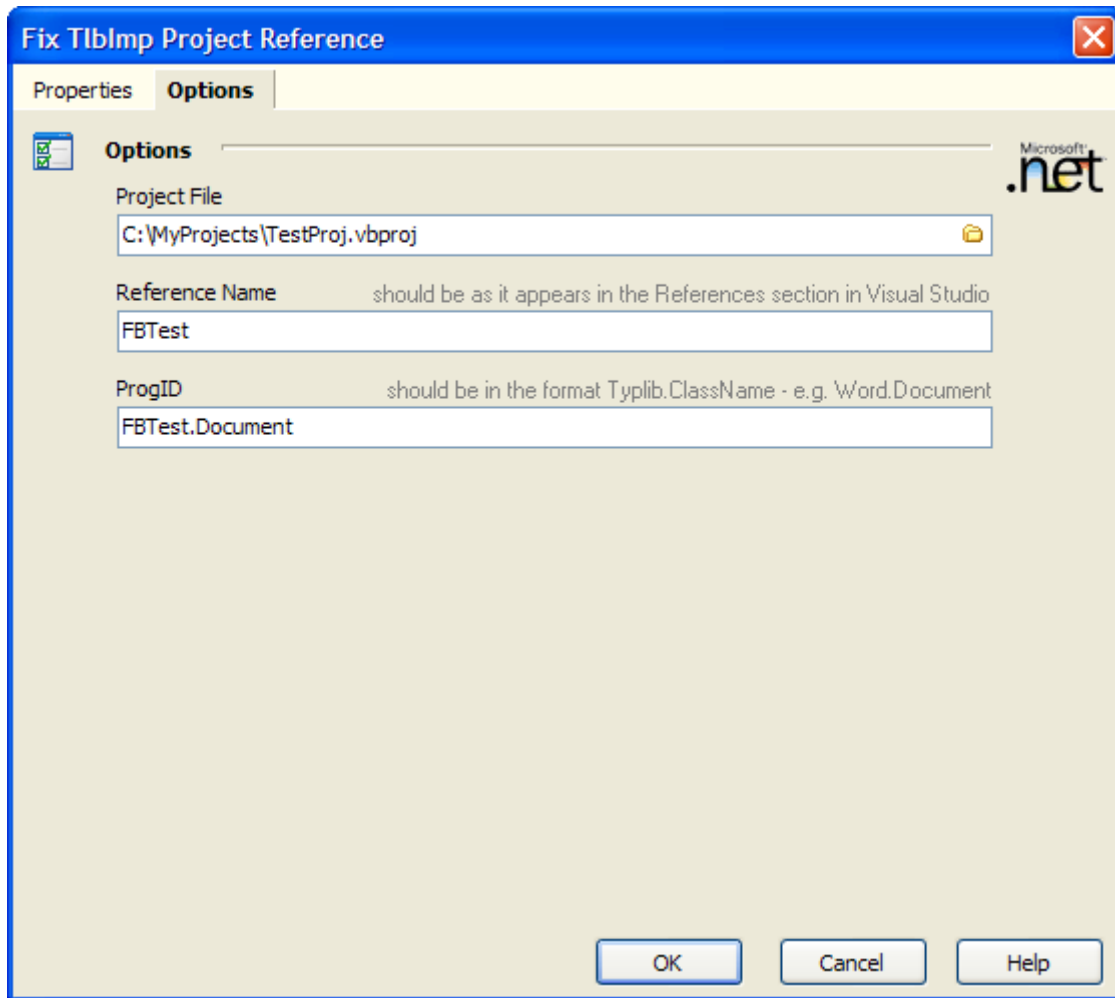
The XenoCode action supports both XenoCode 2004 (version 2.x) and XenoCode 2005 (version 3.x). Set up the locations for the XenoCode executable location and default XenoCode version in the Tools->Options menu.

If no output folder is specified, then the output folder is set to a XenoCode subdirectory of the configuration folder.

## 5.17.4 Other

### 5.17.4.1 Fix TLBImp Project Reference

This action will attempt to fix com interop references in a C#, VB.NET or J# Visual Studio project File. The most reason this might need to be done is if the Typelib guid's have changed since the reference was added to the project in Visual Studio. An example of when this might happen is when a .NET project is importing a VB6 COM dll that is not compiled with binary compatibility enabled. VB6 will change the typelib guid's every time the project is built. This action should be used after the VB6 Action and before the VS.NET action or the C#/VB.NET or J# Project compiler actions.



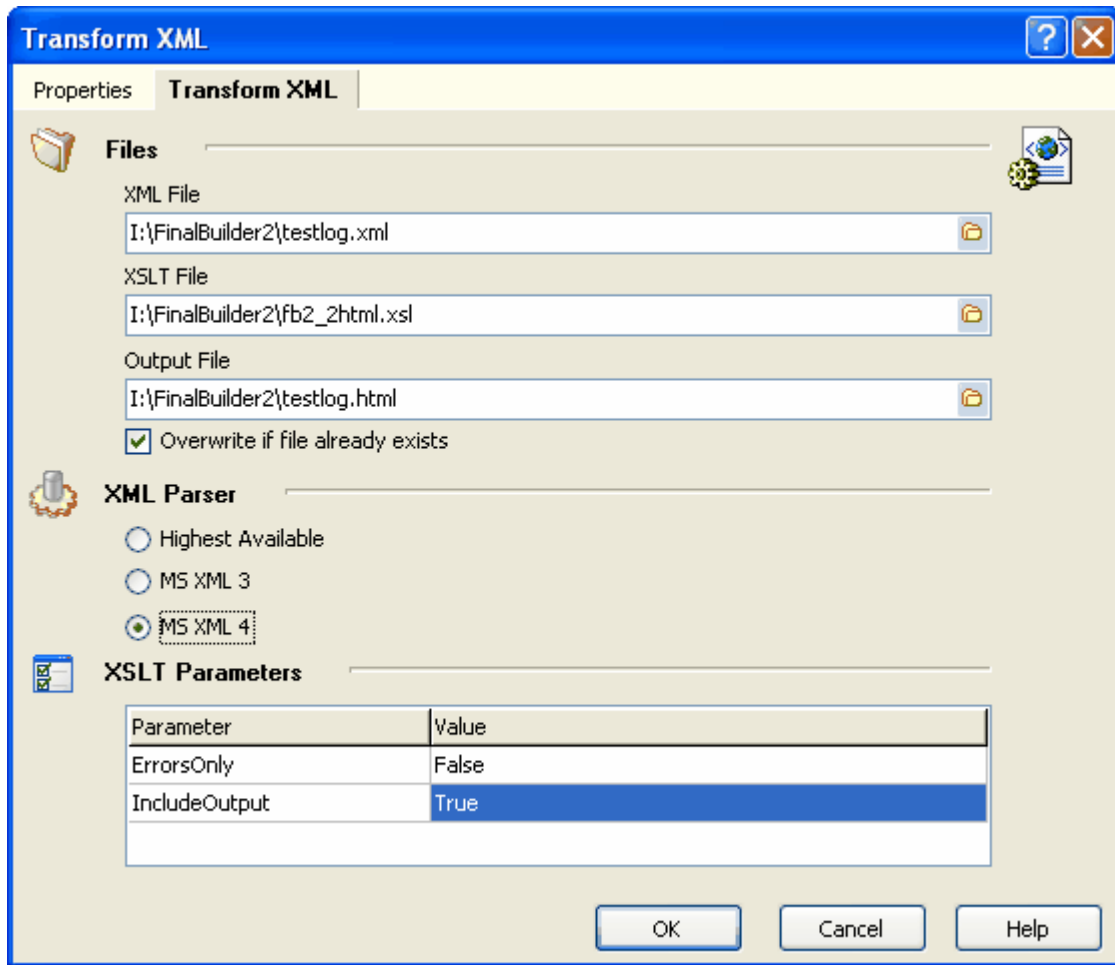
The project File property should point to the VS.NET project file (not the solution) that references the com dll. The reference name is the name of the reference as it appears in the references section of the solution explorer in Visual Studio. The ProgID should be set to the format TypelibName.ClassName, for example Word.Application. This prog id enables the action to find the typelib entry in the registry and find the correct guid and version numbers for the com dll. The vs.net project file must be writeable as the action will update the reference entry in the file.

## 5.18 XML Actions

### 5.18.1 Transform XML

[Professional Edition]

This action performs an XSL Transform using the Microsoft XML parser.

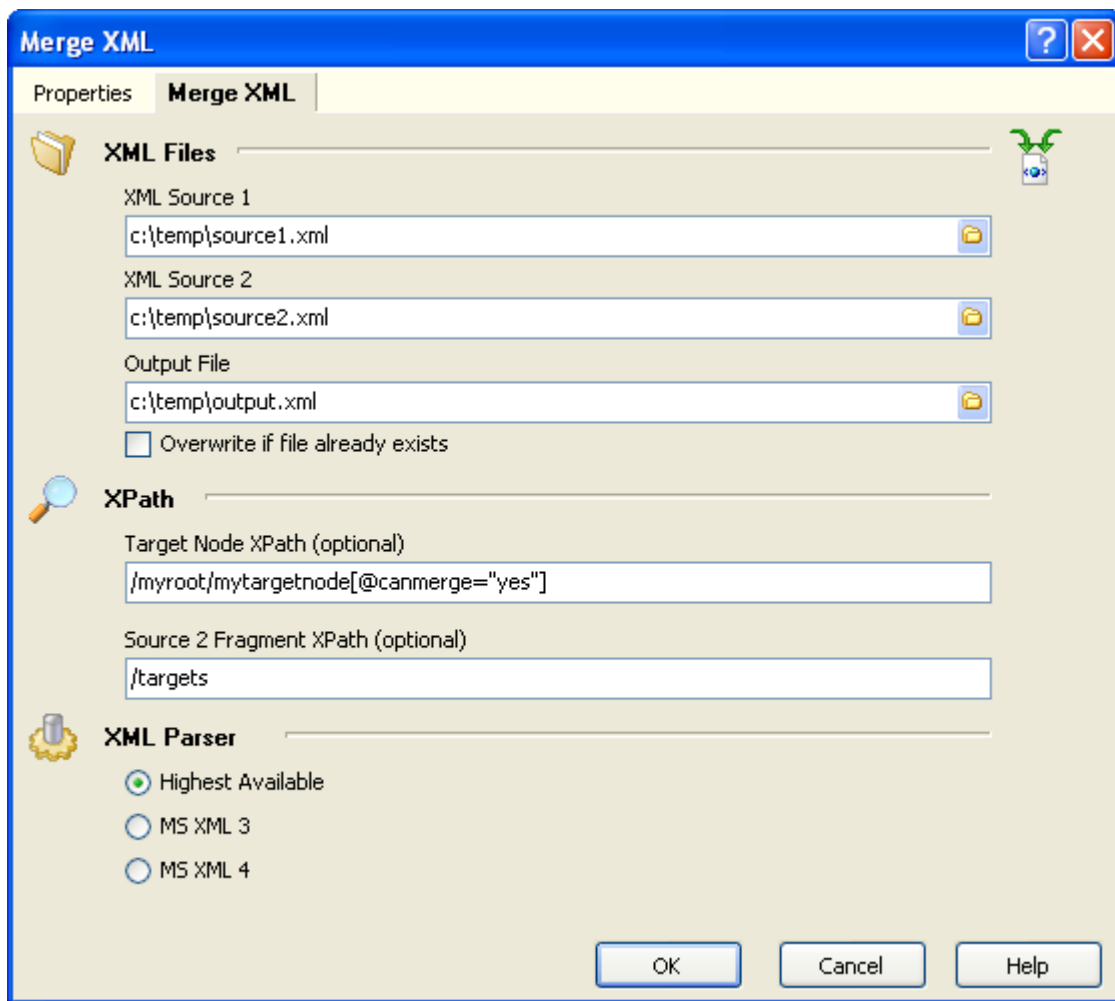


The XSLT Parameters allows you to provide dynamic values (such as build numbers etc) to your stylesheet, which can be used to alter the output of the transform.

### 5.18.2 Merge XML Action

[Professional Edition]

This action merges two XML documents. Source 2 will be merged into Source 1 and saved as a new document.



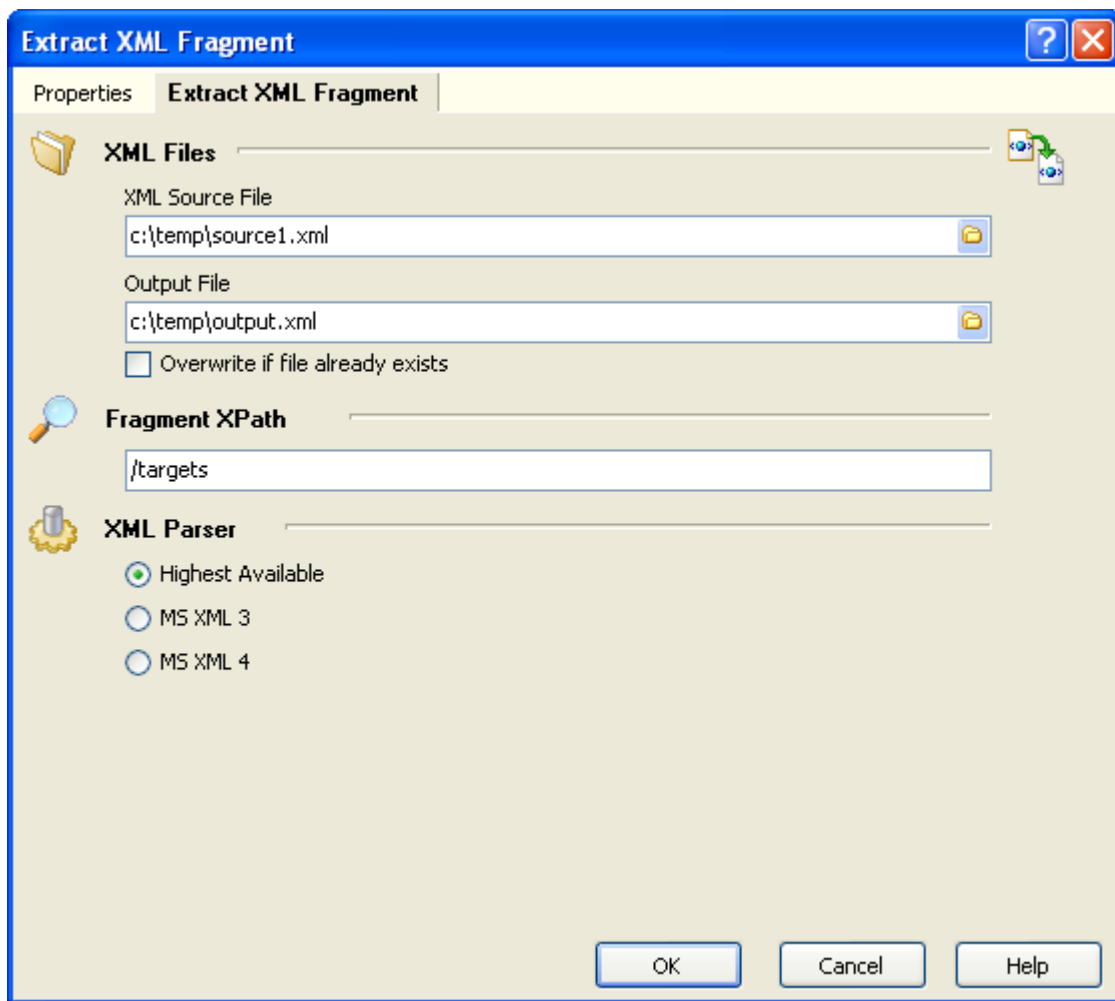
The TargetNode XPath option allows you to specify the node where the Source2 document will be inserted.

The Source 3 Fragment XPath option allows you to specify a document fragment to merge rather than the whole document.

### 5.18.3 Extract XML Fragment Action

[Professional Edition]

This action extracts an xml document fragment from a source xml document and saves it as a new xml document.

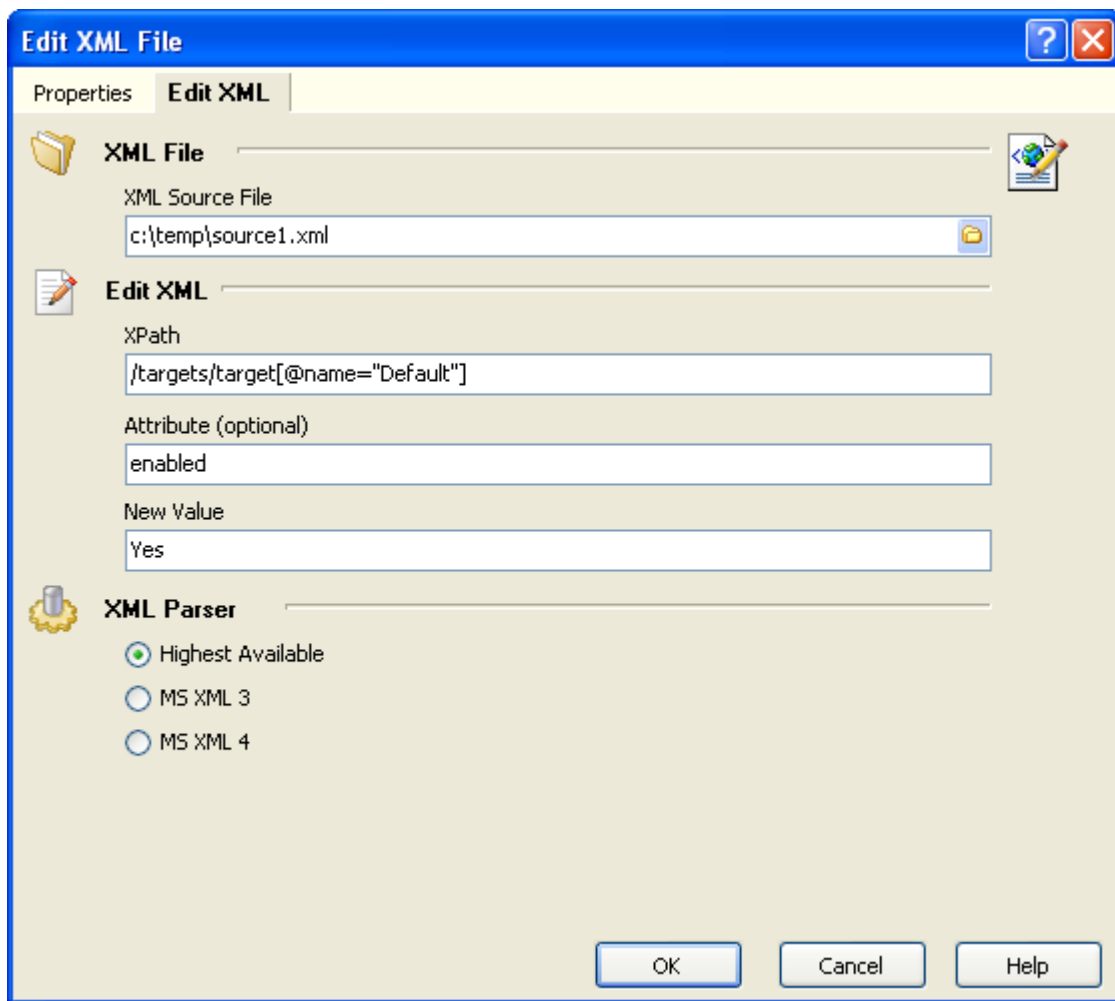


#### 5.18.4 Edit XML File Action

[Professional Edition]

This action allows you to modify a value in an XML document. The value to modify is selected using XPath, if the Attribute field is empty the node text will be set.



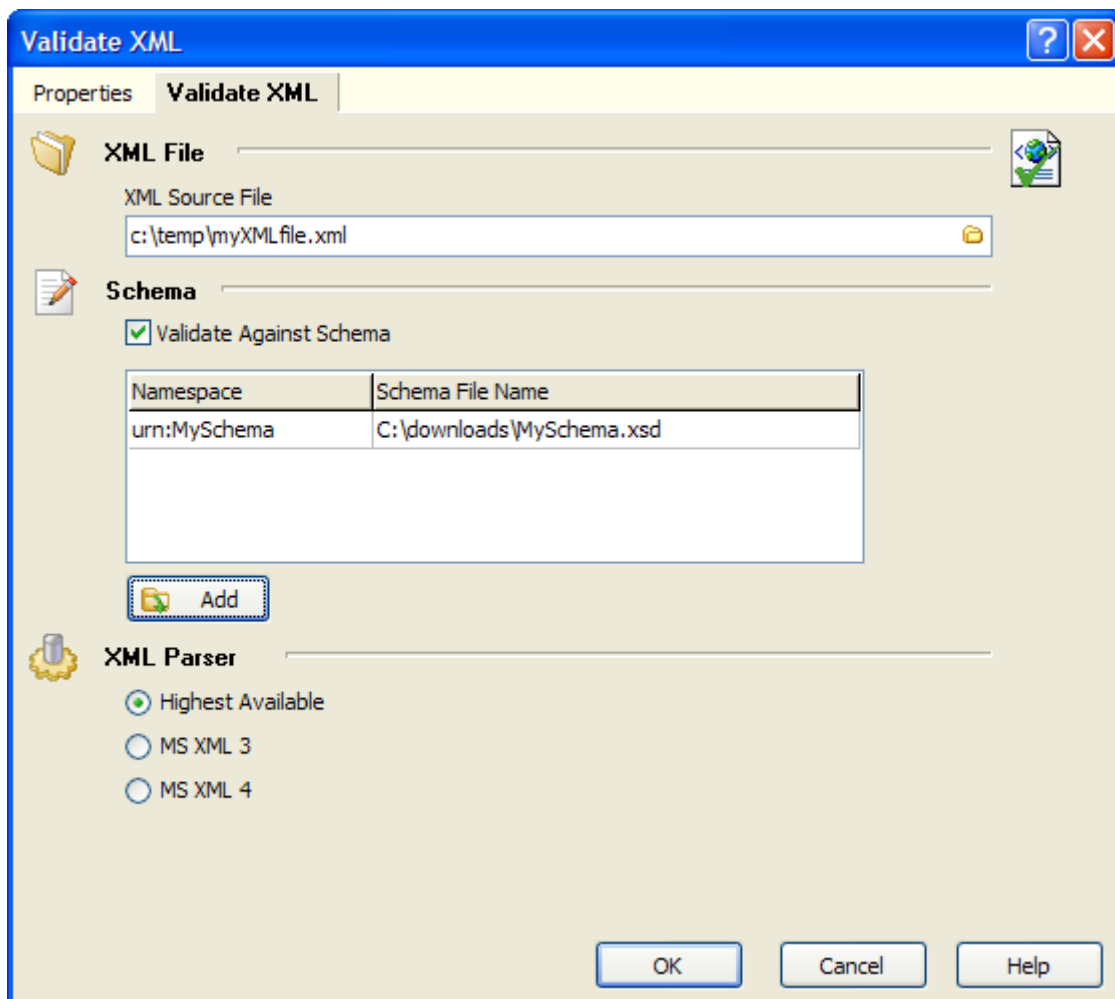


### 5.18.5 Validate XML File

[Professional Edition]

This action validates an XML file using the Microsoft XML parser.

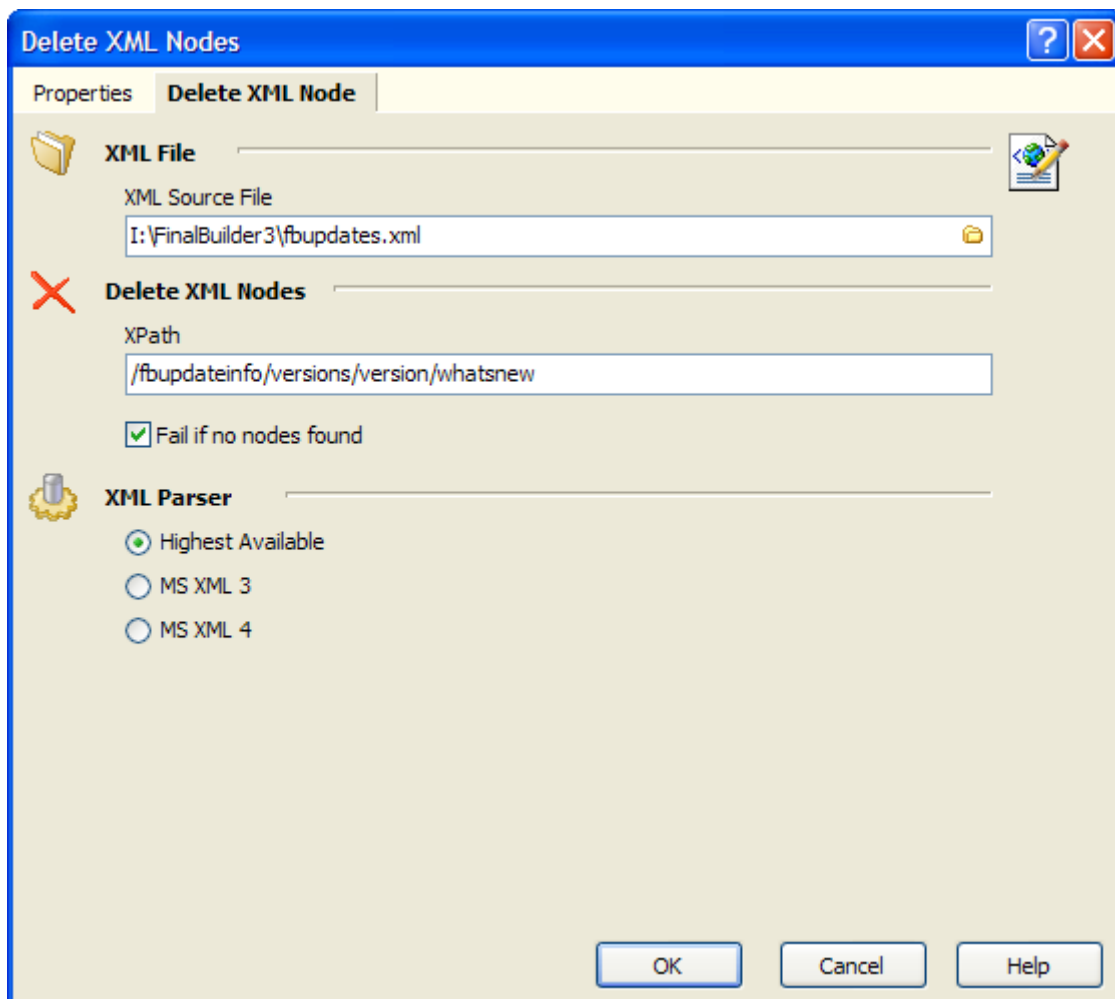
You can validate against schemas, or if no schemas are specified then the action parses the xml file and reports any errors found.



### 5.18.6 Delete XML Nodes

[Professional Edition]

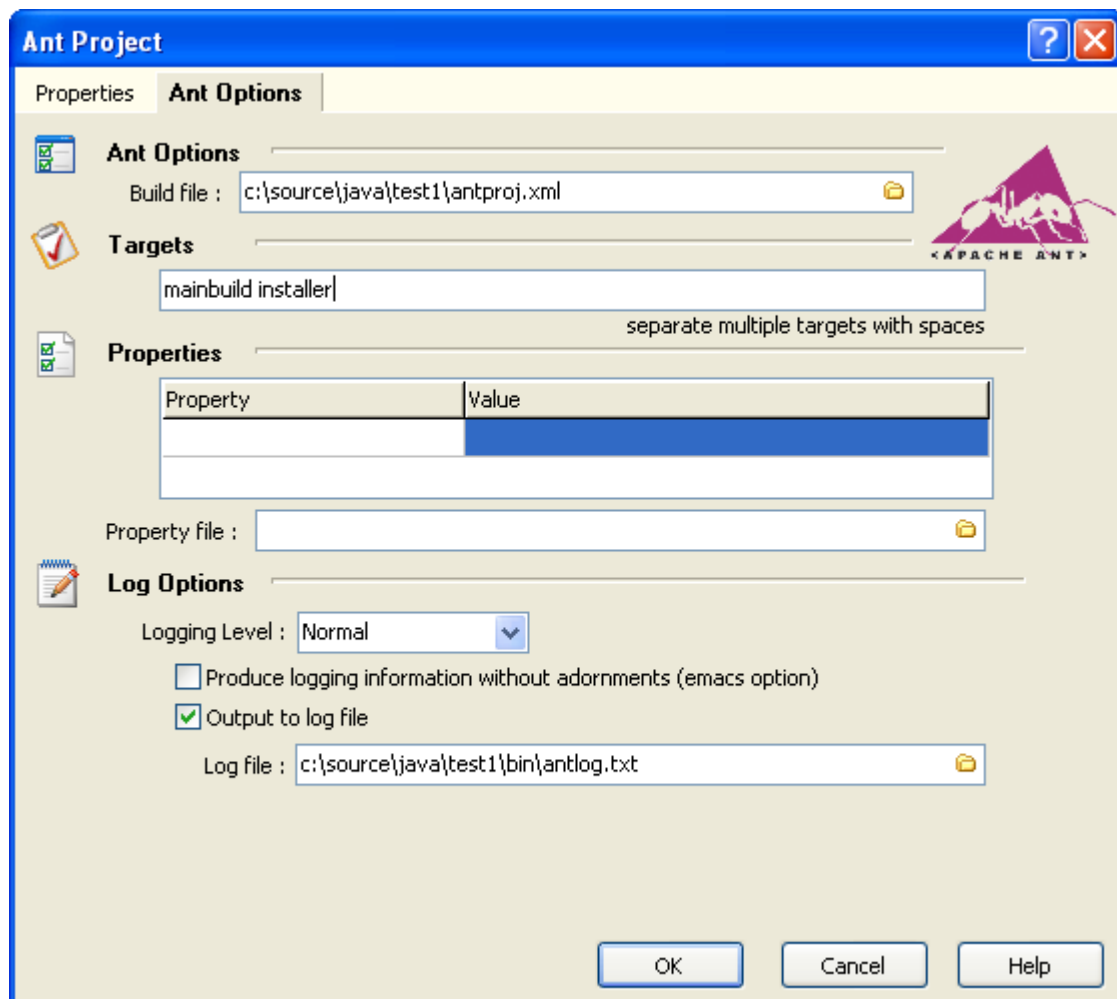
This action allows you to delete a node or set of nodes in an XML document. Set the XPath statement to select the nodes you wish to delete.



## 5.19 Build Tools

### 5.19.1 Ant Project Action

This action executes the Apache Ant command line build tool, making it easy to integrate legacy build scripts into your FinalBuilder Build process.



### 5.19.2 Nant Project Action

This action executes the Nant command line build tool, making it easy to integrate legacy build scripts into your FinalBuilder Build process.

**Nant Project**

Properties **NAnt**

**NAnt Options**

Build file : c:\source\csharp\test1\test1.build

Default Framework :

**Targets**

main installer

separate multiple targets with spaces

**Properties**

| Property | Value |
|----------|-------|
|          |       |

**Log Options**

Logging Level : Normal

Indentation Level : <default>

☐ Surpress display of logo banner

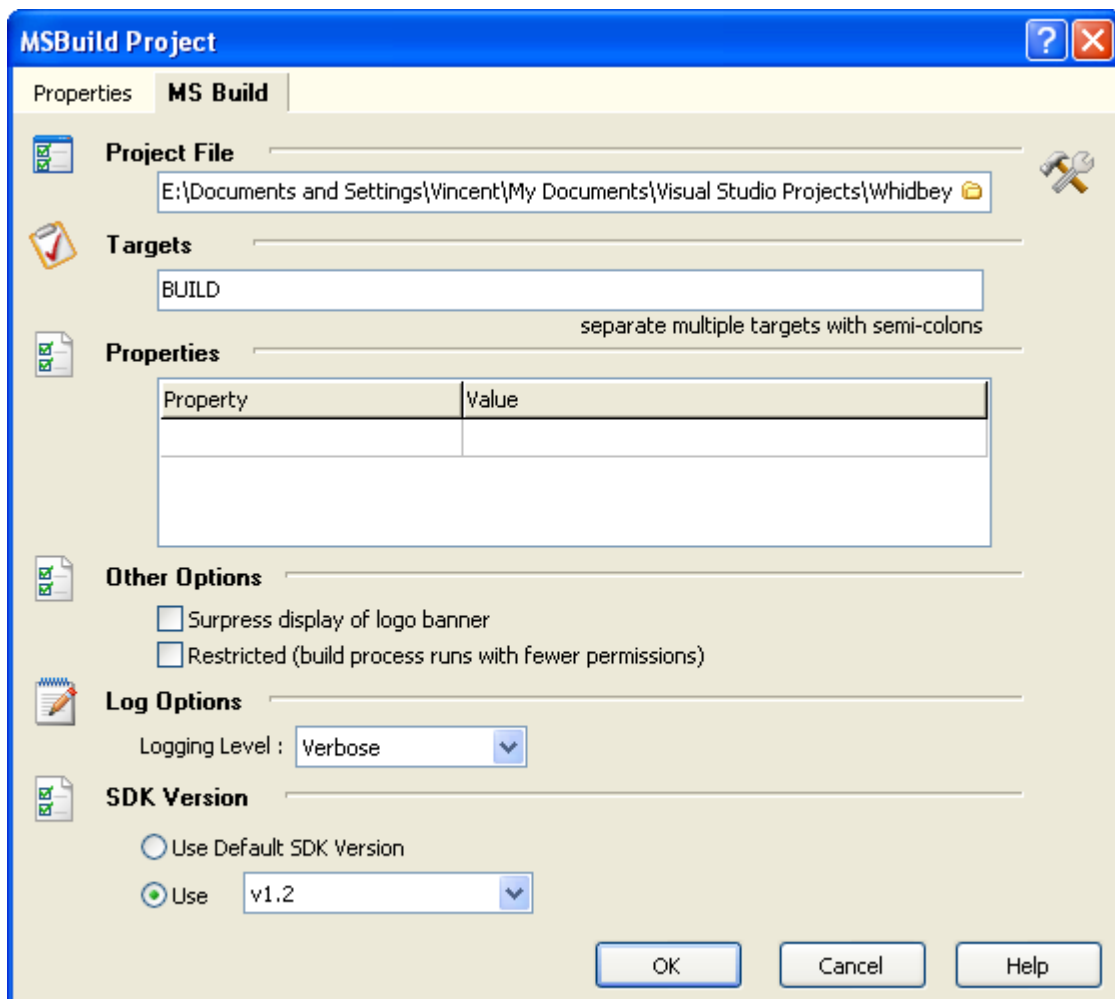
☒ Output to log file

Log file : c:\source\csharp\test1\bin\test1.log

OK Cancel Help

### 5.19.3 MSBuild Project Action

This action runs the Microsoft MSBuild command line tool that is part of the .NET Framework 1.2. MSBuild projects are the format used by Visual Studio.NET 2004/Whidbey. Whidbey projects have the same file extension as earlier versions of visual studio (ie csproj, vbproj) but are actually MSBuild scripts.



## 5.20 Database

### 5.20.1 SQL Server Actions

#### 5.20.1.1 Execute SQL Action

[Professional Edition]

Execute an SQL statement against Microsoft SQL Server and capture the result set.

Connection & Security Tab

The screenshot shows the 'Execute SQL' dialog box with the 'Connection & Security' tab selected. The dialog has four tabs: 'Properties', 'Connection & Security', 'Input', and 'Output'. The 'SQL Server' section contains a 'Server' text box with 'localhost' and a 'View List' button. The 'Use database' text box contains 'northwind'. The 'SQL Server Tool' section has two radio buttons: 'ISQL (uses DB-Library to communicate with Microsoft® SQL Server)' which is selected, and 'OSQL (uses ODBC to communicate with the server)'. The 'Security' section has a checked checkbox for 'Use integrated security', and empty text boxes for 'Username' and 'Password'. At the bottom are 'OK', 'Cancel', and 'Help' buttons.

**Execute SQL**

Properties **Connection & Security** Input Output

**SQL Server**

Server : localhost View List

Use database : northwind

**SQL Server Tool**

☒ ISQL (uses DB-Library to communicate with Microsoft® SQL Server)

☐ OSQL (uses ODBC to communicate with the server)

**Security**

☒ Use integrated security

Username :

Password :

OK Cancel Help

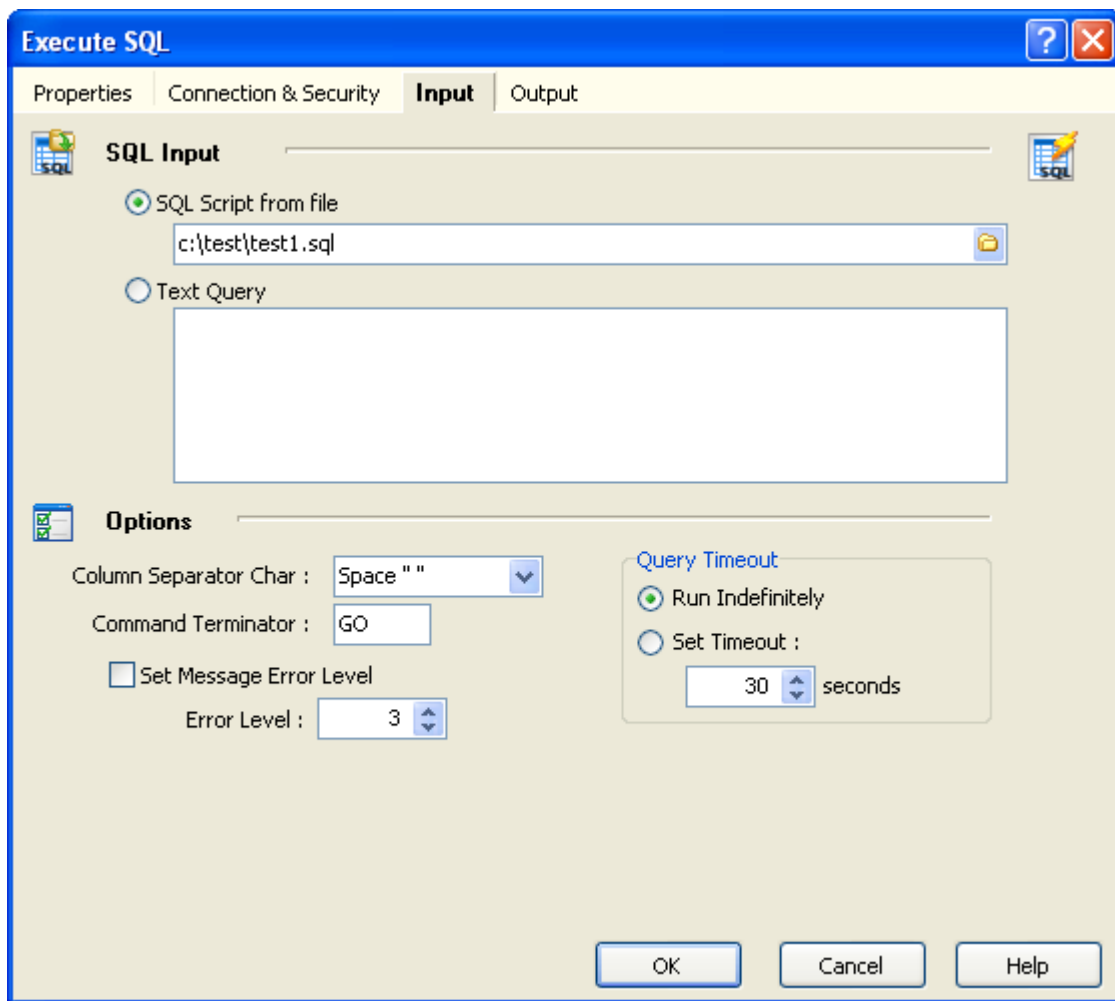
Server - Specify the MS SQL Server to use. Clicking "View List" will attempt to locate any SQL Servers on the network.

Use Database - Specify a database to run the SQL against.

SQL Server Tool - choose the method (isql or osql) to use to connect with.

Security - If not using integrated (ie. Windows) security, you should specify the username and password to use.

Input Tab



SQL Input - SQL Script from file - specify a file which contains an SQL Script.  
- Text Query - Specify the SQL Query to execute.

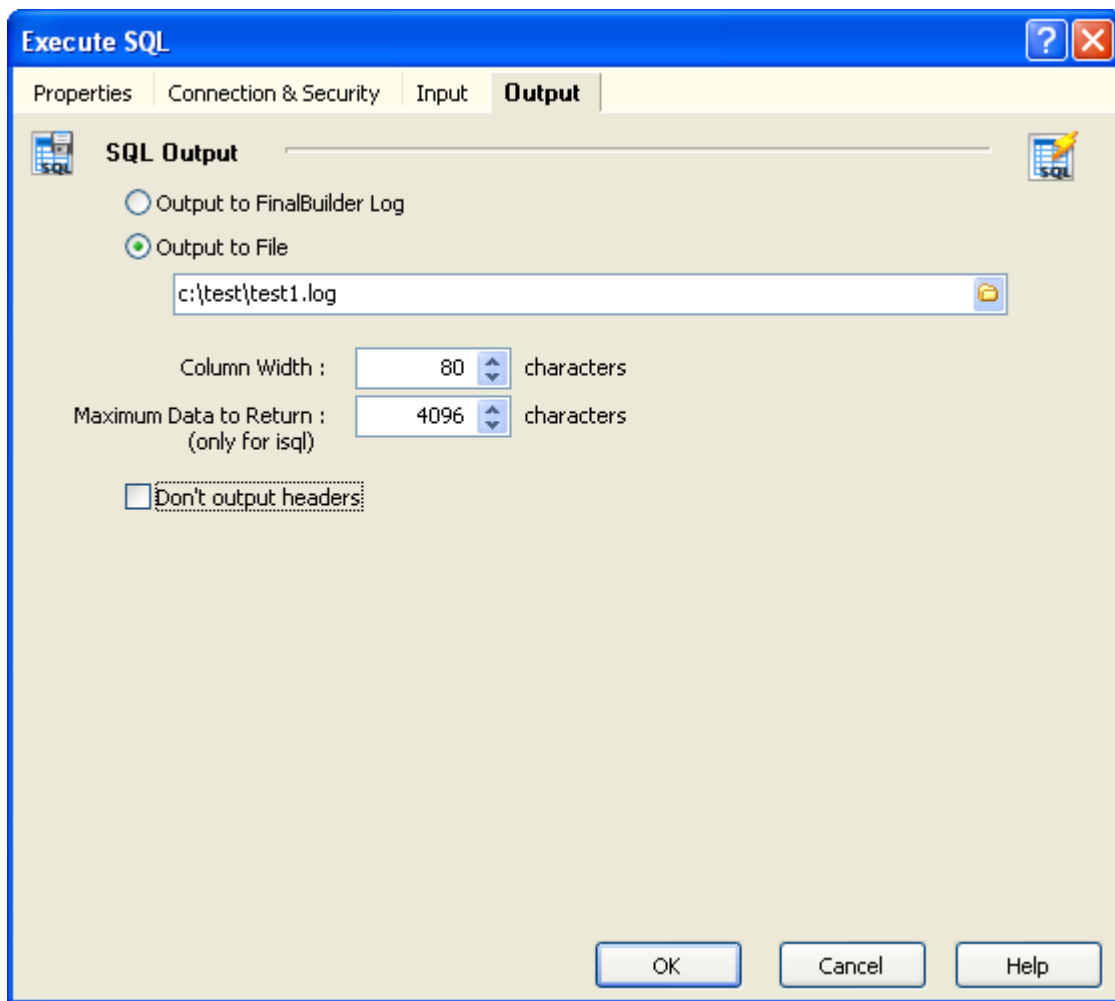
For information on the other options see:

isql : [http://msdn.microsoft.com/library/en-us/coprompt/cp\\_isql\\_8r39.asp](http://msdn.microsoft.com/library/en-us/coprompt/cp_isql_8r39.asp)

osql : [http://msdn.microsoft.com/library/en-us/coprompt/cp\\_osql\\_1wxl.asp](http://msdn.microsoft.com/library/en-us/coprompt/cp_osql_1wxl.asp)

Output Tab





The query results can either be output to a file or to the FinalBuilder Log.  
For information on the other options see:

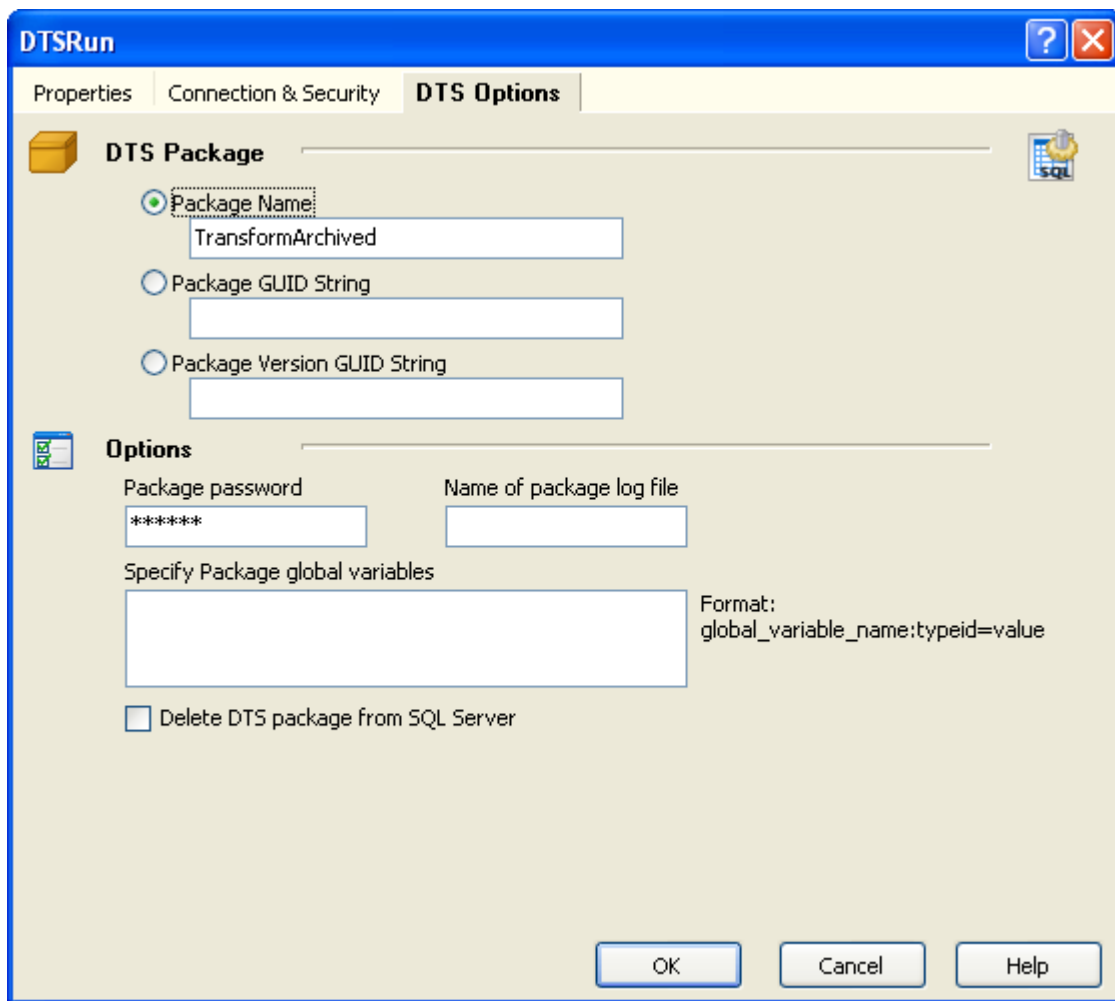
isql : [http://msdn.microsoft.com/library/en-us/coprompt/cp\\_isql\\_8r39.asp](http://msdn.microsoft.com/library/en-us/coprompt/cp_isql_8r39.asp)

osql : [http://msdn.microsoft.com/library/en-us/coprompt/cp\\_osql\\_1wxl.asp](http://msdn.microsoft.com/library/en-us/coprompt/cp_osql_1wxl.asp)

#### 5.20.1.2 DTSTRun Action

[Professional Edition]

The dtstrun utility executes a package created using Data Transformation Services (DTS). The DTS package can be stored in the Microsoft® SQL Server™ msdb database, a COM-structured storage file, or SQL Server Meta Data Services.



For information on the options for this see:

[http://msdn.microsoft.com/library/en-us/coprompt/cp\\_dtsrun\\_95kp.asp](http://msdn.microsoft.com/library/en-us/coprompt/cp_dtsrun_95kp.asp)

#### 5.20.1.3 SQL Server Backup Database

Performs a backup operation on the entire database or only the transaction log. The file name for a database backup is generated automatically as follows:  
dbname\_db\_yyyyMMddhhmm.BAK

The screenshot shows the 'SQL Server Backup Database' dialog box with the 'Maintenance Options' tab selected. The 'Database(s) to perform maintenance on' section has three radio button options: 'Database Name' (selected), 'Databases in Maintenance Plan (specify Plan Name)', and 'Databases in Maintenance Plan (specify Plan ID)'. The 'Database Name' option has a text box containing 'Northwind'. The 'Reporting Options' section includes a 'Report to File' text box with 'C:\temp\NorthwindBackupReport.html', a checked checkbox for 'HTML report (default is text)', a 'Send Report to Operator through SQL Mail' text box, and a checked checkbox for 'Write History to msdb.dbo.sysdbmaintplan\_history'. At the bottom are 'OK', 'Cancel', and 'Help' buttons.

**SQL Server Backup Database**

Properties | Connection & Security | **Maintenance Options** | Backup Details

**Database(s) to perform maintenance on**

☒ Database Name  
Northwind

☐ Databases in Maintenance Plan (specify Plan Name)

☐ Databases in Maintenance Plan (specify Plan ID)

**Reporting Options**

Report to File  
C:\temp\NorthwindBackupReport.html

☒ HTML report (default is text)

Send Report to Operator through SQL Mail

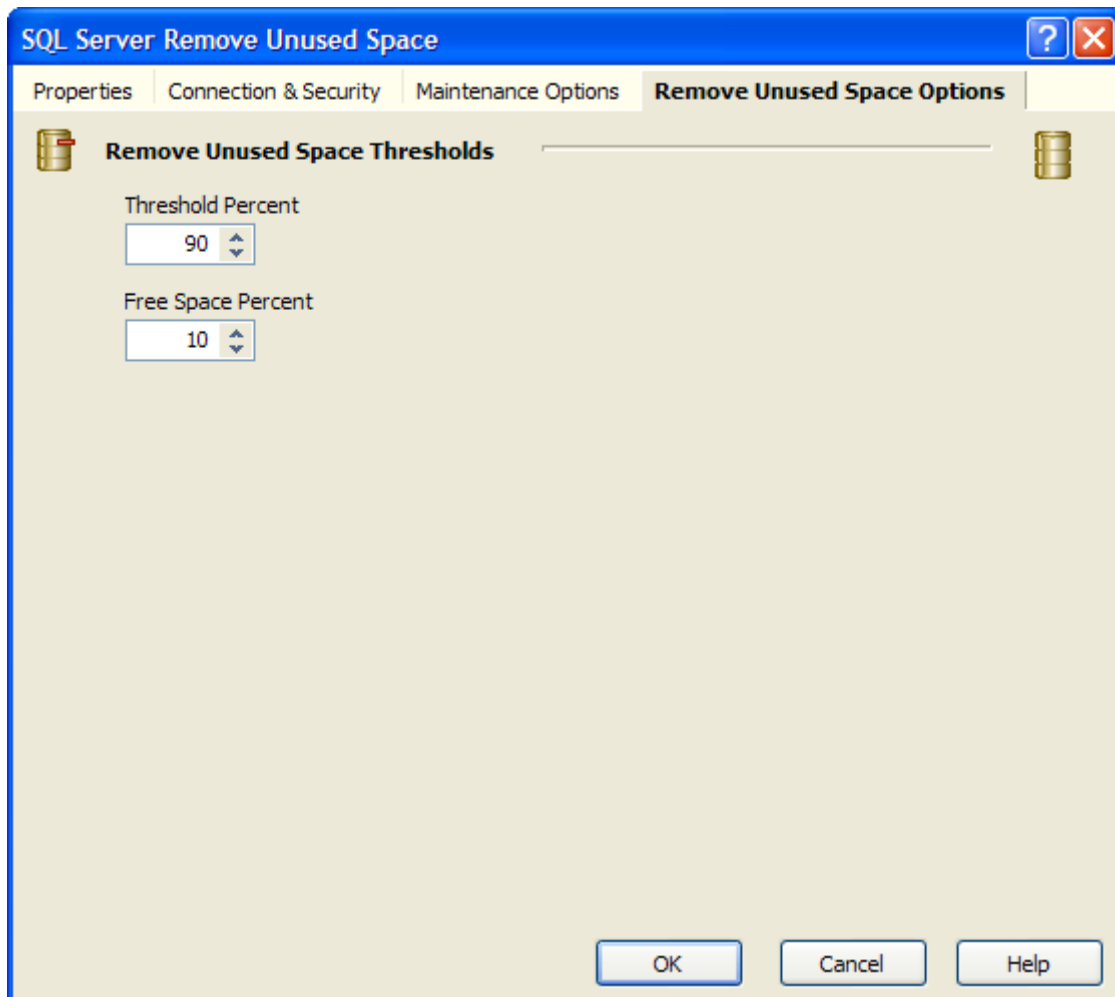
☒ Write History to msdb.dbo.sysdbmaintplan\_history

OK Cancel Help

For more information on the sqlmaint tool see: [http://msdn.microsoft.com/library/en-us/coprompt/cp\\_sqlmaint\\_19ix.asp](http://msdn.microsoft.com/library/en-us/coprompt/cp_sqlmaint_19ix.asp)

#### 5.20.1.4 SQL Server Remove Unused Space

This action will ask SQL Server to remove unused space from the specified database. This option is only useful for databases that are defined to grow automatically. Threshold\_percent specifies in megabytes the size that the database must reach before sqlmaint attempts to remove unused data space. If the database is smaller than the threshold\_percent, no action is taken. Free\_percent specifies how much unused space must remain in the database, specified as a percentage of the final size of the database. For example, if a 200-MB database contains 100 MB of data, specifying 10 for free\_percent results in the final database size being 110 MB. Note that a database will not be expanded if it is smaller than free\_percent plus the amount of data in the database. For example, if a 108-MB database has 100 MB of data, specifying 10 for free\_percent will not expand the database to 110 MB; it will remain at 108 MB.



Specify the threshold percent and free space percent (see above).

For more information on the sqlmaint tool see: [http://msdn.microsoft.com/library/en-us/coprompt/cp\\_sqlmaint\\_19ix.asp](http://msdn.microsoft.com/library/en-us/coprompt/cp_sqlmaint_19ix.asp)

#### 5.20.1.5 SQL Server Check Database

The Check Database action will Check allocation and structural integrity of Database objects.

The "Don't Check Indexes" option specifies that nonclustered indexes for nonsystem tables should not be checked. This decreases the overall execution time because it does not check nonclustered indexes for user-defined tables. "Don't Check Indexes" has no effect on system tables, because system table indexes are always checked.

For more information on the sqlmaint tool see: [http://msdn.microsoft.com/library/en-us/coprompt/cp\\_sqlmaint\\_19ix.asp](http://msdn.microsoft.com/library/en-us/coprompt/cp_sqlmaint_19ix.asp)

#### 5.20.1.6 SQL Server Check Catalogue

The Check Catalogue action checks for consistency in and between system tables in the specified database.

This action checks that every data type in syscolumns has a matching entry in systypes

and that every table and view in sysobjects has at least one column in syscolumns.

For more information on the sqlmaint tool see: [http://msdn.microsoft.com/library/en-us/coprompt/cp\\_sqlmaint\\_19ix.asp](http://msdn.microsoft.com/library/en-us/coprompt/cp_sqlmaint_19ix.asp)

#### **5.20.1.7 SQL Server Update DB Statistics**

The Update DB Statistics action updates information about the distribution of key values for one or more statistics groups (collections) in the specified database.

For more information on the sqlmaint tool see: [http://msdn.microsoft.com/library/en-us/coprompt/cp\\_sqlmaint\\_19ix.asp](http://msdn.microsoft.com/library/en-us/coprompt/cp_sqlmaint_19ix.asp)

#### **5.20.1.8 SQL Server Rebuild Indexes**

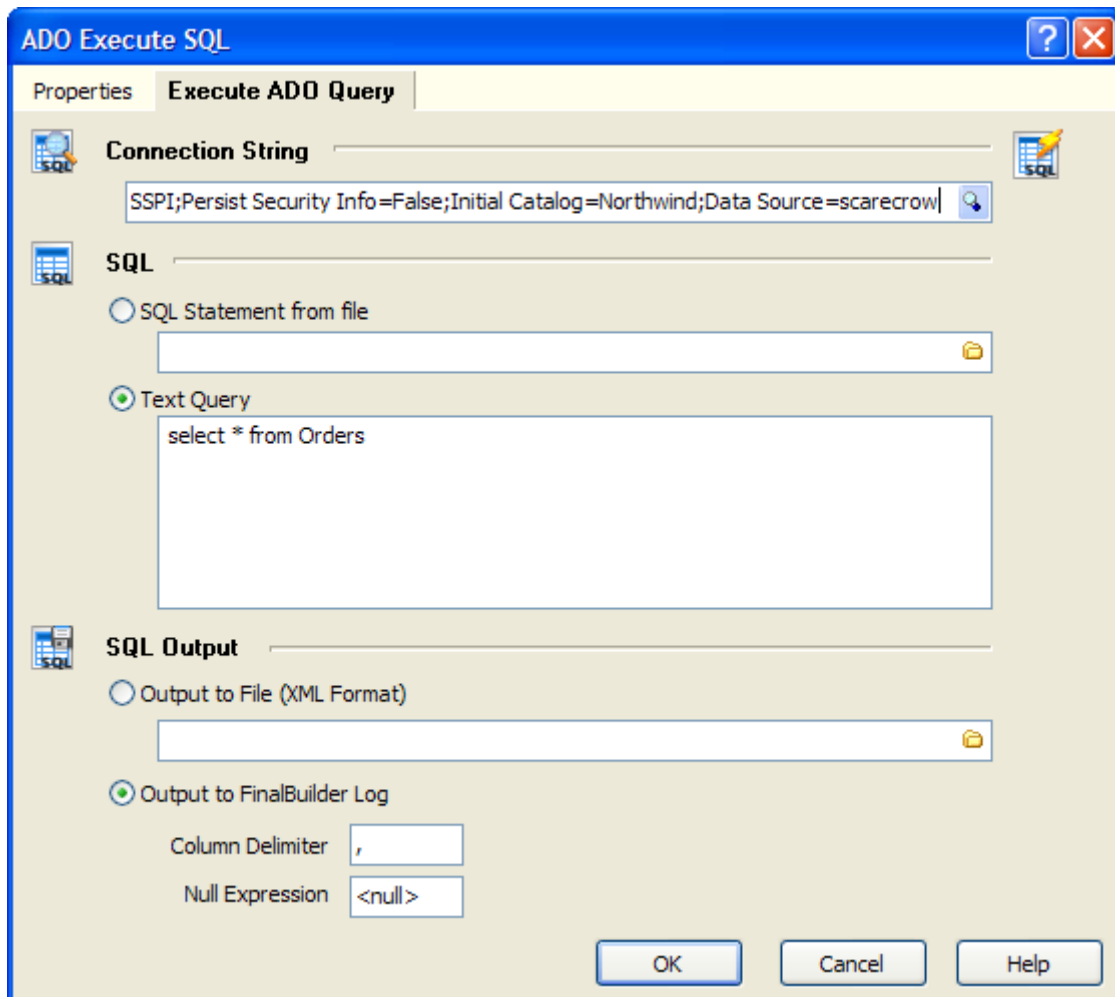
The SQL Server Rebuild Indexes action specifies that indexes on tables in the target database should be rebuilt by using the "free space" percent value as the inverse of the fill factor. For example, if free space percentage is 30, then the fill factor used is 70. If a free space percentage value of 100 is specified, then the indexes are rebuilt with the original fill factor value.

For more information on the sqlmaint tool see: [http://msdn.microsoft.com/library/en-us/coprompt/cp\\_sqlmaint\\_19ix.asp](http://msdn.microsoft.com/library/en-us/coprompt/cp_sqlmaint_19ix.asp)

### **5.20.2 ADO Execute SQL**

[Professional Edition]

The ADO Query action enables you to execute a SQL statement against an ADO database connection



**Connection String** - specify a connection string to your ADO data source. You can use the built in connection string builder to create and test your connection string.

**SQL** - either specify a file containing SQL, or specify the SQL statement in the text field

**SQL Output** - Choose if you want the results output to the FB log, or to and XML file using the ADO XML file format.

### 5.20.3 ADO Execute Stored Procedure

[Professional Edition]

The ADO Stored Procedure action enables you to automate the execution of stored procedures using ADO as part of your build process.

The action property pages helps you build your connection string, displays a list of stored procedures in the target database, and will retrieve the parameters for the selected stored procedure. The value of each input parameter can be set, and the out values and return values can be saved to a variable.

It is important to set "Stored Procedure Returns Recordset" correctly, as this controls how

the action internally calls the stored procedure.

NOTE: This action is only available in the Professional Edition of FinalBuilder

**ADO Execute Stored Procedure**

**Properties** **Details**

**Connection String**  
y=SSPI;Persist Security Info=False;Initial Catalog=Northwind;Data Source=scarecrow

**Stored Procedure**  
CustOrderHist Refresh

| Parameter Name | Type   | Parameter Input Value | Output To |
|----------------|--------|-----------------------|-----------|
| @RETURN_VALUE  | Return |                       |           |
| @CustomerID    | In     | BERGS                 |           |
|                |        |                       |           |
|                |        |                       |           |

+ Add - Delete ☒ Stored Procedure returns Recordset

**SQL Output**  
☒ Output to File (XML Format)  
c:\temp\BergsOrderHist.xml  
☐ Output to FinalBuilder Log  
Column Delimiter ,  
Null Expression <null>

OK Cancel Help

**Connection String** - specify a connection string to your ADO data source. You can use the built in connection string builder to create and test your connection string.

**Stored Procedure** - once the connection string has been specified, you can click the "Refresh" button and a list of available stored procedures will be listed. Once you select a stored procedure, the parameters will be listed. If the stored procedure is yet to be defined then you can manually enter the stored procedure name.

**Parameters** - The Name and Type of the parameters should retrieved automatically, if not then you need set them to the correct values. For any IN or INOUT parameters you should set a value (either a hardcoded value, or use an FB Variable). For any Return, OUT, or INOUT parameters you can optionally set an FB Variable for the value to be saved to when the stored procedure executes.

Use the **Add** and **Delete** buttons to manually define the parameters if the stored procedure is yet to be defined.

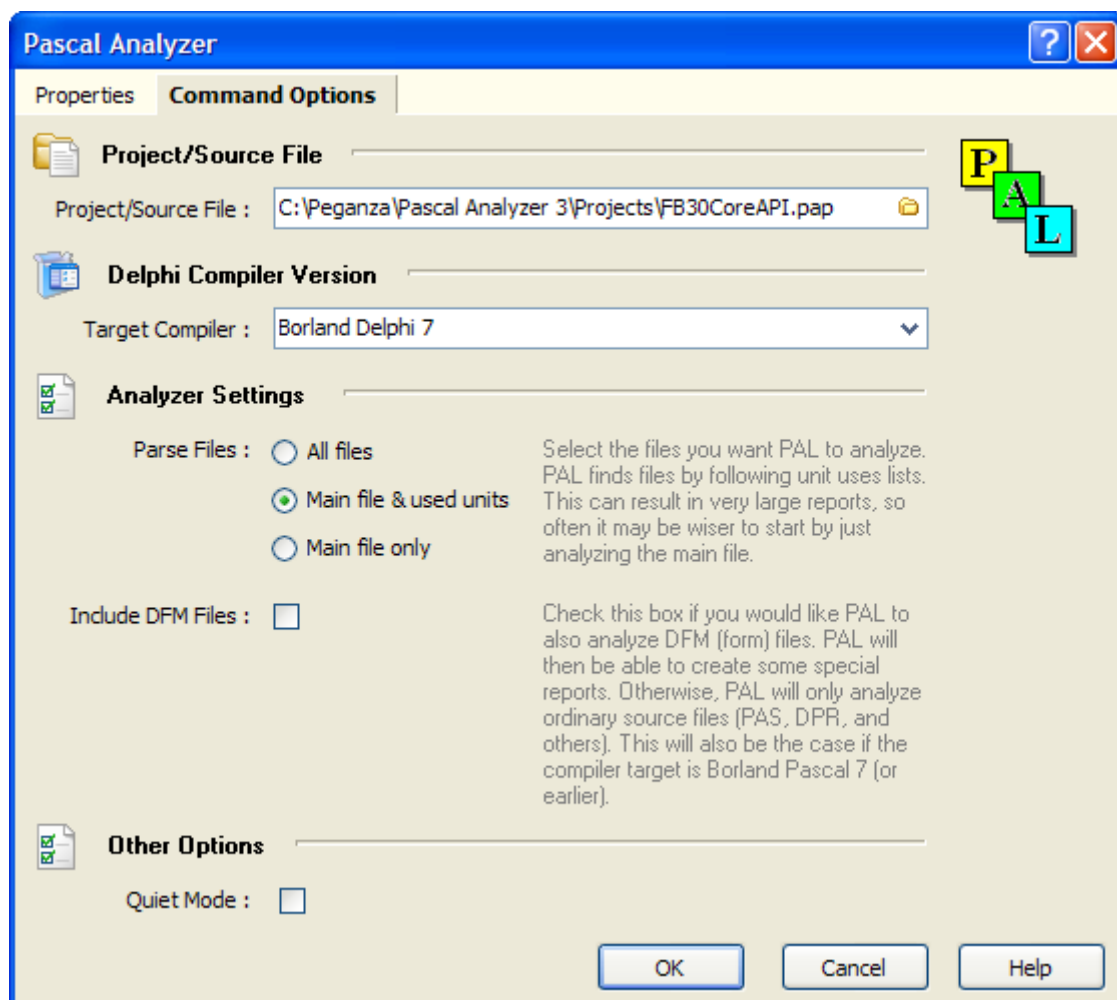
**Stored Procedure returns Recordset** - It is very important that you set the "Stored Procedure returns Recordset" option correctly, as this determines how the stored procedure is called internally.

**SQL Output** - You can output a result recordset to the FB log, and/or to an XML file using the ADO XML file format.

## 5.21 Source Code Tools

### 5.21.1 Pascal Analyzer

The Pascal Analyzer action enables you to analyze your Pascal source code as part of your build process using Peganza Pascal Analyzer.



Select the main file to be analyzed, this can be a Pascal Analyzer project file (.PAP), or Delphi source file (.DPR, .DPK, .PAS), and optionally select the scope of the analysis, and the Borland compiler version you are using.

Make sure that you have configured the location of PALCMD.EXE (the Pascal Analyzer command-line utility), and set other default options via Tools | Options | Source Code Tools | Pascal Analyzer. Other settings such as style and destination of output, and which



reports will be included are read from the Pascal Analyzer configuration file which can be configured via the GUI version of Pascal Analyzer (PAL.EXE).

### **What is Pascal Analyzer**

Pascal Analyzer, or PAL for short, is a utility program that analyzes, documents, debugs, and helps you optimize your source code. It will help you better understand your code and support you in producing code of higher quality, consistency, and reliability. PAL quickly pays itself back in easier maintenance, less errors and improved quality, not only during development, but also throughout the entire life cycle of your code.

For more information see <http://www.peganza.com>

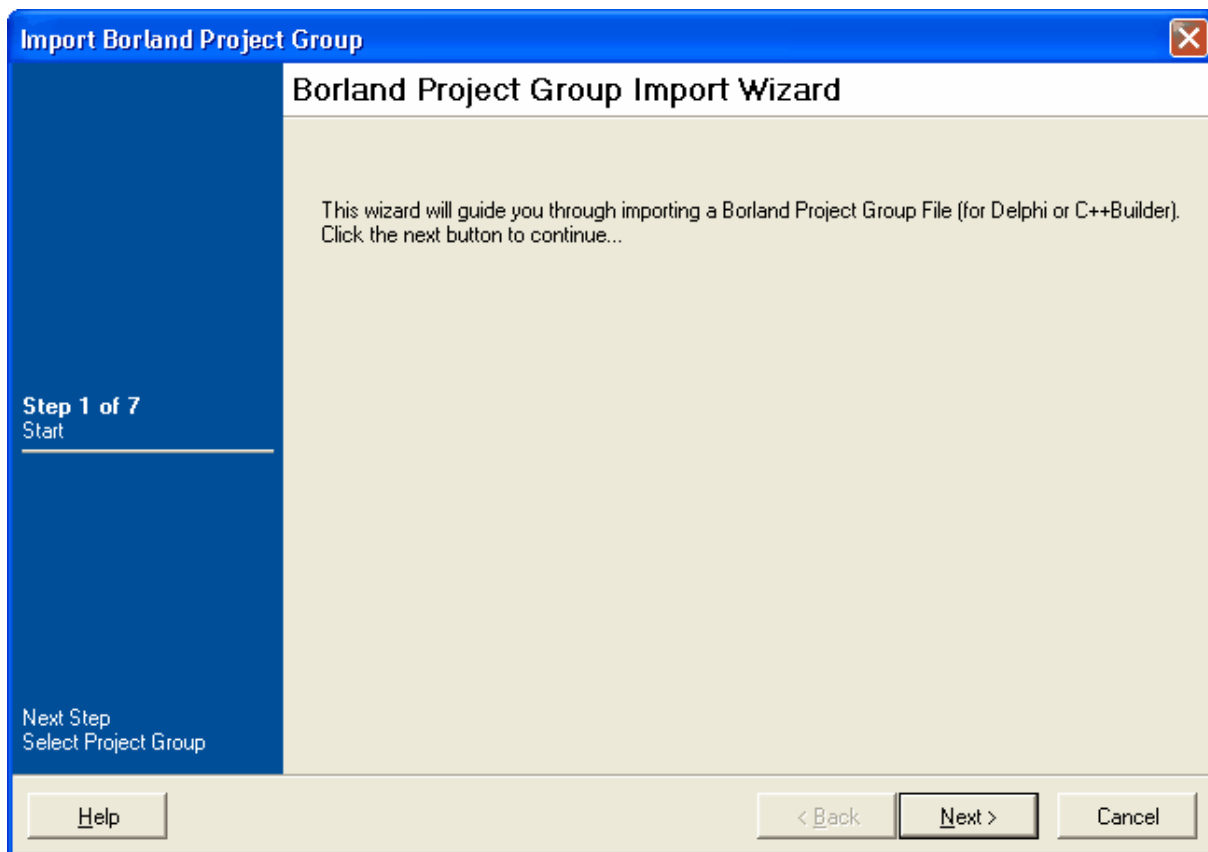
### **How to use PALCMD.EXE**

The FinalBuilder Pascal Analyzer plugin uses the standalone command-line version PALCMD.EXE. PALCMD.EXE uses the same engine as the GUI version PAL.EXE and produces the same output.

## **6 Wizards**

### **6.1 Import Borland Project Wizard**

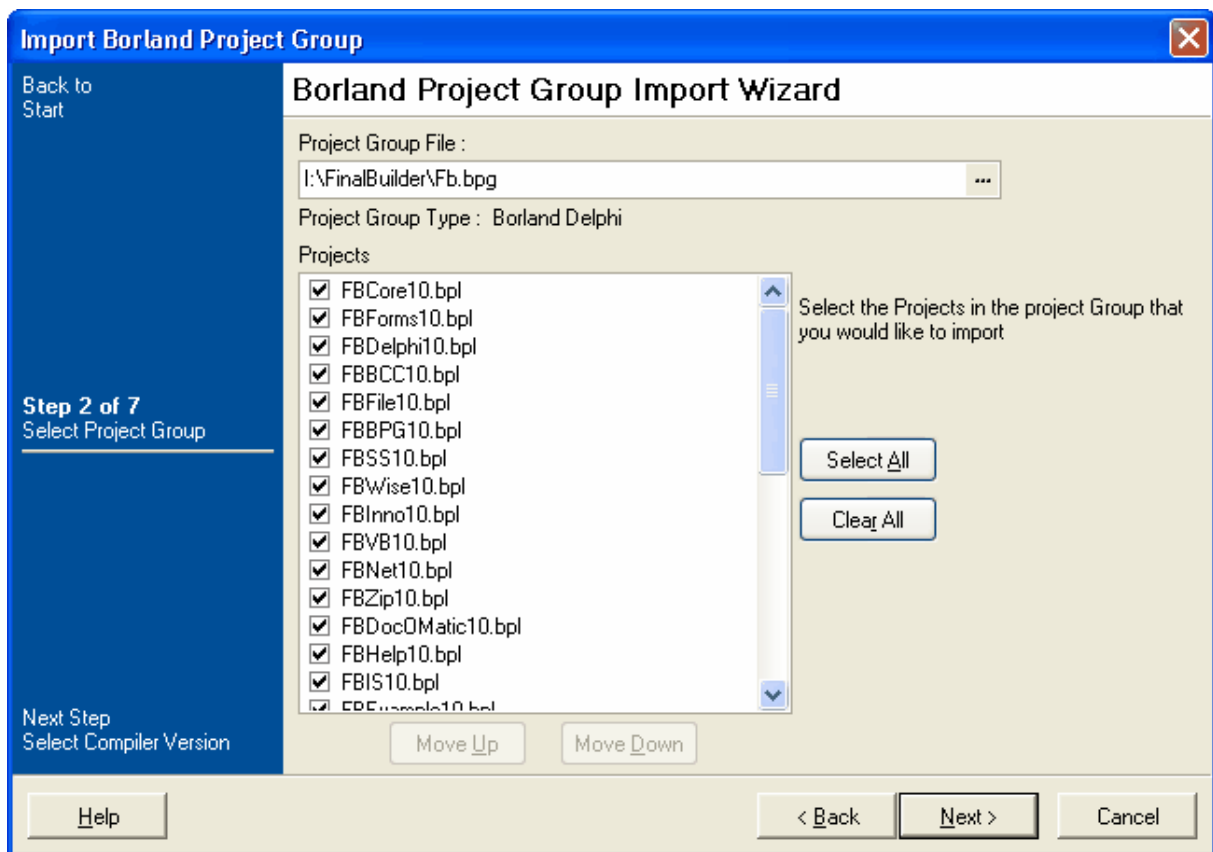
The Import Borland Project Group Wizard imports Delphi project Groups files. Support for BCB project group files will be added in a later version of FinalBuilder.



Next - Select Project Group Page

### 6.1.1 Select Project Group Page

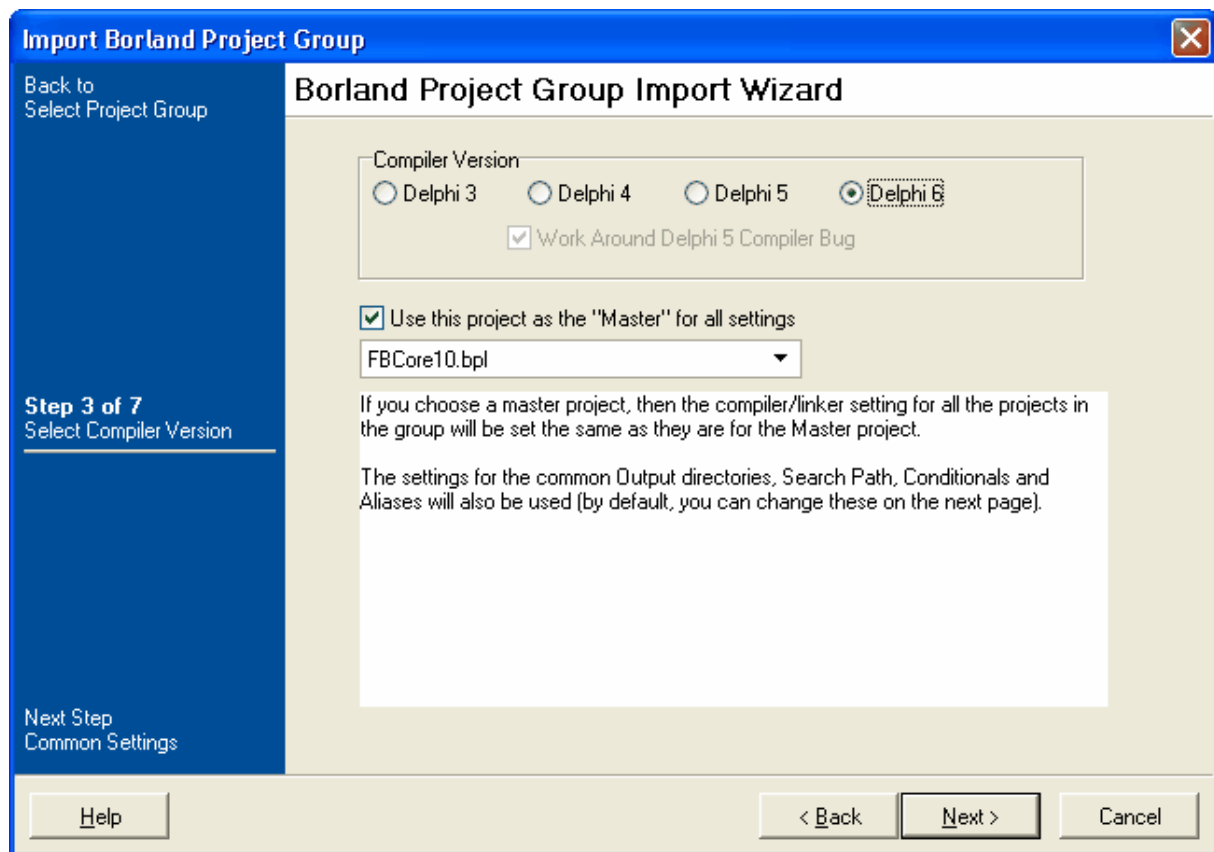
This page allows you to select the Project Group File (.bpg) and select the projects with the project group that you wish to import.



Next - Select Compiler Version Page

### 6.1.2 Select Compiler Version Page

There is no way for FinalBuilder to determine the correct version of Delphi to use for the project, so you must select the version of Delphi you wish to use. You can also select a project to act as the "Master" project, such that the settings from the master project will be applied to all projects. Some of the common settings can be overridden in the next page.



Next - Common Settings Page

### 6.1.3 Common Settings Page

This page allows you to set common settings for Output directories, Library Path, Search Path, Conditionals and Aliases. If you selected a Master project on the previous page, then the default settings will be those of the master project.



**Import Borland Project Group**

Back to Common Settings

**Borland Project Group Import Wizard**

**Step 5 of 7**  
Version Info

Next Step: Import Options

☒ Include Version Info

☒ Link Version Numbers for all projects

| FinalBuilder Project Variables | Initial Values |
|--------------------------------|----------------|
| Major: VERSION_MAJOR           | 1              |
| Minor: VERSION_MINOR           | 2              |
| Release: VERSION_RELEASE       | 9              |
| Build: VERSION_BUILD           | 676            |

☒ Auto Increment Build Number

☒ Apply Common Locale

Locale ID: \$0C09

English (Australia)

Code Page: 1252

☒ Apply Common Product Info

Company Name: VSoft Technologies Pty Ltd

Internal Name: FinalBuilder

Legal Copyright: 2001 - 2002 VSoft Technologies Pty Ltd

Legal Trademarks: FinalBuilder

Product Name: FinalBuilder

Product Version: 1.0.0.0

☒ Include Compiled Date in version info

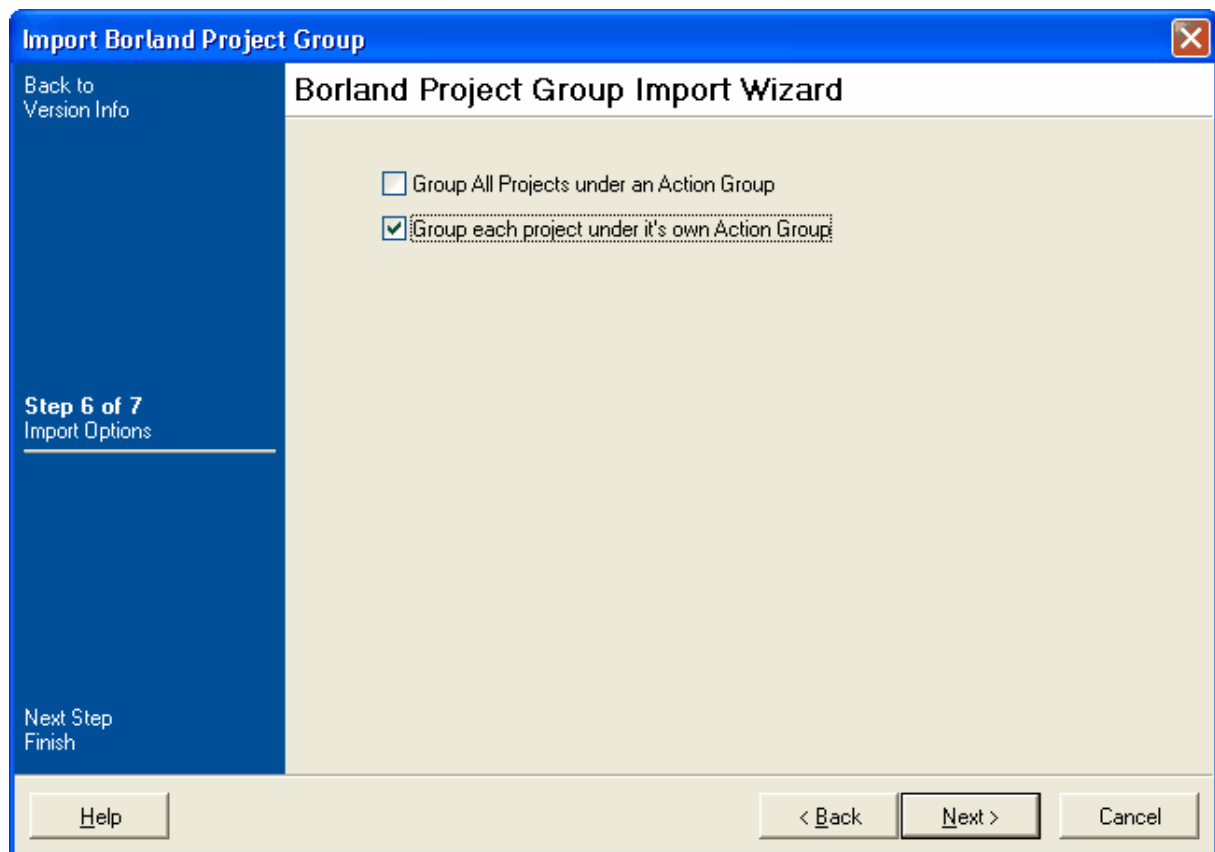
Help < Back Next > Cancel

Next - Import Options Page

### 6.1.5 Import Options Page

**Group All Projects under an Action Group** - This option will make all projects children of an action group. This is useful when the project group is being imported into a FinalBuilder project that already has other Delphi compiler actions.

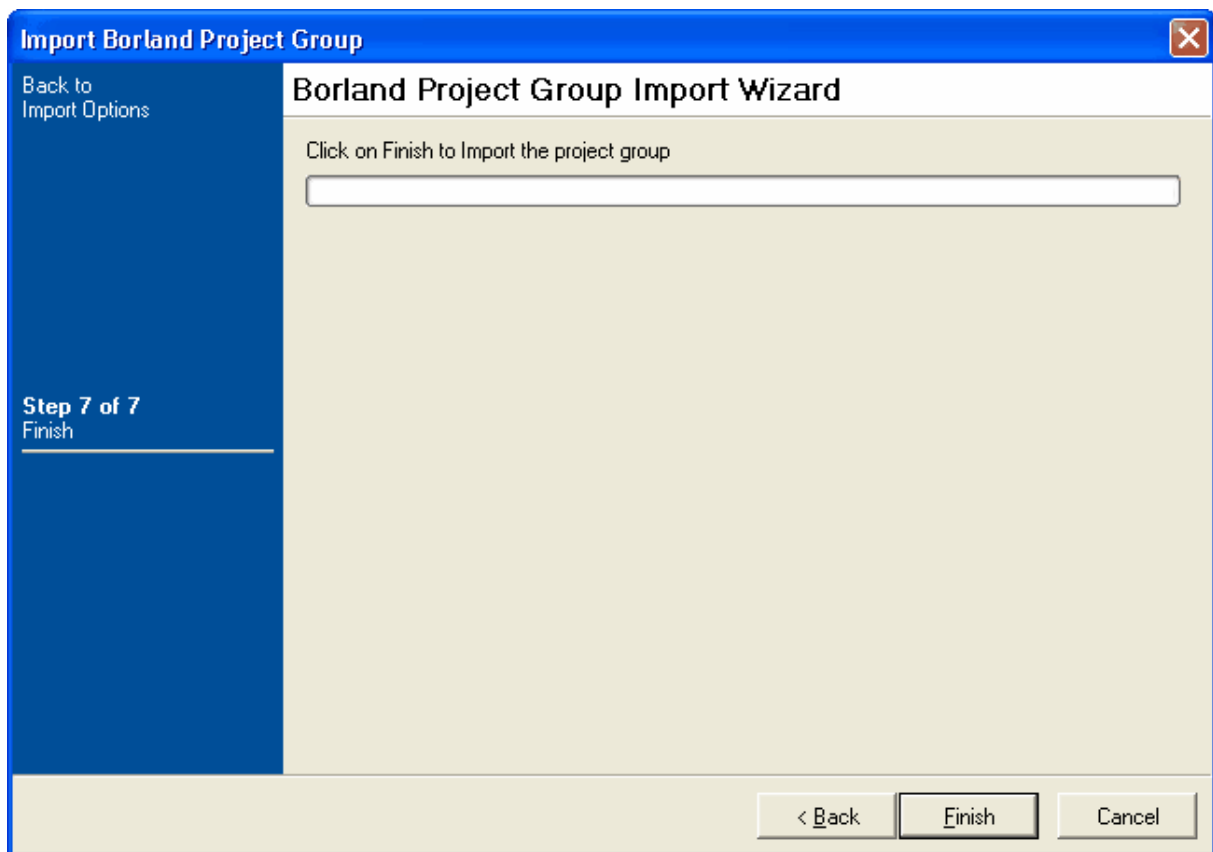
**Group each Project under it's own Action Group** - This option will group each action under an Action Group. This is advisable if you intend to add further action which are specific to each Delphi project.



Next - Finish Page

### 6.1.6 Finish Page

Click on the Finish button to import the project group.



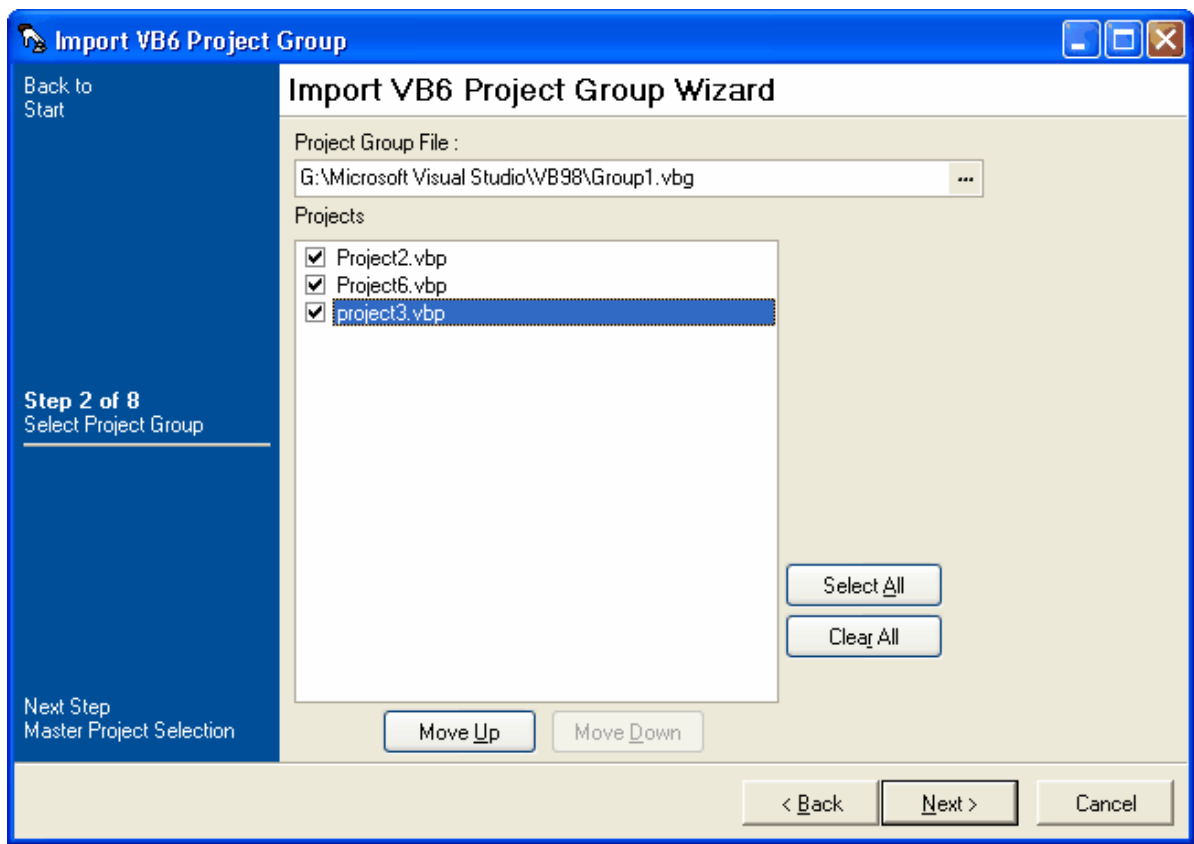
## 6.2 Import VB6 Project Group Wizard

This wizard will guide you through importing a VB6 project group.

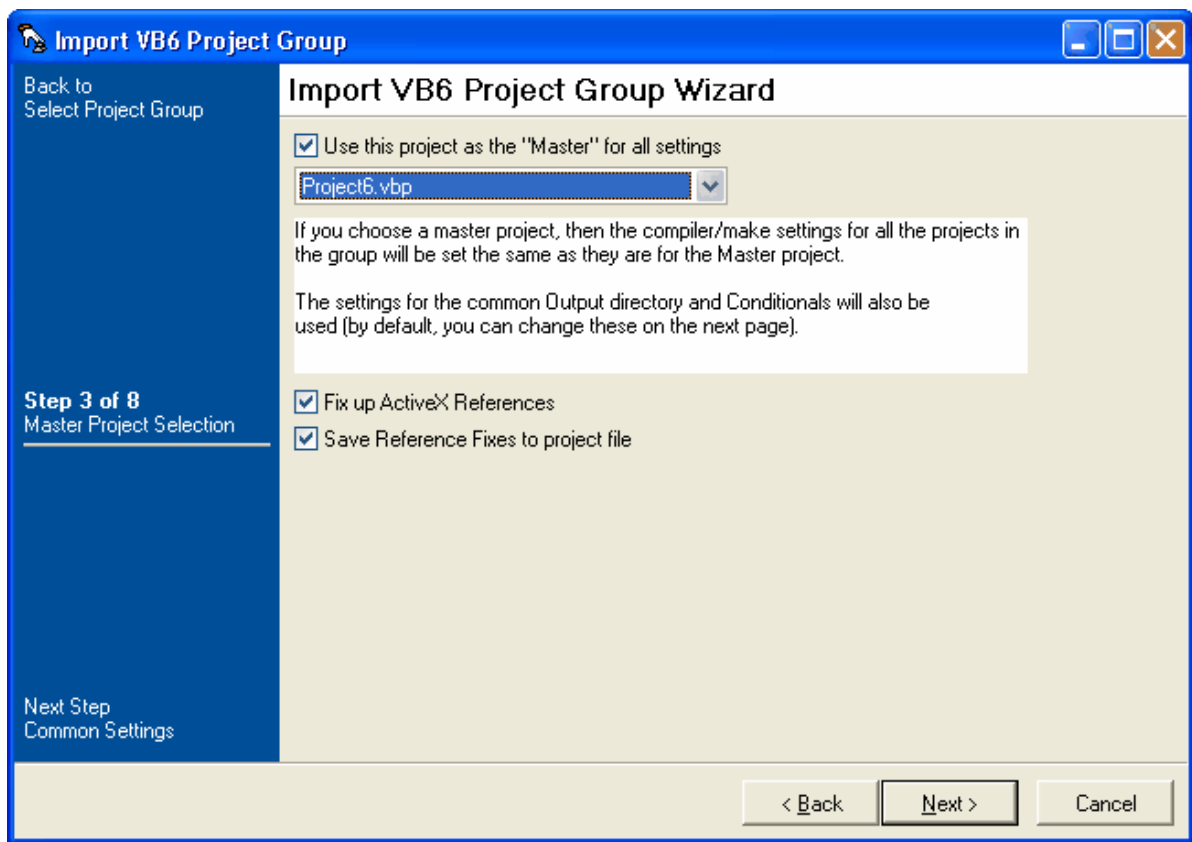
### 6.2.1 Select Project Group

This page allows you to select the Project Group File (.vbg) and select the projects with the project group that you wish to import.

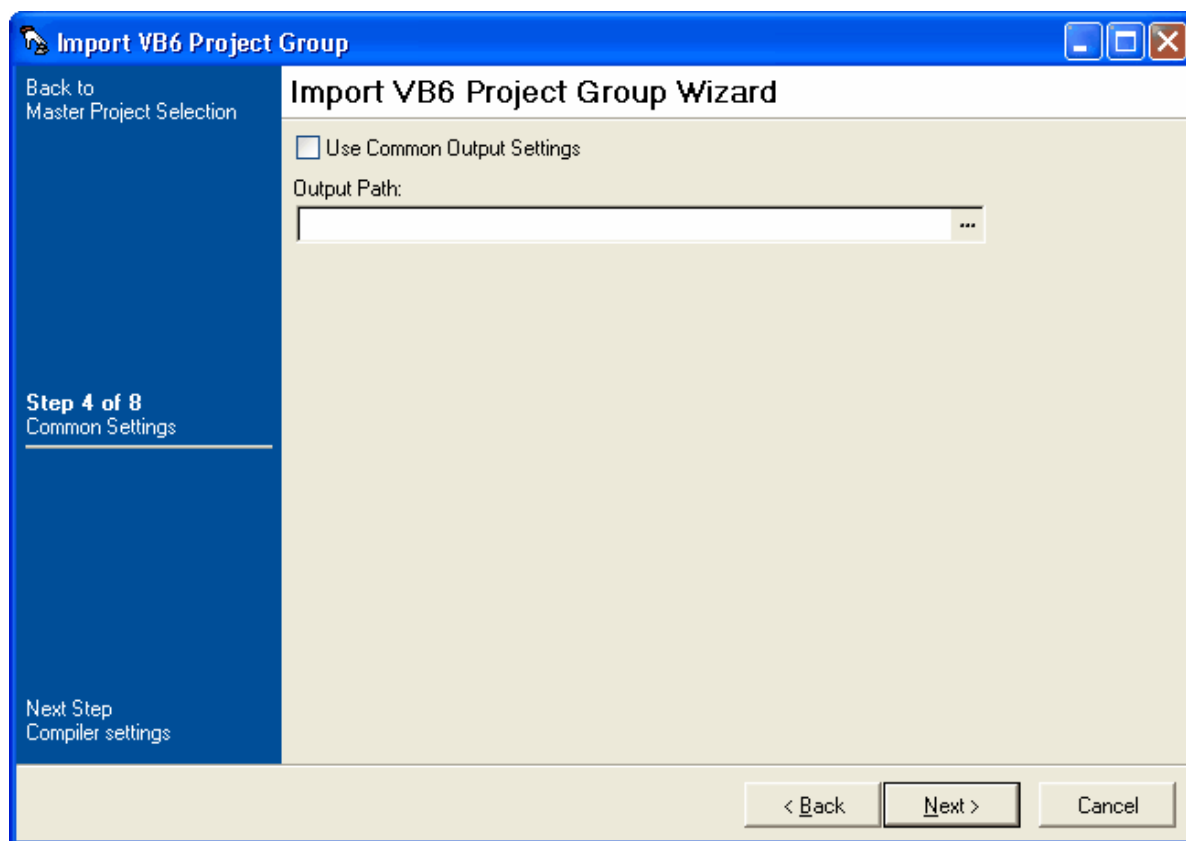




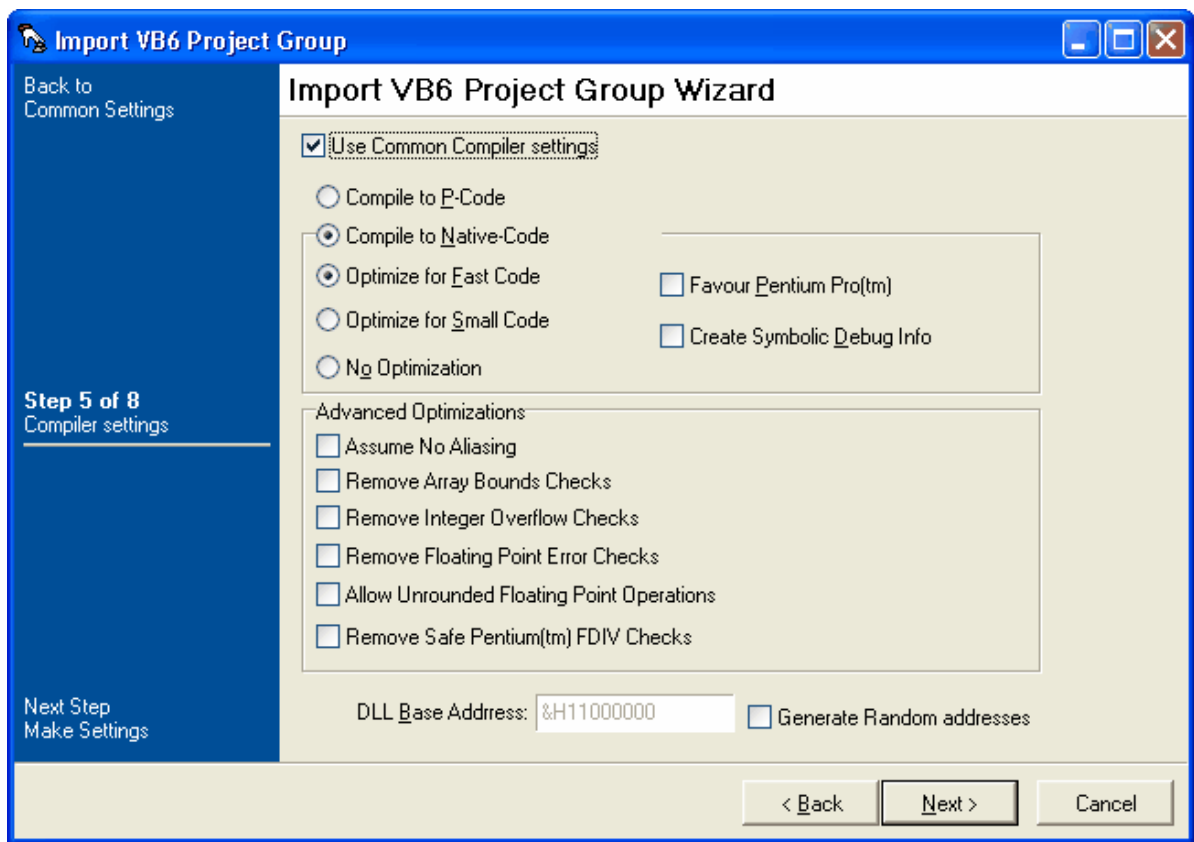
### 6.2.2 Master Project Selection



### 6.2.3 Common Output Path Setting



#### 6.2.4 Compiler Settings



### 6.2.5 Make Settings

**Import VB6 Project Group Wizard**

☒ Use Common Make settings

Version Info

| Key              | Value                      |
|------------------|----------------------------|
| Comments         |                            |
| Company Name     | VSoft Technologies Pty Ltd |
| File Description |                            |
| Legal Copyright  |                            |
| Legal TradeMarks |                            |
| Product Name     |                            |

☒ Include Compile Date in Comment Version Info Field

☒ Link Version Numbers for all projects

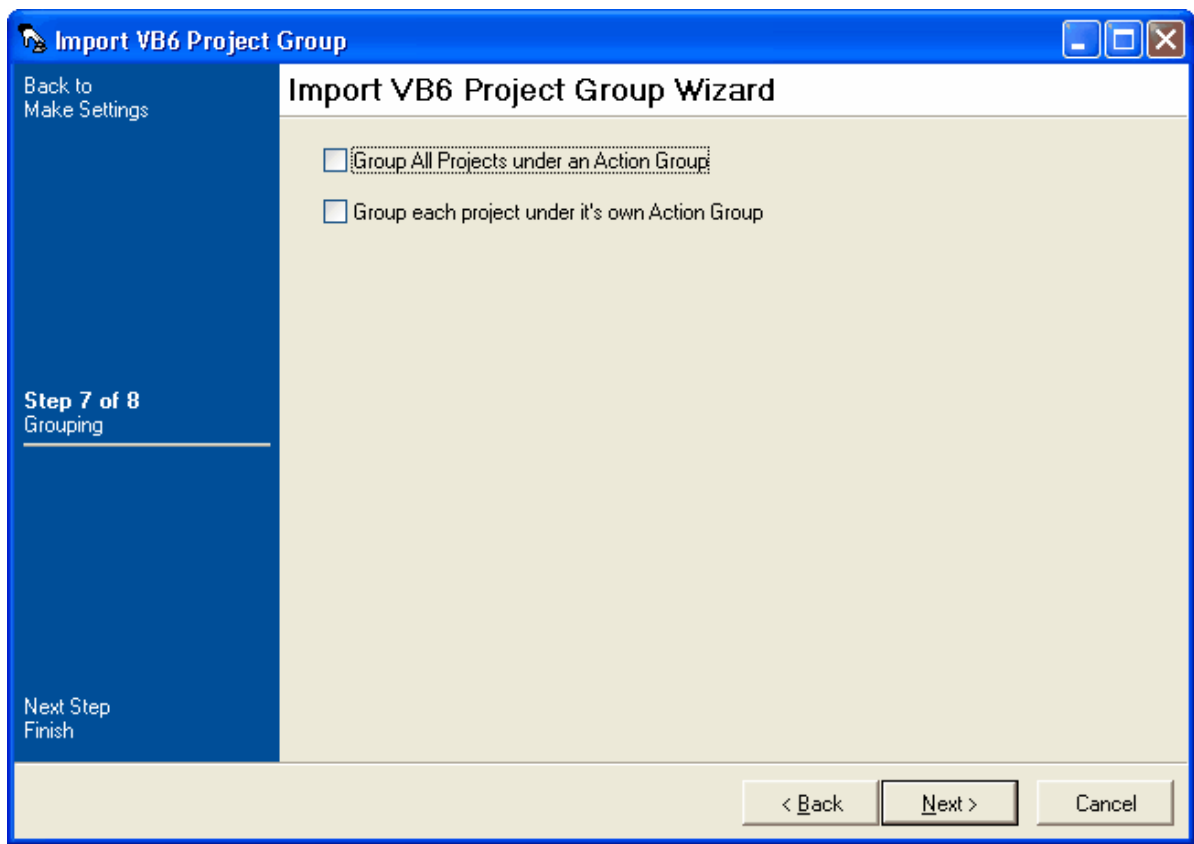
| FinalBuilder Project Variables | Initial Values |
|--------------------------------|----------------|
| Major : VERSION_MAJOR          | 1              |
| Minor : VERSION_MINOR          | 0              |
| Revision : VERSION_REVISION    | 0              |

☐ Auto Increment ☐ Use VBP Version Numbers

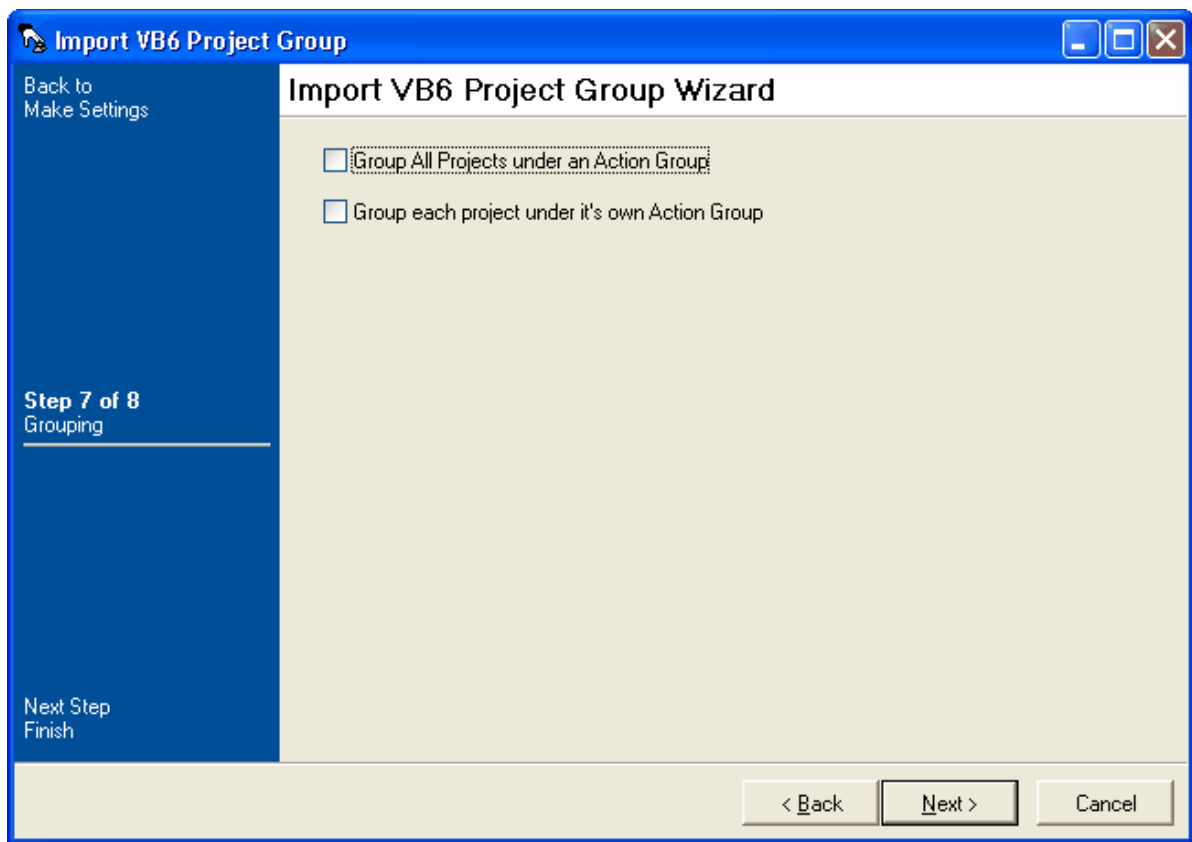
Conditional Compilation Arguments:

< Back   Next >   Cancel

## 6.2.6 Grouping



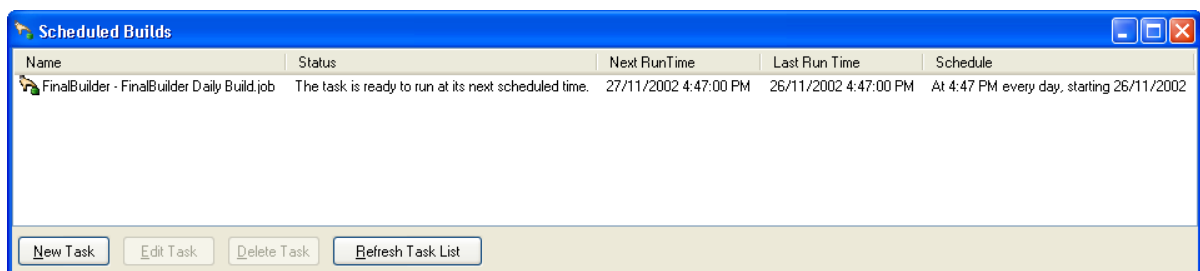
## 6.2.7 Finish



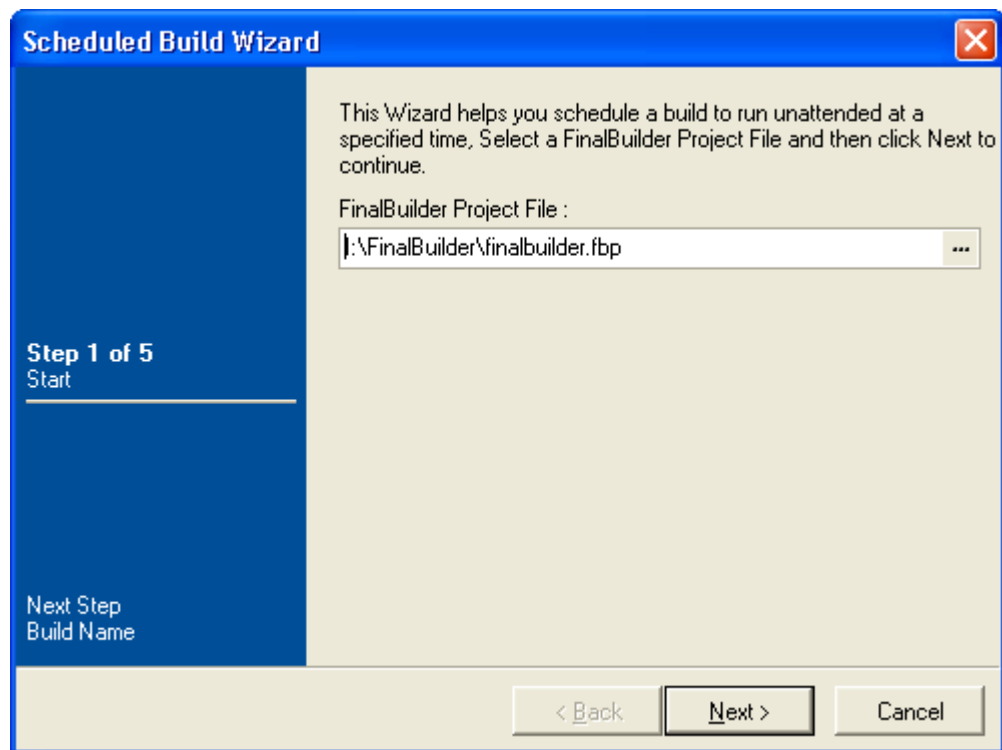
# 7 Automating FinalBuilder

## 7.1 Scheduling Builds

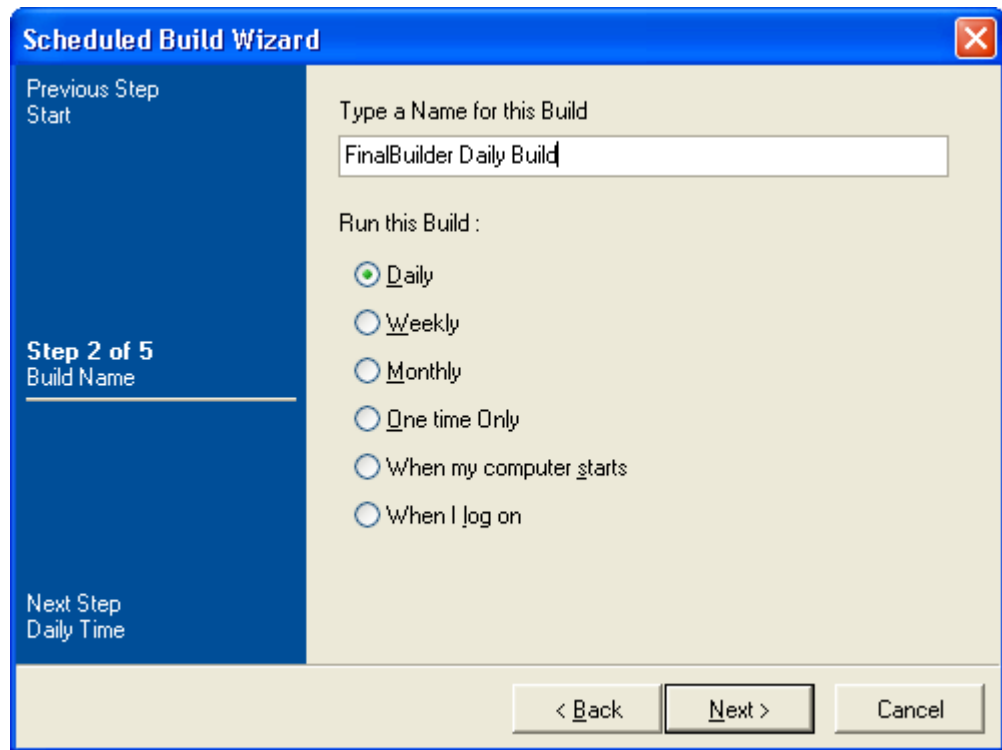
FinalBuilder can schedule builds to be run by the Windows Scheduling service (Win2k & XP). To access the scheduling functions, select the Tools\Scheduled Builds menu



To Create a new New Task, click on the New Task button. This will display the wizard that will step you through scheduling a build.



Select a FinalBuilder Project File and then click Next to continue..



Enter a Title for the scheduled build and then specify the frequency of the build, then click on next to continue.



The screenshot shows the 'Scheduled Build Wizard' dialog box at Step 3 of 5, titled 'Daily Time'. The left sidebar indicates the 'Previous Step' is 'Build Name' and the 'Next Step' is 'User Credentials'. The main area contains the following fields and options:

- Select the time and day you want this build to start.**
- Start Time:** A time picker set to 4:50:57 PM.
- Perform this build:** Three radio button options:
  - ☒ Every Day
  - ☐ Weekdays
  - ☐ Every 1 Days (with a spinner box set to 1)
- Start Date :** A date picker set to 26/11/2002.

At the bottom, there are three buttons: '< Back', 'Next >', and 'Cancel'.

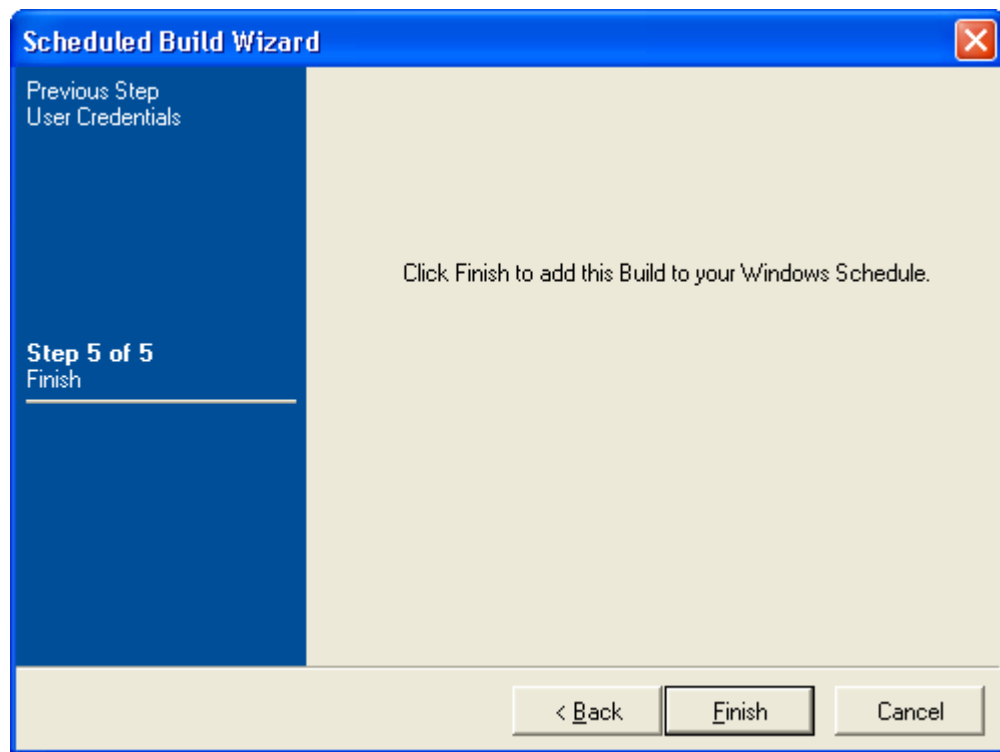
Specify the Time and starting date for the schedule, then click next to continue.

The screenshot shows the 'Scheduled Build Wizard' dialog box at Step 4 of 5, titled 'User Credentials'. The left sidebar indicates the 'Previous Step' is 'Daily Time' and the 'Next Step' is 'Finish'. The main area contains the following fields and text:

- Enter the Name and Password of a User. The build will run as if it were started by that user.**
- Enter the User Name :** A text box containing 'FBGURU\Vincent'.
- Enter the Password :** A text box containing 'xxxxxx'.
- Confirm Password :** A text box containing 'xxxxxx'.

At the bottom, there are three buttons: '< Back', 'Next >', and 'Cancel'.

Provide the user name that the build will run as and the password, then click continue.



Click Finish to add this Build to your Scheduled Tasks.

## 7.2 Command Line Interface

### FinalBuilder Command Line Options

**Usage :** FinalBuilder.exe [switches] projectfile

#### Switches

- n or /n - Hide the splash screen when starting up
- r or /r - auto build the project file passed in on the command line
- e or /e - exit when done building , only valid when autobuild switch is included
- f or /f - don't exit if error occurs , only valid when autobuild switch is included
- v or /v - set variables - for multiple variables, separate the name/value pairs with semi-colons.
- a or /a - allow interactive actions (eg. Prompt For Variable, Ask Question, etc.)
- i or /i - enable live logging
- L or /L - disable writing to the log file

eg. :

```
/vOutputDIR=d:\temp\buildfiles;DCUDIR=d:\temp\dcu
```

Note that if the variable value has spaces in it then enclose the value in double quotes, eg. :

```
/vOutputDIR="d:\temp\build files" .
```

If you do not include the double quotes your build may fail due to incorrect variable values. Separate multiple name=value pairs with semicolons.

Note that variables set from the command line must already be defined as project or user variables. This option is only valid when specified with the /r (auto run).

When running FinalBuilder as a scheduled task, you should always use the /n /r /e switches.

## 7.3 Exit Codes

FinalBuilder uses Exit codes when run from external programs to indicate the outcome of the build :

| Exit Code | Description  |
|-----------|--|
| 0         | The build completed Successfully                               |
| 1         | An Error occurred in the build. Check the log for the details. |
| 20        | Invalid project when Auto build project with "Exit when done"  |

### See Also

Command Line Interface | Scheduling Builds

## 8 Reference

### 8.1 Regular Expression Reference

#### Introduction

Regular Expressions are a widely-used method of specifying patterns of text to search for. Special metacharacters allow you to specify, for instance, that a particular string you are looking for occurs at the beginning or end of a line, or contains n recurrences of a certain character.

Regular expressions look ugly for novices, but really they are very simple (well, usually simple ;) ), handy and powerful tool.

Regular expressions can be used in some actions and can also be used by plugin developers. This reference documents the particular regular expression library used in FinalBuilder: TRegExpr, see <http://regexpstudio.com>

#### Simple matches

Any single character matches itself, unless it is a metacharacter with a special meaning described below.

A series of characters matches that series of characters in the target string, so the pattern "bluh" would match "bluh" in the target string. Quite simple, eh ?

You can cause characters that normally function as metacharacters or escape sequences to be interpreted literally by 'escaping' them by preceding them with a backslash "\", for instance: metacharacter "^" match beginning of string, but "\\^" match character "^", "\\\" match "\" and so on.

Examples:

```
foobar      matchs string 'foobar'
\\^FooBarPtr matchs '^FooBarPtr'
```

### Escape sequences

Characters may be specified using a escape sequences syntax much like that used in C and Perl: "\n" matches a newline, "\t" a tab, etc. More generally, \xnn, where nn is a string of hexadecimal digits, matches the character whose ASCII value is nn. If You need wide (Unicode) character code, You can use '\x{nnnn}', where 'nnnn' - one or more hexadecimal digits.

```
\xnn    char with hex code nn
\x{nnnn} char with hex code nnnn (one byte for plain text and two bytes for Unicode)
\t      tab (HT/TAB), same as \x09
\n      newline (NL), same as \x0a
\r      car.return (CR), same as \x0d
\f      form feed (FF), same as \x0c
\a      alarm (bell) (BEL), same as \x07
\e      escape (ESC), same as \x1b
```

Examples:

```
foo\x20bar  matchs 'foo bar' (note space in the middle)
\tfoobar    matchs 'foobar' predefined by tab
```

### Character classes

You can specify a character class, by enclosing a list of characters in [], which will match any one character from the list.

If the first character after the "[" is "^", the class matches any character not in the list.

Examples:

```
foob[aeiou]r  finds strings 'foobar', 'foober' etc. but not 'foobbr', 'foobcr' etc.

foob[^aeiou]r find strings 'foobbr', 'foobcr' etc. but not 'foobar', 'foober' etc.
```

Within a list, the "-" character is used to specify a range, so that a-z represents all characters between "a" and "z", inclusive.

If You want "-" itself to be a member of a class, put it at the start or end of the list, or escape it with a backslash. If You want ']' you may place it at the start of list or escape it

with a backslash.

Examples:

```
[-az]    matchs 'a', 'z' and '-'  
  
[az-]    matchs 'a', 'z' and '-'  
[a\z-]   matchs 'a', 'z' and '-'  
[a-z]    matchs all twenty six small characters from 'a' to 'z'  
[\n-\x0D] matchs any of #10,#11,#12,#13.  
[\d-t]   matchs any digit, '-' or 't'.  
[]-a]    matchs any char from ']'..'a'.
```

## Metacharacters

Metacharacters are special characters which are the essence of Regular Expressions. There are different types of metacharacters, described below.

Metacharacters - line separators

```
^    start of line  
$    end of line  
\A  start of text  
\Z  end of text  
.    any character in line
```

Examples:

```
^foobar  matchs string 'foobar' only if it's at the beginning of line  
foobar$  matchs string 'foobar' only if it's at the end of line  
^foobar$  matchs string 'foobar' only if it's the only string in line  
  
foob.r   matchs strings like 'foobar', 'foobbr', 'foob1r' and so on
```

The "^" metacharacter by default is only guaranteed to match at the beginning of the input string/text, the "\$" metacharacter only at the end. Embedded line separators will not be matched by "^" or "\$".

You may, however, wish to treat a string as a multi-line buffer, such that the "^" will match after any line separator within the string, and "\$" will match before any line separator. You can do this by switching On the modifier /m.

The \A and \Z are just like "^" and "\$", except that they won't match multiple times when the modifier /m is used, while "^" and "\$" will match at every internal line separator.

The "." metacharacter by default matches any character, but if You switch Off the modifier /s, then '.' won't match embedded line separators.

TRegExpr works with line separators as recommended at [www.unicode.org](http://www.unicode.org) (<http://www.unicode.org/unicode/reports/tr18/> ):

"^" is at the beginning of a input string, and, if modifier /m is On, also immediately following any occurrence of \x0D\x0A or \x0A or \x0D (if You are using Unicode version of TRegExpr, then also \x2028 or \x2029 or \x0B or \x0C or \x85). Note that there is no

empty line within the sequence `\x0D\x0A`.

"\$" is at the end of a input string, and, if modifier /m is On, also immediately preceding any occurrence of `\x0D\x0A` or `\x0A` or `\x0D` (if You are using Unicode version of TRegExpr, then also `\x2028` or `\x2029` or `\x0B` or `\x0C` or `\x85`). Note that there is no empty line within the sequence `\x0D\x0A`.

"." matchs any character, but if You switch Off modifier /s then "." doesn't match `\x0D\x0A` and `\x0A` and `\x0D` (if You are using Unicode version of TRegExpr, then also `\x2028` and `\x2029` and `\x0B` and `\x0C` and `\x85`).

Note that `^.*$` (an empty line pattern) doesnot match the empty string within the sequence `\x0D\x0A`, but matchs the empty string within the sequence `\x0A\x0D`.

Multiline processing can be easely tuned for Your own purpose with help of TRegExpr properties `LineSeparators` and `LinePairedSeparator`, You can use only Unix style separators `\n` or only DOS/Windows style `\r\n` or mix them together (as described above and used by default) or define Your own line separators!

### Metacharacters - predefined classes

|                 |   |
|-----------------|---|
| <code>\w</code> | an alphanumeric character (including "_")     |
| <code>\W</code> | a nonalphanumeric                             |
| <code>\d</code> | a numeric character                           |
| <code>\D</code> | a non-numeric                                 |
| <code>\s</code> | any space (same as [ <code>\t\n\r\f</code> ]) |
| <code>\S</code> | a non space                                   |

You may use `\w`, `\d` and `\s` within custom character classes.

Examples:

`foob\dr` matchs strings like 'foob1r', 'foob6r' and so on but not 'foobar', 'foobbr' and so on

`foob[\w\s]r` matchs strings like 'foobar', 'foob r', 'foobbr' and so on but not 'foob1r', 'foob=r' and so on

TRegExpr uses properties `SpaceChars` and `WordChars` to define character classes `\w`, `\W`, `\s`, `\S`, so You can easely redefine it.

### Metacharacters - word boundaries

|                 |                             |
|-----------------|-----------------------------|
| <code>\b</code> | Match a word boundary       |
| <code>\B</code> | Match a non-(word boundary) |

A word boundary (`\b`) is a spot between two characters that has a `\w` on one side of it and a `\W` on the other side of it (in either order), counting the imaginary characters off the beginning and end of the string as matching a `\W`.

### Metacharacters - iterators

Any item of a regular expression may be followed by another type of metacharacters -

iterators. Using this metacharacters You can specify number of occurrences of previous character, metacharacter or subexpression.

```
*    zero or more ("greedy"), similar to {0,}
+    one or more ("greedy"), similar to {1,}
?    zero or one ("greedy"), similar to {0,1}

{n}   exactly n times ("greedy")
{n,}  at least n times ("greedy")
{n,m} at least n but not more than m times ("greedy")
*?    zero or more ("non-greedy"), similar to {0,}?
+?    one or more ("non-greedy"), similar to {1,}?
??    zero or one ("non-greedy"), similar to {0,1}?
{n}?  exactly n times ("non-greedy")
{n,}? at least n times ("non-greedy")
{n,m}? at least n but not more than m times ("non-greedy")
```

So, digits in curly brackets of the form `{n,m}`, specify the minimum number of times to match the item `n` and the maximum `m`. The form `{n}` is equivalent to `{n,n}` and matches exactly `n` times. The form `{n,}` matches `n` or more times. There is no limit to the size of `n` or `m`, but large numbers will chew up more memory and slow down r.e. execution.

If a curly bracket occurs in any other context, it is treated as a regular character.

Examples:

```
foob.*r    matchs strings like 'foobar', 'foobalkjdfk9r' and 'foobr'

foob.+r    matchs strings like 'foobar', 'foobalkjdfk9r' but not 'foobr'
foob.?r    matchs strings like 'foobar', 'foobbr' and 'foobr' but not 'foobalkj9r'
fooba{2}r  matchs the string 'foobaar'
fooba{2,}r matchs strings like 'foobaar', 'foobaaar', 'foobaaaar' etc.
fooba{2,3}r matchs strings like 'foobaar', or 'foobaaar' but not 'foobaaaar'
```

A little explanation about "greediness". "Greedy" takes as many as possible, "non-greedy" takes as few as possible. For example, `'b+'` and `'b*'` applied to string `'abbbbc'` return `'bbbb'`, `'b+?'` returns `'b'`, `'b*?'` returns empty string, `'b{2,3}?'` returns `'bb'`, `'b{2,3}'` returns `'bbb'`.

You can switch all iterators into "non-greedy" mode (see the modifier `/g`).

## Metacharacters - alternatives

You can specify a series of alternatives for a pattern using `|` to separate them, so that `fee|fie|foe` will match any of `"fee"`, `"fie"`, or `"foe"` in the target string (as would `f(e|i|o)e`). The first alternative includes everything from the last pattern delimiter (`"(`, `"["`, or the beginning of the pattern) up to the first `|`, and the last alternative contains everything from the last `|` to the next pattern delimiter. For this reason, it's common practice to include alternatives in parentheses, to minimize confusion about where they start and end.

Alternatives are tried from left to right, so the first alternative found for which the entire expression matches, is the one that is chosen. This means that alternatives are not

necessarily greedy. For example: when matching `foo|foot` against `"barefoot"`, only the `"foo"` part will match, as that is the first alternative tried, and it successfully matches the target string. (This might not seem important, but it is important when you are capturing matched text using parentheses.)

Also remember that `"|"` is interpreted as a literal within square brackets, so if You write `[fee|fie|foe]` You're really only matching `[feio|]`.

Examples:

`foo(bar|foo)` matchs strings `'foobar'` or `'foofoo'`.

### Metacharacters - subexpressions

The bracketing construct `( ... )` may also be used for define r.e. subexpressions (after parsing You can find subexpression positions, lengths and actual values in `MatchPos`, `MatchLen` and `Match` properties of `TRegExpr`, and substitute it in template strings by `TRegExpr.Substitute`).

Subexpressions are numbered based on the left to right order of their opening parenthesis.

First subexpression has number `'1'` (whole r.e. match has number `'0'` - You can substitute it in `TRegExpr.Substitute` as `'$0'` or `'$&'`).

Examples:

`(foobar){8,10}` matchs strings which contain 8, 9 or 10 instances of the `'foobar'`  
`foob([0-9]|a+)r` matchs `'foob0r'`, `'foob1r'`, `'foobar'`, `'foobaar'`, `'foobaar'` etc.

### Metacharacters - backreferences

Metacharacters `\1` through `\9` are interpreted as backreferences. `\<n>` matches previously matched subexpression `#<n>`.

Examples:

`(.)\1+` matchs `'aaaa'` and `'cc'`.

`(.)\1+` also match `'abab'` and `'123123'`

`(["']?)(\d+)\1` matchs `'"13"` (in double quotes), or `'4'` (in single quotes) or `77` (without quotes) etc

### Modifiers

Modifiers are for changing behaviour of `TRegExpr`.

There are many ways to set up modifiers.

Any of these modifiers may be embedded within the regular expression itself using the `(?...)` construct.

Also, You can assign to appropriate `TRegExpr` properties (`ModifierX` for example to change `/x`, or `ModifierStr` to change all modifiers together). The default values for new instances of `TRegExpr` object defined in global variables, for example global variable `RegExprModifierX` defines value of new `TRegExpr` instance `ModifierX` property.



i

Do case-insensitive pattern matching (using installed in you system locale settings), see also InvertCase.

m

Treat string as multiple lines. That is, change "^" and "\$" from matching at only the very start or end of the string to the start or end of any line anywhere within the string, see also Line separators.

s

Treat string as single line. That is, change "." to match any character whatsoever, even a line separators (see also Line separators), which it normally would not match.

g

Non standard modifier. Switching it Off You'll switch all following operators into non-greedy mode (by default this modifier is On). So, if modifier /g is Off then '+' works as '+?', '\*' as '\*?' and so on

x

Extend your pattern's legibility by permitting whitespace and comments (see explanation below).

r

Non-standard modifier. If is set then range à-ÿ additional include russian letter ' ', À-ß additional include ' ', and à-ß include all russian symbols.

Sorry for foreign users, but it's set by default. If you want switch if off by default - set false to global variable RegExprModifierR.

The modifier /x itself needs a little more explanation. It tells the TRegExpr to ignore whitespace that is neither backslashed nor within a character class. You can use this to break up your regular expression into (slightly) more readable parts. The # character is also treated as a metacharacter introducing a comment, for example:

```
(
(abc) # comment 1
| # You can use spaces to format r.e. - TRegExpr ignores it
(efg) # comment 2
)
```

This also means that if you want real whitespace or # characters in the pattern (outside a character class, where they are unaffected by /x), that you'll either have to escape them or encode them using octal or hex escapes. Taken together, these features go a long way towards making regular expressions text more readable.

## Perl extensions

(?imsxr-imsxr)

You may use it into r.e. for modifying modifiers by the fly. If this construction inlined into subexpression, then it effects only into this subexpression

Examples:

(?i)Saint-Petersburg      matchs 'Saint-petersburg' and 'Saint-Petersburg'

(?i)Saint-(?-i)Petersburg      matchs 'Saint-Petersburg' but not 'Saint-petersburg'

(?i)(Saint-)?Petersburg      matchs 'Saint-petersburg' and 'saint-petersburg'

((?i)Saint-)?Petersburg      matchs 'saint-Petersburg', but not 'saint-petersburg'

(?#text)

A comment, the text is ignored. Note that TRegExpr closes the comment as soon as it sees a ")", so there is no way to put a literal ")" in the comment.

## 9 Support

### 9.1 Known Problems

#### Borland Delphi Compiler Action

**Problem:** Compiling Projects with UNC Paths in the project file property will result in the resource file not being linked into the executable. This is appears to be a bug in the Delphi command line compiler (not yet confirmed).

**Workaround :** Build locally, not on a network drive.

#### Wise InstallMaster/InstallBuilder Actions

**Problem:** The Wise compiler (versions earlier than 9.02) do not return a non zero return code when an error occurs during compilation. This means that FinalBuilder has no way of detecting if the compile of the installer failed or not. Usually Wise displays a message box when an error occurred.

**Workaround :** Stop the build manually after the message box is displayed.

**Problem:** The Wise compiler will hang if you attempt to run it as part of a scheduled build. This is a known problem with wise, not with FinalBuilder! The issue is that the Wise Compiler will display a messagebox or dialog and of course there is no way to respond to it while it is being run unattended.

**Workaround :** None known. Other than to use a different installer, one that fully supports automated builds such as Inno Setup or InstallShield. This problem may be solved with Wise 9.02, however we have not confirmed this.

**Workaround :** Use VBScript or JavaScript.

#### Borland C++ Builder Action

**Problem:** Some static Lib projects will not link correctly when compiled from FinalBuilder. You might see something like this :

```
J:\Borland\CBUILD~1\bin\..\BIN\TLIB /u debug\jpegD.lib @MAKE0000.@@@
DOS-reported error: Bad file number
TLIB 4.5 Copyright (c) 1987, 1999 Inprise Corporation
opening 'MAKE0000.@@@'
```

```
** error 1 ** deleting debug\jpegD.lib
MAKE failed, returned : 1
```

**Workaround :** In some cases (where the "Bad file number" error is seen) it may be possible to work around this by specifying -tDEFLIB.BMK in the BPR2MAKE Options field, and Turning off the "Capture Make Output" option.

**Problem:** Some projects (especially those migrated from an earlier version of BCB) may not compile in FinalBuilder, even though they compile fine in the BCB IDE. Typical symptoms include Unresolved External symbols errors.

**Workaround :** This work around works in some cases, where the package/libraries/sparelibs options in the bpr/bpk file are not correct. Open the project in the BCB IDE, and change the Use Packages option and save the project. Then change the Use Packages option back to what it was before, and save the project. Close BCB and attempt to build the project from FinalBuilder

**Problem:** Make fails with the following error :

```
Error E2266: No file names given
MAKE failed, returned : 1
```

The most common cause of this a space in the intermediate output path. Make does not like this, even the mak files generated from the IDE will fail to build correctly.

**Workaround :** The only way around this is to not use intermediate output paths with spaces in them.

## 9.2 FAQ

**Q. I'm getting this error when running my Delphi action : Error expanding variables in xxx : Variable : DELPHI - does not exist!**

**A.** You are probably using the DELPHI variable outside of the Library Path or SearchPath settings. For example if you have your library path set to %MYLIBRARYPATH% and the default value of %MYLIBRARYPATH% is \$(DELPHI)\lib, then you will see this error. The DELPHI variable is a special variable, at runtime the Delphi action replaces the variable with a compiler version specific variable before the variable expansion is called (eg. DELPHI5DIR for Delphi 5). If you look in the variables dialog you will not find a DELPHI variable declared.

**Q. Why is no executable is being produced when compiling my Delphi 5 project with FinalBuilder**

**A.** This is a bug with the Delphi 5 command line compiler (dcc32.exe), it is fixed in Delphi 6 & 7. To work around this problem, select the "Work around Delphi 5 Compiler bug Check box".

#### **Q. Personal Firewalls & FinalBuilder**

I run ZoneAlarm Pro as a firewall, and every time I update FinalBuilder, it warns that (the changed) FinalBuilder is attempting to access the internet, and that it is attempting to obtain server access (i.e., it listens for connections).

**A.** FinalBuilder checks & listens for other copies of FinalBuilder running on the local network with the same license key. If it detects another copy, and for example the license key is a single user license key, then the second copy will fail to start. If you have a 2 user license key then the third copy will fail to start...

## **9.3 FinalBuilder Support**

VSoft Technologies provide support for FinalBuilder™ on our newsgroup and via email.

News Server: <news://news.finalbuilder.com>

News Server Web interface: <http://news.finalbuilder.com>

Support Email: [support@finalbuilder.com](mailto:support@finalbuilder.com)

# Index

## - . -

- .Net 3rd Party Tools 337
  - Demeanor Action 337
  - Dotfuscator Action 337
  - FxCop Action 337
  - XenoCode Action 338
- .Net Framework Tools 322
  - Register Assembly in COM 322
  - Run AL.EXE Action 321
  - Run ASPNET\_REGIIS.EXE Action 323
- .Net SDK Tools 327
  - Extract Public Key Action 327
  - GAC Download Cache Action 332
  - GAC Install Action 330
  - GAC Uninstall Action 331
  - Generate Key Pair Action 324
  - Install Key in Container Action 326
  - Re-sign Assembly Action 328
  - Run ResGen.exe Action 335
  - Run SN.EXE Action 329
  - Type Library Export Action 334
  - Type Library Import Action 333
  - Verify Strong Name Action 325

## - A -

- Action 21
- Action Group Action 179
- Action List 21
  - Exit Action List 56
  - Run Action List 55
- Action List Parameters 47
- Action Lists 17, 19
- Action Types 17, 18
- Actions 17, 21
- Active Script 32
- ActiveScript 15
- Add Action List 19
- Adding & Editing Variables 44
- ADO Execute SQL Action 355
- ADO Execute Stored Procedure Action 356
- Ant Project Action 345

- AQTest Action 309
- Archiving 303
  - 7Zip 303
  - Create Archive 304
  - Create Zip File Action 299
  - Delete from Archive 309
  - Extract Archive 307
  - Extract Zip File Action 302
  - List Archive 309
  - Test Archive 307
  - Update Archive 308
  - WinRAR Action 302
- Armadillo 314
- Ask Question Action 175
- ASProtect 312
- Assembly Info Updater Action 257
- Authenticode Action 203
- AutomatedQA Test Complete Actions 311
- Automating FinalBuilder 376
  - Command Line Interface 376
  - Exit Codes 377
  - Scheduling Builds 373

## - B -

- Backup Database Action 352
- Beep Action 181
- Borland C# Builder Compiler Action 256
- Borland C++ Builder Action 249
- Borland Java Compiler 228
- Borland Project Group 359
- Borland Resource Compiler Action 247
- Build History 26
- Build Log 17, 23
- Build Tools 345
  - Ant Project Action 345
  - MSBuild Project Action 347
  - NAnt Project Action 346
- Burn CD Action 315
- Burn ISO Action 318

## - C -

- C# Builder Compiler Action 256
- C# Compiler Action 238
- C# Project Compiler Action 239
- C++ Builder Compiler Action 249

Calculate File CRC32 Action 209  
 Calculate File MD5 Action 210  
 Case Action 59  
 Case Statement 59  
     Case Action 59  
     Switch Action 59  
 Catch Action 60  
 Category 18  
 CD/DVD Burning 315  
     Burn CD Action 315  
     Burn ISO Action 318  
     Check Ready Action 319  
     Create ISO Action 317  
     Erase CDRW Action 320  
 Check Catalogue Action 354  
 Check Database Action 354  
 Check File Exists Action 190  
 Check for updates 25, 31  
 Check Ready Action 319  
 Chrome Compiler Action 259  
 CityDesk Action 186  
 Clear Build History 26  
 Command line Interface 376  
 Comment Action 185  
 Compile Borland Resource Script Action 247  
 Compile Delphi 8 for .Net Project Action 257  
 Compile Delphi Project Action 244  
 Compile Visual Basic Project 228  
 Compilers 244, 257  
     Borland C# Compiler Action 256  
     Borland C++ Builder 249  
     Borland Delphi 244  
     Borland Delphi 8 for .Net 257  
     Borland Resource 247  
     Chrome Action 259  
     IncrediBuild Action 260  
     Java 226  
     Microsoft C# Compiler Action 238  
     Microsoft C# Project Compiler Action 239  
     Microsoft J# Compiler Action 242  
     Microsoft J# Project Compiler Action 243  
     Microsoft VB.NET Compiler Action 240  
     Microsoft VB.NET Project Compiler Action 241  
     Visual Basic 228  
     Visual C++ 6 233  
     Visual Studio .NET 236  
 Concatenate Files Action 201  
 Control Service Action 218

Copy Files Action 195  
 Copy/Move File List Action 199  
 Copying Actions 21  
 Create Archive 304  
 Create Directory Action 192  
 Create ISO Action 317  
 Create Text File Action 204  
 Create Zip File Action 299  
 CVS Actions 103  
 CVS Command 105

## - D -

Database 355  
     ADO Execute SQL Action 355  
     ADO Execute Stored Procedure Action 356  
     SQL Server Backup Database Action 352  
     SQL Server Check Catalogue Action 354  
     SQL Server Check Database Action 354  
     SQL Server DTSRun Action 351  
     SQL Server Execute SQL Action 348  
     SQL Server Rebuild Indexes Action 355  
     SQL Server Remove Unused Space Action 353  
     SQL Server Update DB Statistics Action 355  
 DateTime Action 183  
 Delay Action 54  
 Delete Directory Action 193  
 Delete Files Action 191  
 Delete from Archive 309  
 Delete Log Entry 26  
 Deleting Actions 21  
 Delphi 8 for .Net Compiler Action 257  
 Delphi Compiler Action 244  
 Demeanor Action 337  
 Doc-O-Matic 288  
 DOS Command Action 213  
 Dotfuscator Action 337  
 DTSRun Action 351

## - E -

Edit XML File Action 342  
 Else Action 58  
 EMail Action 275  
 End Action 60  
 Environment Variables 46

Erase CDRW Action 320  
Execute Program Action 211  
Execute SQL Action 348  
Exit Action List Action 56  
Exit Codes 377  
Export Log 26, 182  
Extract Archive 307  
Extract File Version Action 208  
Extract Public Key Action 327  
Extract XML Fragment Action 341  
Extract Zip File Action 302

## - F -

FAQ 385  
File Contents Iterator Action 167  
File Iterator Action 164  
Files & Directories 203  
    Authenticode Action 203  
    Calculate File CRC32 Action 209  
    Calculate File MD5 Action 210  
    Check File Exists Action 190  
    Concatenate Files Action 201  
    Copy Files Action 195  
    Copy/Move File List Action 199  
    Create Directory Action 192  
    Create Text File Action 204  
    Delete Directory Action 193  
    Delete Files Action 191  
    Extract File Version Action 208  
    Move Directory Action 194  
    Move Files Action 197  
    Read Text File Action 206  
    Rename File or Directory Action 202  
    RoboCopy Actions 207  
    Set File Attributes Action 198  
    Text Replace Action 201  
    Touch Files Action 198  
    Write Text File Action 205  
    XCopy Action 207  
Filter 18  
Finally Action 60  
Flow Control 59  
    Case Action 59  
    Catch Action 60  
    Delay Action 54  
    Else Action 58  
    End Action 60

Exit Action List 56  
Finally Action 60  
If .. Then Action 57  
If Prev Action Failed Action 58  
Include FinalBuilder Project 55  
Raise Exception Action 59  
Run Action List 55  
Stop Build Action 59  
Switch Action 59  
Try Action 60  
While Loop 56  
Folder Iterator Action 166  
FTP Client 281  
FxCop Action 337

## - G -

GAC Download Cache Action 332  
GAC Install Action 330  
GAC Uninstall Action 331  
Generate Key Pair Action 324  
Get DateTime Action 183  
Get Disk Free Space Action 216  
Getting Started 25  
Global Script Functions 33  
GPInstall Action 270

## - H -

Help & Manual 292  
Help Compilers 288  
    Doc-O-Matic 288  
    Help & Manual 292  
    HTML Help 291  
    NDoc 292  
    WinHelp 290  
HTML Help 291  
HTTP Get Action 286

## - I -

ICQ Action 286  
IDE 17  
If .. Then Action 57  
If Prev Action Failed Action 58  
Important Changes 15  
Include FinalBuilder Project Action 55

IncrediBuild Action 260  
 INI File Iterator Action 168  
 Ini Files 293, 294  
     Load Variables to INI Action 188  
     Read 293  
     Save Variables to INI Action 186  
     Write 294  
 InnoSetup Action 269  
 Install Key in Container Action 326  
 InstallAnywhere .Net Action 272  
 InstallAnywhere Enterprise Action 271  
 InstallAware Action 273  
 Installers 260  
     GPInstall Action 270  
     InnoSetup Action 269  
     InstallAnywhere .Net Action 272  
     InstallAnywhere Enterprise Action 271  
     InstallAware Action 273  
     InstallShield Pro Action 263  
     InstallShield Pro Windows Installer Editon Action 265  
     InstallShield Developer Action 267  
     InstallShield Universal Action 268  
     NSIS Action 274  
     Wise for Windows Installer Action 262  
     Wise InstallBuilder/InstallMaster Action 260  
 InstallShield Pro Action 263  
 InstallShield Pro Windows Installer Editon Action 265  
 InstallShield Developer Action 267  
 InstallShield Universal Action 268  
 Interactive 169, 171, 175, 176, 178  
     Ask Question Action 175  
     Multi Question Action 176  
     Prompt for File or Directory 178  
     Prompt for Variables 169  
     Prompt for Variables (Enhanced) 171  
 Internet 281  
     FTP Client 281  
     HTTP Get Action 286  
     ICQ Action 286  
     NNTP News Post Action 287  
     Send Email Action 275  
     Telnet Action 284  
 Iterators 167  
     File Contents Iterator 167  
     File Iterator 164  
     Folder Iterator 166

INI File Iterator 168  
 List Iterator 165

## - J -

J# Compiler Action 242  
 J# Project Compiler Action 243  
 Java Compiler Action 226  
 Java JDK Configuration 228  
 JavaScript 28, 32  
 JDK Configuration 228  
 Jikes 228  
 JScript 28, 32

## - K -

Known Problems 384

## - L -

License 16  
 Licensing 312, 313, 314  
     Amradillo 314  
     ASProtect 312  
     ProActivate 313  
 List Archive 309  
 List Iterator Action 165  
 Load Variables From INI Action 188  
 Log Action 182  
 Loop 164, 165, 166, 167, 168

## - M -

MadExcept Action 257  
 Main 17, 19  
 Merge XML Action 340  
 Messages 28  
 Microsoft C# Compiler Action 238  
 Microsoft C# Project Compiler Action 239  
 Microsoft J# Compiler Action 242  
 Microsoft J# Project Compiler Action 243  
 Microsoft VB.NET Compiler Action 240  
 Microsoft VB.NET Project Compiler Action 241  
 Move Directory Action 194  
 Move Files Action 197  
 Moving Actions 21  
 MSBuild Project Action 347



MSTest Action 312  
Multi Question Action 176

## - N -

NAnt Project Action 346  
NDoc 292  
Net Send Message Action 215  
NNTP News Post Action 287  
NullSoft NSIS Action 274  
NUnit Action 311

## - O -

OnFailure 17, 19  
Options 40  
Options Dialog 24  
Output Tree 23  
Overview 10

## - P -

Pascal Analyzer Action 358  
Perforce Command 103  
Perforce Sync with View 102  
PerlScript 15  
Plugins 31  
ProActivate 313  
Project Summary 24  
Project Variables 45  
Prompt for File or Directory Action 178  
Prompt for Variables (Enhanced) Action 171  
Prompt for Variables Action 169  
PythonScript 15

## - Q -

Quick help 25  
QVCS 108  
    Add File 108  
    Check In 108  
    Check Out 108  
    Get Latest Version 108  
    Label Files 108  
    Undo Checkout 108

## - R -

Raise Exception Action 59  
RAR Files 302  
Read Ini 293  
Read Text File Action 206  
Rebuild Indexes Action 355  
Register Assembly in COM 322  
Register DLL/OCX Action 217  
Registry 296  
    Delete Registry Value 296  
    Read Registry Value 296  
    Set Registry Value 296  
Remove Unused Space Action 353  
Rename File or Directory Action 202  
Re-sign Assembly Action 328  
Resource Script Compiler Action 247  
RoboCopy Actions 207  
Run Action List Action 55  
Run AL.EXE Action 321  
Run ASPNET\_REGIIS.EXE Action 323  
Run ResGen.exe Action 335  
Run Script Action 185  
Run SN.EXE Action 329  
Run Status 27

## - S -

Save Variables To INI Action 186  
Scheduling Builds 373  
Script 15  
Script Editor 28  
Script Functions 33  
Scripting 32, 37, 40, 42  
    Accessing Options 40  
    Accessing TStrings 42  
    Events 37  
    Global Functions 33  
Search 18  
Selecting Actions 21  
Send Email 275  
Send Message Action 215  
Set File Attributes Action 198  
Set Variable Action 180  
Shell Execute Action 224  
SMTP 275

- Sound 181
- Source Code Tools 358
  - Pascal Analyzer Action 358
- Source Safe 96
  - Add Files 96
  - Branch 97
  - Check File Status 94
  - Check In 88
  - Check Out 95
  - Checkout 90
  - Get Working Directory 99
  - GetLatest Version 91
  - Label Files 92
  - Override Global Options 99
  - Project Checkouts 93
  - Share 98
- SQL Server 352
  - Backup Database Action 352
  - Check Catalogue Action 354
  - Check Database Action 354
  - DTSRun Action 351
  - Execute SQL Action 348
  - Rebuild Indexes Action 355
  - Remove Unused Space Action 353
  - Update DB Statistics Action 355
- StarTeam 108
  - Add Files 108
  - Apply Label 107
  - Check In 107
  - Check Lock/Unlock Files 107
  - Check Out 107
  - Create Label 107
  - Delete Files 108
  - List Files 108
  - Update Status 108
- Stop Build Action 59
- Subst Drive Action 214
- Subversion 149
  - Add 149
  - Checkout 149
  - Cleanup 150
  - Commit 150
  - Copy 150
  - Export 150
  - Import 150
  - MkDir 150
  - Status 150
  - Update 150

- Support 386
- Switch Action 59
- System Tray 29
- System Variables 46

## - T -

- Team Coherence 115
  - Attach Label 115
  - Check In 113
  - Check Out 114
  - Connect 110
  - Create Label 117
  - Create View 120
  - Delete View 122
  - Detach Label 116
  - Get 112
  - Promote 118
  - Set View 111
  - Sync 119
  - Update View 122
- Telnet Action 284
- Test Archive 307
- Testing Tools 309
  - AQTest Action 309
  - AutomatedQA Test Complete Actions 311
  - MSTest Action 312
  - NUnit Action 311
- Text Replace Action 201
- Touch Files Action 198
- Transform XML Action 340
- Tray Icon 29
- Try Action 60
- TStrings 42
- Type Library Export Action 334
- Type Library Import Action 333

## - U -

- Update Archive 308
- Update DB Statistics Action 355
- Updates 25, 31
- User Action Lists 17
- User Variables 45
- Using Variables 44

## - V -

- Validate XML Action 343
- Validation 28
- Variables 43
  - Action List Parameters 47
  - Adding & Editing Variables 44
  - Environment Variables 46
  - Load from INI Action 188
  - Overview 43
  - Project Variables 45
  - Save To INI Action 186
  - Set Variable Action 180
  - System Variables 46
  - User Variables 45
  - Using Variables 44
- Vault 132
  - Add 132
  - Branch 127
  - Check In 129
  - Check Out 124
  - Cloak 130
  - Commit 128
  - Create Folder 133
  - Create Label 136
  - Delete File/Folder 134
  - Diff 145
  - File Status 149
  - Get 125
  - Get Label 143
  - Get using Wildcards 126
  - Get Version 135
  - GetLabelDiffs 146
  - List Checkouts 147
  - Move File/Folder 137
  - Pin 141
  - Rename File/Folder 138
  - Set Working Folder 144
  - Share File/Folder 139
  - UnCloak 131
  - Undo Checkout 140
  - UnPin 142
- VB.NET Compiler Action 240
- VB.NET Project Compiler Action 241
- VB6 Project Group 366
- VBScript 28, 32
- Verify Strong Name Action 325

- Version Control 88, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 102, 103, 107, 108, 110, 111, 112, 113, 114, 115, 116, 117, 118, 119, 120, 122, 124, 125, 126, 127, 128, 129, 130, 131, 132, 133, 134, 135, 136, 137, 138, 139, 140, 141, 142, 143, 144, 145, 146, 147, 149, 150
  - CVS Actions 103
  - CVS Command 105
  - Perforce Command 103
  - Perforce Sync with View 102
  - QVCS 108
  - Source Safe 88, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99
  - StarTeam 107, 108
  - Subversion 149, 150
  - Team Coherence 110, 111, 112, 113, 114, 115, 116, 117, 118, 119, 120, 122
  - Vault 124, 125, 126, 127, 128, 129, 130, 131, 132, 133, 134, 135, 136, 137, 138, 139, 140, 141, 142, 143, 144, 145, 146, 147, 149
- View Log 26
- Visual C++ 6 233
- Visual Studio .NET 236

## - W -

- Watches 27
- While Loop Action 56
- Window Exists Action 214
- WinHelp 290
- WinRAR Action 302
- Wise for Windows Installer Action 262
- Wise Installer Action 260
- Wizards 359, 366
  - Borland Project Group 359
  - VB6 Project Group 366
- WMI Kill Process Action 221
- WMI Process Info Action 222
- WMI Run Process Action 220
- Write Ini 294
- Write Text File Action 205

## - X -

- XCopy Action 207
- XenoCode Action 338
- XML 342
  - Edit XML File Action 342
  - Extract XML Fragment Action 341

XML 342

Merge XML Action 340

Transform XML Action 340

Validate XML Action 343

## - Z -

Zip Files 299, 302, 304, 307, 308, 309