

INDEX

A

active BVTs, 142
Ambrose, Stephen E.,
 Crazy Horse and Custer, 155
Analyze.exe, 172
ANT (Another Neat Tool), 60
Appleton, Brad, *Software Configuration Management Patterns*, 20
applications
 build lab set up, 50
 testing guide, 234-235
Arend, Mark, 171
assemblies
 managed code, 116
 versioning, 93-95
assigned to fields, work item trackers, 10
automated unit tests, XP (extreme programming), 220

B

backups, UPS (uninterruptible power supply), 49
Ballmer, Steve, 186
batch files
 environment set up, 76
 makefile, 83-85
The Beatles, 183
Beck, Kent, 8, 219
Bennis, Warren G.,
 On Becoming a Leader, 75

Berczuk, Stephen P., *Software Configuration Management Patterns*, 20
Berra, Yoggi, 87
binaries
 security verification, 106
 single worldwide, 129-130
binary-generating tools, 57-59
 NANT, 60
 NMake, 59-60
 recommended tools, 62-64
 scripts, 57
 Visual Studio, 60-61
 XML (Extensible Markup Language), 62
BinPlace tool, 148
biometrics, physical security, 104
bits, security verification, 106
body builds, 20
branch release technique, 176
branching, 19
 configurations, 30-31
 TFSC (Team Foundation Source Control), 27-28
 component, 29
 developer isolation, 28-29
label, 29
partial, 29-30

 promotion modeling, 28
 release, 28
break/fix capabilities, 166
breaks, 6
 builds, 39
 fines, 40
 policy enforcement, 41
 priorities, 39-40
bugs, resolving, 11
builds
 breaks, 6, 39
 fines, 40
 policy enforcement, 41
 priorities, 39-40
defect, 6
defining, 1
 centralized process, 3-4
 terminology, 4-8
 types, 1-2
lab, 4, 104-105. *See also*
 labs
 numbers, 89-90
product flow, 8
 projects, 14-16
 release to servers, 14
 shipping meeting, 11-13
 software development, 9
 solution files, 14
 work item tracker fields, 9-11
tools. *See* binary generating tools
build verification tests. *See* BVTs
BVTs (build verification tests), 5, 36, 133-138

- defining scope, 138-142
- states of existence, 142-143
- C**
- C++ with Managed
 - Extensions, creating managed code, 114
- CABinet files, 147
- cameras, physical security, 104
- Canned Heat, 235
- card key access, physical security, 104
- Carter, Rob, 203
- cases, 167
- case studies, XP (extreme programming), 225-226
- catch-up merges, 27
- central build processes, 1-2
- Central Build Teams, 2-4
- Central WAR, 11-13
- CE builds, system basics, 216-217
- change sets, 25
- charts, quality, 6-7
- check-ins
 - process, testing process maintenance, 70
 - Visual Studio Team, 13
- cherry-pick merges, 27
- clean builds, 5
- clone release technique, 176-177
- CLR (common language runtime), 111
 - delay signing, 116-117
 - command problems, 117-118
 - security, 118-119
 - managed codes, 113-114
- CmdletProvider, 212
- Cmdlets, 212
- Codeline, 18
- codes
 - freezes, 19
 - managed
 - assemblies, 116
 - CLR (common language runtime), 113-114
 - defining, 111-112
 - delay signing, 116-119
 - execution process, 114-115
 - solution files, 119-120
 - sources, 18
 - symbols, 237-239
 - versioning, 173
 - XP (extreme programming), 221
- CodeView symbols, 237
- command-line, Monad, 211-213
- command-line utilities
 - Devenv.exe, 61
 - MSBuild.exe, 60
 - VCBuild.exe, 60
- common language runtime. *See* CLR
- companies, culture, 183-185
 - adapting, 197-198
 - defining, 186-187
 - executive leadership, 187-197
 - leading by example, 198
 - NASA space shuttle disasters, 199-201
- competitive philosophies, 186
- components
 - branching, 29
 - copying before overwrite, 97
 - copying on reboot, 97
 - correct installation problems, 98-99
 - correct version install, 96
 - file versioning field, 93
 - installation to proper directory, 96
 - overwriting, 96
 - registration, 97
 - setup, 147
- computer languages, 57
- configurations
 - build lab hardware, 47-48
 - applications, 50
 - build environment, 50-51
 - operating system set up, 49
 - requirements, 48-49
 - software set up, 49
- SNAP builds, 68-70
- solutions, 61
- source trees
 - terminology, 18-21
- TFSC (Team Foundation Source Control), 24-33
- VSS (Visual SourceSafe), 22-24
- continuous builds. *See* SNAP builds
- continuous integration builds, 6, 222
- control process, tracking source code, 105-106
- Covey, Steven R., *The Seven Habits of Highly Effective People*, 55
- CPR (critical problem resolution), 167
- Crazy Horse and Custer*, 155
- critical problem resolution (CPR), 167
- crunch time, 155
- CSS (customer service and support), 163-164
 - goals, 165-166
 - product team communications, 166-168
- culture, companies, 183-185
 - adapting, 197-198
 - defining, 186-187

- executive leadership, 187-197
 - leading by example, 198
 - NASA space shuttle disasters, 199-201
 - Cunningham, Ward, XP (extreme programming), 219
 - Customer Respect Group, 168
 - customer service and support. *See* CSS
- D**
- daily builds, 36-38
 - successful releases, 39
 - policy enforcement, 41
 - priorities of build breaks, 39-40
 - DDK (Microsoft Driver Development Kit), 148
 - death march, 155
 - debugging symbols, 237-239
 - Debugging Applications for Microsoft .NET and Microsoft Windows*, 237
 - debug machines, build lab hardware, 48
 - defects, 6
 - delay signing, managed code, 116-117
 - command problems, 117-118
 - security, 118-119
 - depend.mk files, 83
 - developer.cmd file, 79-81
 - developers
 - build type, 1-2
 - isolation, 28-29
 - new tools, 203-204
 - Monad, 211-213
 - MSBuild.exe, 204-206
 - VSTB (Visual Studio Team Build), 209-211
 - VSTS (Visual Studio Team System), 206-208
 - developer environments, 77
 - Developing International Software*, 235
 - development
 - product flow, 8
 - projects, 14-16
 - release to server, 14
 - shipping meeting, 11-13
 - software, 9
 - solution files, 14
 - work item tracker fields, 9-11
 - teams, build labs, 4
 - Devenuti, Rick, 166
 - devenv.cmd file, 77-79
 - Devenv.exe command-line utility, 61
 - directories, environment set up, 77
 - disabled BVTs, 142
 - Distinguished Engineer (DE), navigating Microsoft culture, 193-194
 - distributable packages, 1
 - drivers, XP (extreme programming), 221
 - Dynamics of Software Development*, 157
- E**
- e-mail, etiquette, 195-197
 - EE (escalation engineer), 167
 - embedded builds, CE
 - build system basics, 216-217
 - environment
 - build lab, 50-51
 - developer.cmd file, 79-81
 - devenv.cmd file, 77-79
 - makefile, 83-85
 - setenv.cmd file, 82
 - set up, 76-77
 - escalation engineer (EE), 167
 - executives, company culture change, 187-188
 - escalating issues to appropriate executive, 194
 - hire consulting firm, 188-189
 - involve as much as possible, 188
 - maintain integrity, 193-194
 - match company values, 189-192
 - publishing policies and processes, 194-197
 - understand financial impacts, 192
- Extensible Markup Language (XML), 62
- extreme programming. *See* XP
- Extreme Programming*, 8
- Extreme Programming Explained*, 219
- Extreme Programming in Practice*, 219
- F**
- Feldman, Stu, Make, 59
 - Feynman, Richard P., *What Do You Care What Other People Think?*, 200
 - FI (forward integration), 20
 - fields, work item trackers, 9-11
 - filelist.mk files, 83
 - FileMon, 235
 - files
 - CABinet files, 147
 - paths, testing guide, 231

solutions, 61
 testing guide, 229-230
 versioning, 88-93
 WiX, 147
 fines, build breaks, 40
 FixBy fields, work item
 trackers, 10
 Flaar, Chris, 61
 forking, 19
 forward integration (FI), 20
 Fowler, Martin, *Refactoring:
 Improving the Design
 of Existing Code*, 221
 full PDBs, 238

G

Gates, Bill, 185
 general development
 projects, 14
 Gerstner, Lou, 145
 glass masters, 21
 Glasser, Danny, 1
 global makefiles, 83
 globalization, international
 builds, 124
 goals, CSS (customer
 service and support),
 165-166
 golden masters, 21
 golden source trees. *See*
 source trees
 Golden Trees, 19
 green builds, 73
 groups, culture, 183-185
 adapting, 197-198
 defining, 186-187
 executive leadership,
 187-197
 leading by example, 198
 NASA space shuttle
 disasters, 199-201
 guidelines, testing, 143
 Guthrie, Scott, 13

H

Hall, Mike
 mobile and
 embedded devices
 (MED) group, 215
 website, 218
 hardware
 build labs, 47
 configuration build lab,
 47-48
 applications, 50
 build environment,
 50-51
 operating system
 set up, 49
 requirements, 48-49
 software set up, 49
 hard drives, build lab
 requirements, 48-49
 Harvey, Paul, 111
 hidden cameras, physical
 security, 104
 Hilbert, Christopher, 146
 Howard, Michael, *Writing
 Secure Code*, 235
 How to Break Software, *A
 Practical Guide to
 Testing*, 235

I

IDE (Integrated
 Development
 Environment), 61
 IDW (internal developers
 workstation), 6
 IM Wright, 44
 inactive BVTs, 142
 incidents, 167
 incremental builds, 6
 incremental developments,
 XP (extreme
 programming), 220
 inputs, testing guide,
 233-234
 InstallShield website, 146

Integrated Development
 Environment
 (IDE), 61
 integration builds. *See*
 SNAP builds
 internal developers
 workstation (IDW), 6
 internal release servers,
 build lab hardware, 48
 internationally ignorant
 codes, 126
 international builds,
 123-124
 internationally ignorant
 code, 126
 locale-dependent source,
 127-128
 single worldwide binary,
 129-130
 single worldwide source,
 128-129
 support concepts, 124-125
 Unicode, 131-132
International Software, 123
 issues, 167
 iterative developments, XP
 (extreme program-
 ming), 220
 IT department, inherited
 security, 106-107

J-L

Jeffries, Ron, XP (extreme
 programming), 219
 Klingon language, 125
 labeling
 SCC (source code
 control), 91
 SLM (Source Library
 Manger), 91-92
 labeling release
 technique, 176
 labels, branching, 29

- labs
 - builds, 4
 - hardware, 47-51
 - personnel, 51-53
 - physical security, 104-105
 - reasons for, 45-46
 - rules, 46
- languages, international
 - builds, 123-124
 - internationally ignorant code, 126
 - locale-dependent source, 127-128
 - single worldwide binary, 129-130
 - single worldwide source, 128-129
 - support concepts, 124-125
 - Unicode, 131-132
- laptops, physical security, 104
- last known good (LKG), 6
- leaders, company culture change, 187-188
 - escalating issues to appropriate executive, 194
 - hire consulting firm, 188-189
 - involve as much as possible, 188
 - maintain integrity, 193-194
 - match company values, 189-192
 - publishing policies and processes, 194-197
 - understand financial impacts, 192
- LeBlanc, David C., *Writing Secure Code*, 235
- Ledgard, Josh, testing guidelines, 143
- Lennon, John, 183
- Leno, Jay, 185
- Lewis, Ian, 222
- LKG (last known good), 6
- locale-dependent sources, 127-128
- locales, international
 - builds, 124
- localization, international
 - builds, 124
- localizing builds, 123-124
 - internationally ignorant code, 126
 - locale-dependent source, 127-128
 - single worldwide binary, 129-130
 - single worldwide source, 128-129
 - support concepts, 124-125
 - Unicode, 131-132
 - Windows XP, 130-131
- local machines, build type, 1-2
- Lucovsky, Mark, VBLs (Virtual Build Labs), 18
- M**
- machines, SNAP build configuration, 69
- mainlines, 19, 67
- maintenance, testing process, 70
- major versions, 88
- Make, 59
- makefiles, 59, 83-85
- managed codes
 - assemblies, 116
 - CLR (common language runtime), 113-114
 - defining, 111-112
 - delay signing, 116-117
 - command problems, 117-118
 - security, 118-119
- managed execution process, 114-115
- mapping, 25
- Maraia, Vincent, 163
- Maritz, Paul, 133
- marketing, product version numbers, 90
- master languages, international builds, 124
- McCarthy, Jim, 38, 156-157
- McCartney, Paul, 51
- meetings, shipping, 11-13
- Mensching, Rob, 146
- merging, TFSC (Team Foundation Source Control), 26-27
- merging versions, patch releases, 177-179
- Microsoft
 - build tools, 56
 - BVTs (build verification tests), 133
 - cultural shifts, 184-186
 - Developer Division
 - testing guide applications, 234-235
 - file paths, 231
 - files, 229-230
 - inputs, 233-234
 - network connections, 234
 - numeric values, 233
 - references, 235
 - registry, 232
 - strings, 232-233
 - tools, 235-236
 - e-mail etiquette, 195-197
 - IM Wright, 44
 - new developer tools, 203-204
 - Monad, 211-213
 - MSBuild.exe, 204-206
 - VSTB (Visual Studio Team Build), 209-211
 - VSTS (Visual Studio Team System), 206-208

- NT lab hidden
 - cameras, 104
 - setup programs, 146
- Ship It Award, 161
- SoftRel, 160, 161
- Microsoft Developers
 - Network website, 15
- Microsoft Driver
 - Development Kit (DDK), 148
- Microsoft operations
 - manager (MOM), 107
- Microsoft Solution
 - Framework (MSF), 15, 207
- Microsoft website,
 - nmake, 76
- milestones, 19
- minor versions, 88
- MOM (Microsoft operations manager), 107
- Monad, 211-213
- motherboards, build lab
 - requirements, 49
- MSBuild.exe, 204-206
- MSBuild.exe command-line utility, 60
- MSF (Microsoft Solution Framework), 15, 207
- MSH, 211-213
- MUI (multilingual user interface), international builds, 125
- multilanguage builds,
 - 123-124
 - internationally ignorant code, 126
 - locale-dependent source, 127-128
 - single worldwide binary, 129-130
 - single worldwide source, 128-129
 - support concepts, 124-125
 - Unicode, 131-132
- multilingual user interface (MUI), international builds, 125
- Murray, Mike, 184
- N**
- NANT, binary-generating tools, 60
- Nasarre, Christophe, 204
- NASA shuttle disasters, 199
 - background, 199
 - ignoring engineer e-mails, 199
 - investigation conclusions, 200-201
 - top level organization, 200
- navigators, XP (extreme programming), 221
- .NET Framework
 - assembly versioning, 93-95
 - installation platforms, 112
 - managed code
 - assemblies, 116
 - CLR (common language runtime), 113-114
 - defining, 111-112
 - delay signing, 116-119
 - managed execution process, 114-115
 - rewrites, 108
 - networks, connection
 - testing guide, 234
 - nightly builds, 35-36
- NMake, binary-generating tools, 59-60, 76
- nmake all command, 85
- nmake clean command, 85
- nmake depend command, 85
- nmake tree command, 85
- NT lab, hidden cameras, 104
- numbers
 - file versioning, 89, 92-93
 - versioning, 90
- numeric values, testing guide, 233
- O-P**
- On Becoming a Leader*, 75
- operations staff, SNAP
 - builds, 71
- Orcas, 204
- OS (operating system), 49, 96
- pace, XP (extreme programming), 222
- packages, 1
- pair programming, 221
- parallel releases, 179
- Parkerson, Scott, 94
- partial branching, 29-30
- patch releases, 177-179
- paths, file testing guide, 231
- personnel, build lab, 51-53
- Peters, Chris, 39, 165
- physical security, 103-105
- Pipelines, 212
- Plan of Attack*, 39
- post-builds, 5
- Pottery Barn rule, 39
- power supplies, build lab
 - requirements, 49
- pre-builds, 5
- primary releases, 174
 - branch, 176
 - clone, 176-177
 - labeling, 176
 - share and pin, 174
- priority fields, work item trackers, 10
- private branches, SNAP
 - builds, 67
- private builds, 19, 23
- processors, build lab
 - requirements, 48
- product flows, 8
- projects, 14-16
- release to servers, 14
- shipping meeting, 11-13

- software development, 9
- solution files, 14
- work item tracker fields, 9-11
- product marketing, version numbers, 90
- product support services (PSS), 163
- product teams, CSS, 166-168
- programming, XP (extreme programming)
 - fundamental characteristics, 219-222
 - Microsoft case study, 225-226
 - refactoring, 222-224
 - scenario, 224-225
 - TDD (test-driven development), 222-224
- projects, 1-2, 14-16
- promotion modeling, 28
- PSINFO.EXE, 50
- PSS (product support services), 163
- public builds, 19, 23
- Q-R**
- QFE (quick-fix engineering), 167
- quality charts, 6-7
- RAM, build lab requirements, 48
- Refactoring
 - Improving the Design of Existing Code*, 221
 - XP (extreme programming), 221-224
- references, Developer Division testing guide, 235
- registry, testing guide, 232
- RegMon, 235
- regression testing, XP (extreme programming), 220
- Reis, Luis Miguel, 220
- released to Web (RTW), 21
- releases
 - daily builds, 39
 - policy enforcement, 41
 - priorities of breaks, 39-40
 - Microsoft SoftRel, 160-161
 - parallel, 179
 - patch, 177-179
 - preparation steps, 157-158
 - primary, 174
 - branch, 176
 - clone, 176-177
 - labeling, 176
 - share and pin, 174
 - Rule 21, 156-157
 - source trees, 158-159
 - VSS (Visual SourceSafe), 172-173
 - XP (extreme programming), 220
- release branching, 28
- release build machines, build lab hardware, 48
- repositories, 25
- reverse integration (RI), 20
- revisions, 89
- RI (reverse integration), 20
- Richter, Jeffrey, .NET Framework, 108
- Robbins, John, *Debugging Applications for Microsoft .NET and Microsoft Windows*, 237
- Rockne, Knute, 198
- RTW (released to Web), 21
- rules, build labs, 46
- RULES.mk files, 83
- Rule 21, McCarthy, Jim, 156-157
- S**
- sandbox builds, 19
- SCC (Source Code Control), 17, 91
- SCC tool (source code control tool), 171
- scripts, 57
- security
 - binary verification, 106
 - delay signing, 118-119
 - IT infrastructure, 106-107
 - multilevel approach, 102
 - .NET Framework rewrite, 108
 - physical, 103-105
 - tracking source changes, 105-106
- Seiwald, Christopher, 20
- self-extracting EXE, 97-98
- self-host builds, 134
- self-test builds, 134
- servers
 - build lab hardware, 48
 - product release, 14
 - service packs, 177
 - build numbers, 93
 - merging versions, 177-179
- Service Request (SR), 167
- setenv.cmd file, 82
- setup, 145-146
 - basic definitions, 147-148
 - components, 147
 - programs used by Microsoft, 146
 - SKU, 147
 - WiX build components, 148-153
- Setupbuild.cmd, 152
- The Seven Habits of Highly Effective People*, 55

- severity fields, work item trackers, 10
 - share and pin release technique, 174
 - shared code ownership, XP (extreme programming), 221
 - shell, Monad, 211-213
 - shelving, TFSC (Team Foundation Source Control), 31-32
 - Shiny New Automation Process builds. *See* SNAP builds
 - shipping, 37
 - meeting, 11-13
 - Microsoft SoftRel, 160-161
 - preparation steps, 157-158
 - Rule 21, 156-157
 - source trees, 158-159
 - Ship It Award, 161
 - single worldwide binaries, 129-130
 - single worldwide sources, 128-129
 - SKU (Stock Keeping Unit), 147
 - SKU.XML files, 147, 150-152
 - slime, 91-92
 - SLM (Source Library Manager), 91-92
 - Smith, Edward J., 171
 - smoke tests, 133-137
 - snapshots, 19
 - SNAP builds (Shiny New Automation Process builds), 65-66
 - defining, 66
 - implementing, 67
 - operations staff, 71
 - sample configuration, 68-70
 - system core, 67-68
 - throughput management, 71-73
 - SoftRel, Microsoft, 160-161
 - software
 - build lab set up, 49
 - daily builds, 37-41
 - development flow, 9
 - shipping
 - Microsoft SoftRel, 160-161
 - preparation steps, 157-158
 - Rule 21, 156-157
 - Software Configuration Management Patterns*, 20, 91
 - Software for Your Head*, 157
 - solutions
 - configuration, 61
 - files, 14, 61
 - solution files, managing
 - code, 119-120
 - sources, 18
 - codes, 18
 - physical security, 104-105
 - tracking changes, 105-106
 - Source Code Control (SCC), 17, 91
 - source code control tool (SCC tool), 171
 - Source Library Manager (SLM), 91-92
 - source life cycles, 102
 - source trees
 - numbering, 89
 - shipping mode, 158-159
 - SNAP builds, 67
 - terminology, 18-21
 - TFSC (Team Foundation Source Control), 24-25
 - branching, 27-30
 - configurations, 30-31
 - merging functionality, 26-27
 - offline checkout/
 - check-in, 32-33
 - shelving, 31-32
 - VSS (Visual SourceSafe)
 - setup, 22
 - source control, 24
 - VBLs, 22-24
 - space shuttle disasters, 199
 - background, 199
 - ignoring engineer e-mails, 199
 - investigation conclusions, 200-201
 - top level organization, 200
 - SP (support professionals), 167
 - SR (Service Request), 167
 - SS.exe, 172
 - status fields, work item trackers, 10
 - Stock Keeping Unit (SKU), 147
 - strings, testing guide, 232-233
 - stripped PDBs, 238
 - substatus fields, work item trackers, 10
 - support professionals (SP), 167
 - symbols, debugging, 237-239
 - SysInternals website, 50
- T**
- TDD (test-driven development), 222-224
 - teams
 - Central Build, 2-4
 - testing check list, 141-142
 - WAR, 12
 - Team Foundation Source Control (TFSC), 92
 - test-driven development (TDD), 222-224
 - testing, 133-135

- bugs, 11
- BVTs (build verification tests), 137-138
 - defining scope, 138-142
 - states of existence, 142-143
- check list, 141-142
- guide
 - applications, 234-235
 - files, 229-230
 - file paths, 231
 - inputs, 233-234
 - network connections, 234
 - numeric values, 233
 - references, 235
 - registry, 232
 - strings, 232-233
 - tools, 235-236
- guidelines, 143
- smoke tests, 135-137
- SNAP build machine
 - configuration, 69
- XP (extreme programming), 220-224
- Test Driven Development*, 8
- TFSC (Team Foundation Source Control), 24-25, 92
- branching, 27-28
 - component, 29
 - developer isolation, 28-29
 - label, 29
 - partial, 29-30
 - promotion modeling, 28
 - release, 28
- configurations, 30-31
- merging functionality, 26-27
- offline checkout/check-in, 32-33
- shelving, 31-32
- throughput, SNAP builds, 71-73
- tickets, 167
- tools
 - binary generating, 57-59
 - NANT, 60
 - NMake, 59-60
 - recommended tools, 62-64
 - Visual Studio, 60-61
 - XML (Extensible Markup Language), 62
 - new developments, 203-204
 - Monad, 211-213
 - MSBuild.exe, 204-206
 - VSTB (Visual Studio Team Build), 209-211
 - VSTS (Visual Studio Team System), 206-208
 - scripts, 57
 - testing guide, 235-236
 - VSS (Visual SourceSafe), 172
- trackers, work items, 9-11
- tracking source changes, 105-106
- triage, 13
- trunks, 19
- U**
- UI (user interface)
 - languages, international builds, 125
- Unicode, international builds, 131-132
- uninterruptible power supply (UPS), 49
- unit tests, 135
- UPS (uninterruptible power supply), 49
- usability tests, Chris Peters, 165
- user interface (UI)
 - languages, international builds, 125
- users
 - locale, international builds, 124
 - XP (extreme programming), 221
- V**
- Vaughn, Dr. Diane, 201
- VBLs (Virtual Build Labs), 18-19, 22-24, 92
- VCBuild.exe command-line utility, 60
- VerCheck tool (vercheck.exe), 141
- verification test, 133-135
 - BVTs (build verification tests), 137-138
 - defining scope, 138-142
 - states of existence, 142-143
 - smoke tests, 135-137
- versioning
 - affects on set up, 95
 - component copying
 - before overwrite, 97
 - copying components, 96-97
 - overwriting
 - components, 96
 - registering
 - components, 97
 - self-extracting EXE, 97-98
 - specific components, 96
 - testing on real-world systems, 98
- assembly, 93-95
- build numbers, 90
- correct installation
 - problems, 98-99
- files, 88-93
- merging, 177-179
- reasons important, 87-88

- SCC (source code control), 91
 - VSS (Visual SourceSafe), 173
 - vf.exe (Visual File Information tool), 140
 - video cameras, physical security, 104
 - Virtual Build Labs (VBLs), 18-19, 22-24, 92
 - Visual C++, creating managed code, 114
 - Visual File Information tool (vf.exe), 140
 - Visual SourceSafe. *See* VSS
 - Visual Studio
 - binary-generating tools, 60-61
 - projects, 14-16
 - solution files, 14
 - team check-ins, 13
 - Visual Studio .NET
 - projects, 14
 - Visual Studio Integrated Development Environment (VS IDE), 61
 - Visual Studio Team Build (VSTB), 209-211
 - Visual Studio Team Developer, 207
 - Visual Studio Team Foundation, 207
 - Visual Studio Team Suite, 207
 - Visual Studio Team System (VSTS), 17, 206-208
 - Visual Studio Team Test, 207
 - VSS (Visual SourceSafe), 17, 171-172
 - Administrator, 172
 - Explorer, 172
 - source control, 24
 - source tree setup, 22
 - tools, 172
 - VBLs multisite development, 22-24
 - versioning, 173
 - VSS projects, 14
 - VSTB (Visual Studio Team Build), 209-211
 - VSTS (Visual Studio Team System), 17, 206-208
 - VS IDE (Visual Studio Integrated Development Environment), 61
- W**
- WAR, 11-13
 - websites
 - Hall, Mike, 218
 - InstallShield, 146
 - Microsoft, nmake, 76
 - Microsoft Developers Network, 15
 - SysInternals, 50
 - Windows Update, 112
 - Wise Solutions, 146
 - WiX, 146
 - Welch, Jack, 123
 - What Do You Care What Other People Think?*, 200
 - Whittaker, James A., *How to Break Software: A Practical Guide to Testing*, 235
 - wicks, 146
 - Windows, debugging symbols, 238-239
 - Windows Installer, 147
 - Windows Installer XML (WiX), 146
 - Windows Update website, 112
 - Windows XP, localized builds, 130-131
 - Windows XPe (XP Embedded), 215-216
 - Wingerd, Laura, 20
 - Wise Solutions website, 146
 - WiX (Windows Installer XML), 146-147
 - Wooden, John, 8
 - Woodward, Bob, *Plan of Attack*, 39
 - working folders, 25
 - workspaces, 25
 - work items, trackers, 9-11
 - Writing Secure Code*, 235
 - WXS files, 147
- X-Y-Z**
- XML (Extensible Markup Language), 62
 - XP (extreme programming), 6, 219
 - fundamental characteristics, 219-222
 - Microsoft case study, 225-226
 - programming scenario, 224-225
 - refactoring, 222-224
 - TDD (test-driven development), 222-224