

Numerics

100% test coverage 291
5-rappen round 174
test 174

A

acceptance testing 169, 192–197
introduction 170
ActualValue 150
Adzic, Gojko 195
AfterBuild 80
agile
and documentation 11
development process 5
angle-bracket tax 95
AnkhSVN 14, 42
Ant 68–70
Anthill 95
application lifetime management (ALM) 288
ASP.NET
handlers 253
modules 253
assembly, signing 24
AssemblyInfo 84, 86
AssemblyInfoReference 88
AssemblyVersion 84
Assert.AreEqual 150
assertion, definition of 150
Automake 68
automatic test 168
automating standards
compliance 11

automation platform,
definition 67

B

bad dependency 282
Bamboo 95
BeforeBuild 80
BisSubscribe 137
black-box testing 168
broken build, rules 122
Build 74
target 72, 153
build
automating 20
breaking 172–173
broken. *See* broken build
continuous. *See* continuous
build
cyclical, no such thing as 280
daily. *See* daily build
definition of 5
driving with a build script 279
incremental. *See* incremental
build
manual integration. *See* manual
integration build
output location 281
parallel, running often to
reveal race conditions 282
pass-based. *See* pass-based
build
QA. *See* QA build
release. *See* release build

seven deadly sins of slow software builds 278–282
spreading to multiple machines 277
staging. *See* staging build types 7–8
weekly. *See* weekly build
build agent
categorized 283
theory 282–283
workload 283
build automation 67–70
build machine
bigger, to speed up CI 277
different, for each project 277
build manager 15
build master 92
build process
linear 92
looped 92
build rules, centralizing 282
build script 92
naming 81
targets, side effects of 279
build state, immediate
access 122
build token 90
BuildDate 78
building continuously 7

C

C#, analysis with StyleCop 209–213

- Capability Maturity Model (CMM) 288
- CCNet 15, 95–102
 - Administrator Dashboard 126
 - alternative feedback 128–129
 - and NUnit 151–154
 - and StyleCop 212
 - and TeamCity 212
 - angle-bracket tax 95
 - artifacts 98
 - configuration file 97
 - Configuration Validation tool 98
 - configuring 97–100
 - console mode 99
 - distributing builds 101
 - email publisher 129
 - feedback 122–129
 - feedback mechanisms 96
 - installing 96–97
 - introduction 94
 - MSBuild 98
 - NUnit XSL transformer 154
 - project 97
 - publishers 101, 128
 - remoting endpoint 125
 - source control path 98
 - tasks 98
 - triggers 98, 100–102
 - inner trigger 101
 - interval trigger 98, 100
 - project trigger 100
 - schedule trigger 101
 - using IIS 96
 - Web Dashboard 96, 122–126, 153
 - and PartCover 155
 - build 123
 - build reports 125
 - configuring 123
 - FitNesse report 125
 - FxCop report 125
 - NUnit report 125
 - organization of 123
 - plug-ins 123
 - project 123
 - server 123
 - working as service 96
 - working directory 98
 - working standalone 96
 - XML Log Publisher 128
 - XSL transformer 154
- CCTray 94, 126–128
 - configuration 126
 - feedback 126
 - indicator 127
 - noise level 126
 - Windows Notification area 126
- change, and CI 8
- CI 4–12
 - and developer tasks 9
 - and ensuring deployable software 9
 - and existing projects 9
 - and source control 30
 - and system maintenance 8
 - and the development process 5–7
 - as centerpiece of development 4
 - automating code coverage with 11
 - automating
 - documentation 11
 - automating standards compliance 11
 - beginning with 7
 - changing via baby steps 8
 - complying with Sarbanes-Oxley 287
 - continuous database integration 11
 - creating and testing the installation process 11
 - definition of 4–5
 - example project. *See* example project
 - feedback mechanism 10
 - golden rule 21
 - hardware and software costs 9
 - increasing visibility of project with 9, 12
 - infinite loop 91
 - legal roadblocks 287–288
 - maturity model 288–292
 - multiple integrations per day 4
 - objections to, overcoming 8–10
 - reasons to use 9
 - reducing risk with 9–12
 - scaling 282–287
 - server 15
 - speeding up 277–278
 - speeding up incremental builds with 10
 - testing 168–173
 - tools 13–20
 - build manager 15
 - CI server 15
 - code analysis 18
 - essential 14–16
 - feedback mechanism 15
 - source code control 14
 - testing tools 20
 - unit test framework 16
- CI process 90–92
 - broken build 122
 - ClickOnce 251–253
 - creating installation package 241
 - custom code-analysis rules 217
 - failed 122
 - feedback
 - extending programmatically 139, 142
 - getting programmatically 139
 - LED message board 139–141
 - over Skype 142
 - programming mechanisms 140
 - SMS notification 142
 - via LED 141
 - via USB gadget 139–141
 - periodic poll 93
 - reducing dependencies 91
 - state discovery 121–122
 - StyleCop 210–213
 - WiX 247–248
 - working state 121
 - yet another successful build state 121
- CI server 90
 - and Linux kernel development 94
 - angle-bracket tax (XML) 94
 - CCNet 95–102
 - choosing 92–95
 - aspects to consider 94
 - cost 94
 - dedicated machine 91
 - documentation 94
 - for .NET 94–95
 - freeing from assumptions 91
 - functionality 94
 - hardware 92
 - integrating unit tests with 151
 - interoperability 94
 - physical vs. virtualized 91
 - simple 93

- CI server (*continued*)
 - support 94
 - TeamCity 102–112
 - TFS Team Build 113–118
 - usability 94
 - vanilla 91
 - your own, vs. manual integration build 92–93
 - Clean target 72, 74
 - ClickOnce 20, 248–253
 - advantages 248
 - automating 251
 - deployment technology 248
 - Front Page Extensions 249
 - getting version number from SVN revision 252
 - installing 249
 - manipulating publication HTML 252
 - Publish target 251
 - publishing 249
 - sandbox 248
 - smart clients 248
 - use of web server 249
 - versioning 251
 - Windows Form 248
 - with MSBuild 251
 - WPF 248
 - Code Analysis 200–208
 - breaking the build 206
 - enabling 200
 - Microsoft Minimum Recommended Rules 201
 - Suppress Message window 201
 - suppressions 201
 - code analysis
 - dynamic. *See* dynamic analysis
 - extending 218–223
 - static. *See* static analysis
 - tools 18
 - code coverage, definition 11
 - Code Query Language. *See* NDepend, CQL
 - code, removing, to speed up CI 277
 - code-facing test 171
 - Command 74
 - compile manager 67
 - Condition 72
 - continuous build 7
 - continuous database integration
 - definition of 261–262
 - rolling your own 262–264
 - using SSMS 262
 - with RoundhouseE 264–265
 - with Visual Studio 266–275
 - Continuous Integration and Testing Conference (CITCON) 289
 - continuous integration. *See* CI
 - Copy 73–74
 - CruiseControl 94
 - CruiseControl.NET. *See* CCNet
 - Csc 72
 - csc 69
 - Cunningham, Ward 192
 - CustomBuildExtensions 84
 - customer-facing test 172
-
- D**
-
- daily build 7–8, 90
 - dashboard.config 123, 154
 - Data Bound Generator 269
 - Data Dude 266–275
 - database
 - deploy options 268
 - files, keeping under source control 262
 - placing under source control 267
 - schema, handling changes via scripts 261
 - target connection 267
 - test data
 - generating 268–271
 - readability 269
 - unit testing 271–273
 - database integration, continuous. *See* continuous database integration
 - database object, creating script for 262
 - data-generation plan 269
 - Delete 72, 74
 - demilitarized zone (DMZ) 288
 - dependency, bad. *See* bad dependency
 - depends 69
 - Deploy 78
 - Design Guidelines for Class Library Developers 200
 - developers, and CI 9
 - development
 - CI as centerpiece of 4
 - process 5
 - documentation
 - automating 11
 - formatting 228
 - XML notation 228–233
 - common tags 228–231
 - formatting comments 231–233
 - formatting with Sandcastle 233–239
 - Dream Cheeky 139
 - Duval, Paul, definition of CI 4
 - dynamic analysis, definition of 200
-
- E**
-
- Electric Cloud 95, 278
 - EmitDebugInformation 72
 - Enterprise CI Maturity Model (ECIMM) 288–292
 - building 289
 - deploying 290–291
 - feedback 292
 - testing 291
 - error-driven development 146
 - example project
 - architecture 21
 - calculation core 21–26
 - folder structure 27
 - introduction 20–27
 - user interfaces 26–27
 - Exec 73–74
 - Execute() 85
 - ExpectedRate 150
 - [ExpectException] 151
 - extreme programming. *See* XP
-
- F**
-
- failonerror 69
 - feedback mechanism 15
 - file, updating more than once 279–280
 - filesystem deployment 258
 - FileUpdate 88
 - financial calculator, putting under CI 90
 - Fit 192
 - FitNesse 20, 169, 192–197
 - column fixture 193
 - edit and test mode 195
 - fixtures 193
 - in the CI process 196
 - installing 192
 - Java Runtime environment 192

FitNesse (*continued*)

- log file 197
 - running the server 192
 - syntax 194
 - test
 - running 194
 - running remotely 196
 - writing 194
 - test runner 194
 - TestRunner.exe 196
 - wiki style 192
- Fowler, Martin
- definition of CI 4
 - first article about CI 94
- Fredrick, Jeffrey 288
- functional test,
- introduction 170
- FxCop 18, 200–208
- and StyleCop, conflicting rules 209
 - and TeamCity 207
 - and TFS 207
 - Check method 214
 - continuous analysis 203–206
 - custom rule, incorporating into CI process 217
 - custom rules 213–218
 - extending 214–216
 - for compiled .NET 200
 - FxCopCmd.exe 204
 - GUI 203
 - ignoring rules 204
 - integrating with CI servers 206–208
 - project 203
 - report 206
 - Rule tag 215
 - standalone 203
 - suppressions 201
 - XML rule definition 216
 - XSD transformation 206
 - XSL style sheet 207

G

- GhostDoc 231
- Git 14, 277
 - as distributed system 33
 - cost-benefit factor 33
 - Linus Torvalds' use of 33
 - source control aspects 32
- Global Assembly Cache (GAC) 24
- golden rule of CI 21
- GUI testing, tools 180

H

- Hello World 12–13
- HID 139
 - feedback, via HID-enabled device 139
 - generic .NET device 141
- horizontal scaling 284
- HTML
 - table 233
 - tables in XML documentation 233
- Hudson 15, 95
- human interface device (HID). *See* HID

I

- IBM, Telelogic 14
- Identity 88
- [Ignore] 151
- IgnoreTest(). 150
- Importance 74
- importData 163
- incremental build 7
 - speeding up, with CI 10
- Inno Setup 20
- installation package 241
- installation, creating and testing 11
- integration test 170
 - separating from unit tests 277
- integration testing 169
 - bottom-up approach 175
 - in CI 173–179
 - leasing calculator 174
 - MSTest 173
 - NUnit 173
 - project layout 175
 - running 174
- IntelliSense 78, 233
 - XML file to use for 228
- ITask 83
- ItemGroup 72
- ItemGroups 79

J

- Jabber, receiving TeamCity notifications in 134
- JetBrains Duplicates Finder 223

K

- Kaner, Cem, *Testing Computer Software* 11

L

- leasing/credit calculator. *See* example project
- LOCAL.DBDeployment.CommandLine.bat 264

M

- Mail 78
- maintenance, and CI 8
- make 67–68
- MakeDir 73–74
- manual integration build 92
 - vs. your own CI server 92–93
- manual test 168
 - and the CI process 172
- MbUnit 16
- Message 73–74
- Meyers, G. J. 11
- Microsoft
 - FxCop. *See* FxCop
 - StyleCop. *See* StyleCop
 - Team Foundation Server (TFS). *See* TFS
 - Visual SourceSafe (VSS). *See* VSS
 - Microsoft Deployment Tool. *See* MS Deploy
 - Microsoft Installer package. *See* MSI package
 - Microsoft UI Automation. *See* UIA
 - Microsoft Unit Testing Framework. *See* MSTest
 - Microsoft.TeamFoundation 140
 - Microsoft.TeamFoundation.Build.Client 140
 - Microsoft.TeamFoundation.Client 140
 - Microsoft.VisualStudio.TestTools.UnitTesting 157
- Minick, Eric 288
- mocking a unit test 170
- mocking file operations 176–179
- monolith 281
 - breaking up 281
- Moq 177

MS Deploy 253–258
 automating 256–258
 command file 258
 command-line
 installation 258
 configuration files 255
 creating packages 254
 database management 256
 deployment 254
 gathering MSBuild targets
 and DLLs 256
 IIS settings 254
 manifests 258
 migration 254
 Package/Publish settings 254
 publishing 254
 replacing configuration
 values 256
 synchronization 254
 using MSBuild 256
 Visual Studio 2010 254–256
 XML transformations 255
 MSBuild 15, 68, 70–78
 and Visual Studio 78–83
 code-analysis task 204
 Community Tasks 76–78
 example 77
 installing 76
 Copy 73
 creating using Visual Studio
 and MSBuild 243
 custom tasks 83–88
 implementing 84–86
 using 86–88
 dry run 258
 Exec 73
 FileUpdate 205, 252
 Hello World 71–72
 ItemGroup 72
 Mail 78
 MakeDir 73
 Message 73
 Project 72
 PropertyGroup 72
 RemoveDir 73
 running NUnit in 152
 script
 extending 73–76
 starting with Visual
 Studio 82–83
 set of tasks 72
 SvnInfo 252
 Time 78
 UsingTask 78
 verbosity 75

MSBuild Community Task 152
 MSBuild Community tasks 252
 MSBuild Sidekick 15
 MSBuild task 81
 MSBuildProjectDirectory 74
 MSBuildTasks 84
 MSBuildToolsPath 80
 MSDN subscription 95
 MSI package 241–243
 adding project output 242
 configuring 241
 creating continuously 243–
 244
 creating in Visual Studio 241
 using Visual Studio,
 drawbacks 244
 MSTest 16, 144, 157
 and TFS 2010 160–161
 and third-party CI servers 161
 integrating with CCNet 162
 integrating with
 TeamCity 162
 test coverage 160
 unit test, creating 157–160
 mstest 163
 Muzaffar, Usman 278

N

NAnt 15, 68
 executables 69
 launching 69
 properties 69
 NCover 18, 154
 NDepend 218–222
 and CCNet 222
 artifacts 221
 automation 219
 breaking build 221
 code metrics 218
 CQL 218
 extending with CQL 222
 integrating with
 TeamCity 221
 licensing 218
 MSBuild task 219
 report 221
 static code analysis 218
 Visual NDepend 220
 Visual Studio plug-in 219
 XSL 222
 nmake 68
 Non-Sucking Service Manager.
See NSSM
 NSSM 191

NUnit 16
 and CCNet 151–154
 attributes 150
 command-line test
 runner 152
 GUI test runner 150
 installing 149
 running with MSBuild 152
 unit testing with 149–151
 using with White 180
 XML report 153
 NUnit.Mocks 177
 NUnitForms 20

O

O'Brien, Mike, HID device
 library 141
 objections, overcoming 8–10
 Oshero, Roy, *The Art of Unit
 Testing* 164
 OutputAssembly 72

P

PartCover 154
 pass-based build 280–281
 Path 86
 PostBuildEvent 80
 PowerCommands 79
 PreBuildEvent 80
 Project 72
 project directory, organizing 21
 project, existing, placing under
 CI 9
 Property 86
 PropertyGroup 72
 PropertyGroups 79
 psychopath 200

Q

QA build 7–8

R

race condition, revealing with
 parallel builds 282
 rake 68
 release build 7–8
 release engineer 92
 RemoveDir 73–74
 repository 21

revision control. *See* source control

RevisionNumber 88

Rhino Mocks 177

risk

database updates 11

of bugs 11

of schedule slippage 10

reducing with CI 9–12

RoundhouseE

calling 265

command-line switches 265

continuous database integration 264–265

data migration 264

description of 264

non-interactive mode 265

running without switches 265

sample applications 264

S

Sandcastle 20, 233–239

daily build 238

GUI 235

Help File Builder (SHFB). *See* SHFB

HelpFileFormat 236

HtmlHelp 236

in the CI process 237–239

using with TeamCity 238

Sarbanes-Oxley. *See* SOX

Satir Change Model

and technology adoption 10

steps 10

Satir, Virginia 10

scaling wall 70

SCons 68

script

creating for database

object 262

running automatically 264

scrum, development process 5

Selenium 20, 169, 185–189

.NET 189

ASP.NET 185

assertion 187

C# test format 187

commands 187

Firefox plug-in 185

IDE 185

installing 189

performing test remotely 189

RC 185

RC server 189

installing 189

installing as Windows

service 191

recording tests 185

running in different browser contexts 189

running tests 187

table test format 185

targets 187

test suite 187

values 187

self-service test deploy 290

separation of powers 279

serialization, unnecessary, fixing 279

setup project 241

seven deadly sins of slow software builds 278–282

SHFB 233

Documentation Sources 235

MSBuild 235–237

MSBuild task 236

Silverlight, test

automation 182–184

Skype, providing build notifications via 142

SmartSVN 42

Smith, Steven 10

source control 14, 29–65

and CI golden rule 32

benefits 31–32

centralized systems 33

centralized vs. distributed 32–33

choosing 30–35

CI 30

distributed systems 33

explicit checkout 34

explicit lock 31

file blocking 34

vs. non-file blocking 32, 34

free vs. paid 32–33

labeling revisions 31

locking vs. blocking 35

transactional vs.

nontransactional 32

Sources attribute 72

SOX 259

and CI 287

SQL script, editing in T-SQL

Editor 267

SQL Server, developers running

local copies 261

SSL, and SVN 36

SSMS, and continuous database integration 262

stability test, introduction 171

staging build 7–8

standards, compliance,

automating 11

static analysis, definition in 200

stored procedure, unit

testing 271–273

StyleCop 18, 209–213

analysis

continuous 210–213

performing 210

analysis report 211

AnalyzeDocument() 217

and FxCop, conflicting rules 209

and TeamCity 212

breaking the build 211

build report 213

custom rule, incorporating

into CI process 218

custom rules 213–218

extending 216–217

installing 209

MSBuild StyleCopTask 211

MSBuild targets file 210

report page 212

rules

suppressing 210

turning on and off 210

Settings 209

Visual Studio plug-in 209

XML rule definition 217

XSL 212

Subversion. *See* SVN

SVN 14

AnkhSVN. *See* AnkhSVN

as centralized system 33

authentication 36

command-line utilities 42

commit 45–50

frequent 48

connection protocol 39

cost-benefit factor 33

directory structure with

references 52

externals 50

files

excluding 45

ignoring 47

merging 48

integration with Windows

Explorer 42

locking 35

SVN (*continued*)
 messages 45
 on Windows Server 35
 protocol 38
 pulling before committing 48
 repository
 branch 41
 clean 47
 creating 37–39
 multiple-repository
 layout 39
 referencing 50–53
 single-repository layout 39
 structure 41
 tag 41
 trunk 41
 vs. directory 39
 resolving conflicts 48
 SmartSVN. *See* SmartSVN
 source control aspects 32
 source control server 35–42
 svn directory 43
 TortoiseSVN client. *See*
 TortoiseSVN
 update 48–50
 URL construction 38
 virtual directory 38
 VisualSVN Server. *See*
 VisualSVN Server
 working copy 43–45
 SvnInfo 84
 system test, introduction 170
 system testing 169

T

task loop 91
 TaskName 88
 Team Foundation Server (TFS).
 See TFS
 Team Foundation Server 2010.
 See TFS
 team, objections,
 overcoming 8–10
 TeamCity 15, 102–112
 Add-in for Visual Studio 110
 alternative notification 134–
 135
 analyzing code
 duplication 223
 and NCover 156
 and NDepend 221
 and PartCover 156
 artifacts 132
 Build Agent 104
 build agent 102
 build feedback 130–132
 build grid 283
 build runner 108
 build script 130
 checkout mode 106
 configuration 104–110
 continuous build 109
 dry run 110
 Duplicates Finder 223
 email notifier 134
 feedback 129–135
 IDE plug-ins 133
 installing 102–104
 integration build 93
 introduction 94
 Jabber notification 134
 JetBrains 94
 labeling builds 108
 licensing 102
 login 104
 MSBuild 108
 NAnt 108
 pre-test commit 103
 pre-tested commit 110–112
 project 104
 project configurations 130
 project settings 106
 project-level feedback 130
 Projects tab 130
 scaling 283–287
 server-level feedback 130
 service messages 163
 tabs 130
 test report 163
 triggering 108
 using Sandcastle with 238
 version control system 106
 Visual Studio Solution 108
 Windows Tray Notifier 132–
 134
 noise level 132
 technology adoption, and Satir
 Change Model 10
 Telelogic 14
 [Test] 150
 test
 acceptance test. *See* accep-
 tance test
 automatic test. *See* automatic
 test
 code-facing test. *See* code-
 facing test
 customer-facing test. *See*
 customer-facing test
 execution time 172
 failing fast 172–173
 integration test. *See* integra-
 tion test
 manual test. *See* manual test
 stability test. *See* stability test
 system test. *See* system test
 timing 172
 types 169–171
 user-acceptance test. *See* user-
 acceptance test
 written by domain
 experts 168
 written by someone other
 than developer 168, 171–
 172
 test coverage 154–157
 test runner 194
 test-driven development
 (TDD) 168
 [TestFixture] 150
 testing
 abstracting actual
 implementation 176
 acceptance testing. *See* accep-
 tance testing
 black-box testing. *See* black-
 box testing
 CSV 176
 I/O 176
 injecting functionality 177
 integration testing. *See* integra-
 tion testing
 known state 178
 system testing. *See* system
 testing
 timeline 169
 UI testing. *See* UI testing
 unit testing. *See* unit testing
 white-box testing. *See* white-
 box testing
 testing pyramid 171
 testing tools 20
 [TestMethod()] 159
 TFS 15, 94, 113–118
 Administration Console 55
 alternative feedback 137
 and build controllers 113
 and CI 54
 and Reporting Services 54
 and SharePoint 54
 and SQL Server Express 137
 and StyleCop 211
 and Visual Studio Team
 Explorer 58–60

- TFS (*continued*)
- API 137
 - as centralized system 33
 - associating work item with check-in 62
 - Basic configuration 53
 - BisSubscribe 137
 - build agents 113, 115
 - build artifacts 113
 - build configuration 116–118
 - build controller 113
 - build details 137
 - Build Explorer 118
 - build layout 113
 - Build Notification 137
 - build queue 113
 - build scripting 117
 - Build Service 54
 - build-agent pool 113
 - checking in 61
 - checking out, branching, merging 62
 - CI builds 116
 - Code Analysis report 207
 - collections 55–57
 - populating with project 57
 - cost-benefit factor 33
 - distributed teams 54
 - distributing building tasks 113
 - email notification 138
 - feedback 135–138
 - LED message board 139
 - using API for extended feedback 140
 - file storage 57
 - installing 53–54
 - Lab Management 57
 - licensing 53, 113
 - locking 63
 - locking and blocking 35
 - messages 62
 - migrating from VSS 53
 - mixing with SVN 64
 - notification 113
 - notification database 137
 - notification subscriptions 137
 - pending changes 62
 - process templates 59
 - Microsoft Solution Framework (MSF) for Agile Software Development 60
 - MSF for CMMI Process Improvement 60
 - Proxy 54
 - reporting services 137
 - retention 118
 - scheduled builds 116
 - server, connecting 58
 - setting up 53–64
 - setting up a server 53–64
 - shelvesets 64
 - shelving 63
 - Single Server
 - configuration 54
 - source control 53–64
 - adding solution to 61
 - managing 61–63
 - SQL Server integration 113
 - team project 55
 - testing on 160–161
 - TFS 2010 vs. VSS 64
 - tray notification 136
 - triggering 116
 - triggers
 - continuous integration trigger 116
 - gated check-in 117
 - manual trigger 116
 - rolling build trigger 116
 - schedule trigger 117
 - turning on Code Analysis 207
 - using SQL Server 53
 - Visual Studio integration 58–60
 - web feedback 137
 - Windows Workflow Foundation 117
 - work items 62
 - XAML 117
- TFS Basic
 - feedback limitations 135
 - reporting services, lack of 137
- TFS Version Control, source control aspects 32
- ThoughtWorks 20
- Time 78
- TortoiseSVN 14
 - commit 48
 - icons 44
 - installing 42
 - resolving conflicts 48
 - reverting, copying, branching 50
 - working with repository 42–53
- Torvalds, Linus, use of Git 33
- .trx file 163
- T-SQL 261
- Typemock Isolator 177
- ## U
-
- UI testing 180, 192
 - integrating into the CI process 189–192
 - Silverlight 182–184
 - tools 180
 - web applications 185–189
 - Windows Forms 180–182
- UI tests in CI
 - reliability 190
 - Selenium 191
 - speed 189
- UIA 180
- unit test
 - as part of automated tests 145
 - design rules 145
 - framework 16
 - integrating with CI server 151
 - mocking 164–170
 - removing, to speed up CI 277
 - separating from integration tests 277
- unit testing
 - as base of testing pyramid 171
 - as white-box testing 168
 - bird's-eye view 145–146
 - error-driven development 146
 - example 146–157
 - stored procedures 271–273
 - with NUnit 149–151
- Urban Code 288
- user-acceptance test 172
- UsingTask 78
- ## V
-
- Value 86
- Vault
 - as centralized system 33
 - cost-benefit factor 33
 - source control aspects 32
- verbosity 75
- version control system. *See* source control
- Version Control with Subversion* 35
- vertical scaling 282
- virtual PC, deployment 259

visibility of project, increasing 9, 12

Visual Basic, comments 228

Visual SourceSafe (VSS). *See* VSS

Visual Studio
and MSBuild 78–83
Code Analysis. *See* Code Analysis
Command Prompt 71
continuous database
integration 266–275
database maintenance, putting into CI 274
generating test data 268
Import Database Wizard 266
project file 79–81
solution file 81–82
starting MSBuild script 82–83

Visual Studio 2010
MS Deploy 254–256

Visual Studio Installer 20

VisualSVN 14

VisualSVN Server
free vs. Enterprise 36
installing 35–37
integrating with Windows user management 36
Management Console 35–37
on Windows 35
repository
creating 37–39
keeping healthy 39–42
location 36
setting up 35–42
SSL 36
users, managing 37

VSS 14
as centralized system 33
blocking 35
cost-benefit factor 33
replaced by TFS Basic 33
source control aspects 32

W

web application
deploying 253
developing under IIS 254

Web Deployment Agent Service 258

Web Deployment Tool 253–258

web package 254

weekly build 7–8

White Framework 169
creating tests 182, 184
installing 180
integrating results with CI server 182
reusing Silverlight tests 184
running with higher privileges 182
testing Silverlight applications 182–184
testing Win32 applications 182
testing Windows Forms 180–182
testing WPF applications 182

XML NUnit report 182

white-box testing 167
definition of 168

Windows Installer XML. *See* WiX

WiX 20, 244–248
automating 247
components 247
creating shortcuts 247
installation directory
definition 246
installing 245–247
MSI package 247
product description 246
project file description 246
registry keys 247
resources 247
setup project 245
toolset 245

UI 247
XML project files 244

WorkingDirectory 74

X

XML build script 68

XML documentation 228–233
<c> tag 232
<code> tag 232
<example> tag 230
<exception> tag 230
<list> tag 232
<param> tag 229
<paramref> tag 230
<remarks> tag 230
<returns> tag 229
<see> tag 230
<summary> tag 229
<table> tag 233
<value> tag 230
comments 229
creating 231
formatting text in 231–233
comments, formatting 231–233
common tags 228–231
cref attribute 230
exceptions 230
formatting with
Sandcastle 233–239
HTML-style tables in 233
lists 232
name attribute 230
table 233

XML file, using for
documentation 228

xmlogger 154

xmlns 72

XP, development process 5

xUnit.net 16

Z

ZipFileName 78

