INDEX

Symbols

(|...|) (active patterns), 173–174
[|...|] (array expressions), 142
<- operator (assignment), 29, 71, 120
* (asterisk)
    multiplication operator, 35
tuple delimiter, 113
@ (at)
    list concatenation operator, 151–152
verbatim string prefix, 38
<< operator (backward function composition), 108, 109
| operator (backward pipelining), 108
&&& operator (bitwise AND), 35
^^^ operator (bitwise exclusive OR), 35
<<< operator (bitwise left shift), 35
~~~ operator (bitwise negation), 35
||| operator (bitwise OR), 35
>> operator (bitwise right shift), 35
(*...*) (block comments), 60
&& operator (Boolean AND), 34
|| operator (Boolean OR), 34
:: operator (cons), 151
- (dash)
    set difference operator, 154
    subtraction operator, 35
    unary negative operator, 35
/ operator (division), 35
?: operator (dynamic cast), 83
// (end-of-line comments), 60
= operator (equality), 35, 36
** operator (exponent), 35
>> operator (forward function composition), 108, 109
|> operator (forward pipelining), 42, 79, 107, 108
;; (FSI expression terminator), 14
> operator (greater than), 35
>= operator (greater than or equal to), 35
<> operator (inequality), 35
< operator (less than), 35
<= operator (less than or equal to), 35
#light directive, 6
% operator (modulus), 35
?? operator (null coalescing, C#), 42
? (optional parameter prefix), 41, 75
+ (plus)
    addition operator, 35
    set union operator, 153
    string concatenation operator, 38
    unary positive operator, 35
~ (prefix operator), 94
``...`` (quoted identifier delimiter), 33
.. operator (range expression), 135–136
:= operator (reference cell assignment), 30
! operator (reference cell dereferencing), 30
-> (right arrow)
    function value, 105, 112
    sequence expressions, 135
- (set difference operator), 154
[... ] (square brackets)
    indexed properties, 71
    list expressions, 149
^ (statically resolved type parameters), 49, 52
:: operator (reference cell assignment), 30
? operator (static cast), 82
^ operator (string concatenation, ML style), 38
@...@ (strongly typed quoted literal), 190, 194
() (unit value), 42
| (vertical bar)
pattern matching delimiter, 11, 160
union case delimiter, 127
@@...@@ (weakly typed quoted literal), 190, 193, 194
_ (Wildcard patterns), 115
// (XML comment), 60–61

A
abstract
classes, 84–86
keyword, 85
members
methods, 86
properties, 85–86
AbstractClassAttribute, 85
access modifiers, 66
internal, 66, 70
private, 66, 69, 70
protected, 66
public, 66, 70
accessor (property), 69
Action (delegate), 105
active patterns ([...[]])
defined, 173–174
parameterized, 176
partial, 174–175
active recognizer functions, 173
add function (Event module), 78
additional constructors, 66–67
addition operator (+), 35
agent-based programming, 250–255
counting queued messages, 252
receiving messages, 251
replying to messages, 252–253
scanning messages, 251–252
sending messages, 251
starting agents, 251
Agent<'T> type alias. See
MailboxProcessor<'T> class
AggregateException class, 240, 241
Flatten method, 241
Handle method, 240, 241
InnerExceptions property, 240, 241
all operator (query expressions), 215
AllowNullLiteralAttribute, 41
and keyword, 111
mutually recursive functions, 111
mutual recursion, 91
property accessors, 70
AND patterns, 171–172
antecedents, defined, 237
Apache 2.0 license, 2
ArgumentException, 56
arguments, named, 74
array
expressions, 142–143
keyword, 143
Array2D module, 147
array2D operator, 147
Array3D module, 147
ArrayList class, 49, 133–134
Array module
copy function, 145
empty function, 143
get function, 145
init function, 144
set function, 145
sortInPlaceBy function, 146
sortInPlace function, 145–146
sortInPlaceWith function, 146
zeroCreate function, 144
Array patterns, 168
arrays, 133, 142–149
accessing elements, 144–145
copying, 145
defined, 142
empty, 143
initializing, 144
jagged, 148–149
multidimensional, 147–148
slicing, 145, 149
sorting, 145–147
as keyword (self-identifiers), 67–68
As patterns, 171
assignment operator (-), 29, 71, 120
Async class
AwaitTask method, 248–249
CancelDefaultToken method, 245–246
Catch method, 247–248
Parallel method, 245
RunSynchronously method,
243–245, 248
StartAsTask method, 248–249
StartImmediate method, 243
Start method, 243–244,
246–247, 249
StartWithContinuations method,
243, 244
TryCancelled method, 246
asynchronous programming model, 230
asynchronous workflows, 241–250
cancelling, 245–247
defined, 241
exception handling, 247–248
let! keyword, 242, 249, 250
return! keyword, 242, 249–250
with TPL, 248–250
use! keyword, 242
async modifier (C#), 249
AsyncReplyChannel<'T> class, 252–255
Async<'T> class, 242
automatic generalization, generics, 49
automatic properties, 70–71
AutoOpenAttribute (modules), 8
averageByNullable operator (query expressions), 214
averageBy operator (query expressions), 213
await operator (C#), 249
backward function composition operator (<<), 108, 109
backward pipelining
defined, 108
operator (<<), 108
banana clips, 173
base implementations, calling, 84
base keyword (inheritance), 84
bigint data type, 35
bindings
do, 33
let, 28
use, 30
Bind method (computation expressions), 259
bitwise operators
AND (&&&), 35
left shift (<<<), 35
negation (~~~), 35
OR, exclusive (^^^), 35
OR, non-exclusive (|||), 35
right shift (>>>, 35
block comments ((*...*)), 60
Boolean
data type, 34
operators
AND (&&&), 34
OR (|||), 34
values, 34
branching, 47–48
building strings example (computation expressions), 264–269
byte data type, 34
C
Call active pattern, 196
callbacks, defined, 237
CancellationTokenSource class, 234, 239, 246–247
casting, 82–83
cdata type, 37
Choice union type, 248
classes, 64–80
CLIEventAttribute, 77, 80, 189
CLIMutableAttribute, 121
closures, 29, 112
code quotation. See quoted expressions
collections, enumerable, 134
collection types, converting between, 157–158
Combine method (computation expressions), 260, 262, 263, 266–267
CommandLineArgs property, 22
comments, 59–61
block, 60
d-end-of-line, 60
XML, 60–61
computation expressions
anatomy of, 258–260
builder classes, 258–259
computation type, 258
defined, 257–258
desugaring, 259
concat function (string extension), 38
collection types, converting between, 157–158
continuations, defined, 237–239

The Book of F# © 2014 by Dave Fancher
conversions, numeric, 36
count operator (query expressions), 213
currying, 106–109
custom exceptions, 56–58

D
data parallelism, 230, 231–234
data types, built-in
  bigint, 35
  Boolean, 34
  byte, 34
  double, 34
  float, 34
  float32, 35
  int, 34
  int8, 34
  int16, 34
  int32, 34
  int64, 34
  nativeint, 35
  sbyte, 34
  single, 35
  string, 37
  uint, 34
  uint16, 34
  uint32, 34
  uint64, 34
  unativeint, 35
  unit, 42, 104

decimal data type, 34
defaultArg function, 41–42, 75
default constructor, 64
default indexed property, 71–72
default keyword (inheritance), 87
DefaultValueAttribute, 67–69
Delay method (computation expressions), 260, 262, 263, 267
DerivedPatterns module (quoted expressions), 194
discriminated unions, 41, 47, 122–130
  additional members, 129–130
  defined, 122
  as object hierarchies, 124–126
  self-referencing, 126
  single case, 127
  as tree structures, 126–127
  as type abbreviations, 127–129
distinct operator (query expressions), 206
division operator (/), 35
dot keyword (computation expressions), 259
do bindings, 33
double data type, 34
downcasting, 83
downto keyword (simple for loops), 46
dynamic cast operator (?:), 83
Dynamic Type-Test patterns, 170

E
eager evaluation, 130
elif keyword, 47–48
end-of-line comments (//), 60
entry point, 9
EntryPointAttribute, 9
enumerable collections, 134
Enumerable.Range method, 136
enumerations, 43–45
  changing base type, 43
  defined, 43
  FlagsAttribute, 43–45
  reconstructing
    enum function, 45
    EnumOfValue function, 45
equality operator (=), 35, 36
escape sequences, 37
Event module, 78, 80
  add function, 78
  filter function, 78
  map function, 80
  pairwise function, 78
  partition function, 78
events, 77–80
  custom, 79–80
  observing, 78–79
Event<'T> class, 77
  Publish property, 79
  Trigger function, 79
exactlyOne operator (query expressions), 208
exactlyOneOrDefault operator (query expressions), 208
Exception class, 53, 56
exception keyword, 56
exceptions, 53
  custom, 56–58
  handling, 53–55
  raising, 55–56
  reraising, 54
  try...finally, 55
exists operator (query expressions), 213, 215
exn type abbreviation, 53
explicit properties, 69–70
exponent operator (**), 35
expressions, 8–9
expression trees, 187, 188–190
Expr<'T> type, 190
Expr type, 190, 191, 192, 194
ExtCore project, 269
ExtensionAttribute, 100, 105
extension methods (C# and Visual Basic), 99

F
F# for Fun and Profit, 269

F# Interactive
  defined, 13
directives
    #help, 16
    #I, 17, 21
    #load, 16–17, 21
    #quit, 16
    #r, 17, 21
    #time, 17–18
expression terminator (;;), 14
fsi.exe, 13
it identifier, 15
options
  --, 22
  --define, 21
  --exec, 22
  -I, 21
  --lib, 21
  --load, 20–21
  --optimize, 23
  --quiet, 22–23
  -r, 21
  --reference, 21
  --tailcalls, 23
  --use, 21
  in Visual Studio, 20
reset interactive session, 16
timing, 17
val (output), 15
Visual Studio window, 13–14

F# Software Foundation, 2

Factory pattern, 90
FailureException, 56
failwithf function, 56
failwith function, 56
fields, 68–69
  explicit, 68–69
  let bindings, 68

file extensions
  .fs, 18
  .fsx, 18
FileNotFoundException, 54
filter function (Event module), 78
find operator (query expressions), 208
FizzBuzz example
  active patterns, 173–174
  computation expressions, 261–264
  partial active patterns, 174–175
FlagAttribute enumerations, 43–45
flexible types, 52
float32 data type, 35
float data type, 34
flow control, 45–48
  for loops, 46–47
  if...then expressions, 47–48
  while loops, 46
for each loop (C#), 46
for loops, 46–47
for method (computation expressions), 259, 263
forward function composition operator (>, 108, 109
forward pipelining, 107–108
forward pipelining operator (|>, 42, 79, 107, 108
FSharpFunc (delegate), 105, 112
FSharpFuncUtil class, 105
FSharpList<'T> class, 149
.fs files, 18
FSI. See F# Interactive
fsi.exe, 13
fst function, 114
.fsx files, 18
Func (delegate), 105
function
  composition, 108–109
  expressions, 78, 112
  keyword, 161
  values, 105
functional purity, 27–28
functions, higher-order, 105

G
generic measures, 184
generics
  constraints, 50–52
  comparison, 52
default constructor, 51
defined, 50
delegate, 51
generics: constraints (continued)
  enumeration, 51
  equality, 52
  member, 51
  nullness, 50
  reference type, 51
  subtype, 50
  unmanaged, 51
  value type, 51
defined, 48
generalization
  automatic, 49
  explicit, 50
type parameters, statically resolved, 49, 52
Wildcard pattern, 52
GetCommandLineArgs method, 22
GetEnumerator method, 134
GetSlice method, 76–77
global keyword (namespaces), 7
greater than operator (>), 35
greater than or equal to operator (>=), 35
groupBy operator (query expressions), 210–211
groupJoin operator (query expressions), 216–217
groupValBy operator (query expressions), 211

H
handling exceptions, 53–55
HasFlag method (System.Enum), 44–45
Hashable interface, 134
head operator (query expressions), 207
headOrDefault operator (query expressions), 207
higher-order functions, 105

I
IComparable<'T> interface, 139
Identifier patterns, 128, 129, 163, 164–165
identifiers, quoted, 33
IDisposable interface, 30, 92, 93
IEnumerable interface, 134
IEnumerable<'T> interface, 46, 134, 149
if...then expressions, 47–48
ignore function, 42
immutability, 26–28
implicit properties, 70–71
indexed properties
  one-dimensional, 71
  multidimensional, 72
inequality operator (<>), 35
inheritance, 81–88
inherit keyword, 82, 94
initializing properties, 72–73
instance methods, 73
int8 data type, 34
int16 data type, 34
Int32 active pattern, 196
int32 data type, 34
int64 data type, 34
int data type, 34
interface keyword, 93
interfaces, 91–94
  defining, 93–94
  implementing, 92–93
  inheritance, 94
  marker, 93
internal access modifier, 66, 70
International System of Units, 178
invalidArg function, 56
InvalidCastException, 83
IEqualityComparer interface, 115
Item property (indexed properties), 71
it identifier (F# Interactive), 15

J
join operator (query expressions), 216

L
lambda expressions, 78, 112
LanguagePrimitives module, 180
last operator (query expressions), 207
lastOrDefault operator (query expressions), 207
lazy evaluation, 130–131
lazy keyword, 130
Lazy<'T> class, 130
less than operator (<), 35
less than or equal to operator (<=), 35
let! keyword
  asynchronous workflows, 242
  computation expressions, 259
let keyword, 28
license, Apache 2.0, 2
lightweight syntax, 6
LINQ, 49, 76, 99, 104, 187, 201
list comprehensions. See sequence expressions
list concatenation operator (@), 151–152

List module
- append function, 152
- concat function, 152
- contains function, 151
- empty function, 150
- exists function, 151
- head function, 150
- nth function, 150
- tail function, 150

List patterns, 168–169
- lists, 149–152
  - accessing elements, 150
  - combining, 151–152
  - creating, 149–150
  - defined, 149
  - head, 150
  - tail, 150
- List<'T> class, 49, 134, 149
- LiteralAttribute, 28, 164, 165
- Literal patterns, 165
- literals, 28
- locking, 232
- loops, 45–47
  - for, enumerable, 46
  - for, simple, 46
  - while, 46

M

MailboxProcessor<'T> class, 250
- CurrentQueueLength property, 252
- PostAndReply method, 253
- Post method, 251, 252
- Receive method, 251, 252
- Reply method, 253
- Scan method, 251
- Start method, 250–251
- main method. See entry point
- map function (Event module), 80
- Map<'Key, 'Value> class, 155
- Map module
  - containsKey function, 156
  - find function, 156
  - findKey function, 157
  - tryFind function, 157
  - tryFindKey function, 157
- maps, 155–157
  - creating, 156
  - defined, 155
  - finding keys, 157
  - finding values, 156–157
- marker interfaces, 93

Mars Climate Orbiter, 177
- match expressions, 127, 159–162
  - defined, 159–160
  - exhaustive matching, 162–163
  - guard clauses, 160–161
  - pattern matching functions, 161–162
- MatchFailureException, 162
- maxByNullable operator (query expressions), 214
- maxBy operator (query expressions), 213
- MeasureAttribute, 178
- measures. See units of measure
- member keyword, 69, 73
- member val keyword pair (implicit properties), 70
- metaprogramming, 187
- method accessibility, 73–74
- methods, 73–77
  - instance, 73
  - overloaded, 75
- minByNullable operator (query expressions), 214
- minBy operator (query expressions), 213
- ML programming language, 1
- modules, 7–8
  - declaring, 7
  - defined, 7
  - local, 7
  - opening, 8
  - top-level, 7
- modulus operator (%), 35
- monads, 257. See also computation expressions
- multiplication operator (*), 35
- mutability, 29
- mutable bindings, 29
- mutable keyword, 29, 120, 121
- mutual recursion
  - between functions, 111
  - between types, 91

N
- named arguments, 74
- namespace keyword, 6
- namespaces, 6–7
  - declaring, 6
  - global, 7
  - nesting, 6
  - opening, 7
- nativeint data type, 35
new keyword
  additional constructors, 66
  instance creation, 65
  object expressions, 98, 99
not (Boolean operator), 34
nth operator (query expressions), 207
nullability, 41–42
nullable operators, 205–206
null keyword, 41
Null patterns, 165–166
numeric data types, 34–35

O

object expressions, 97–99
OCaml programming language, 1
of keyword (discriminated unions), 122
open keyword
  modules, 8
  namespaces, 7
OperationCanceledException, 233–234, 240
operators
  custom, 94–97
  global, 96–97
  infix, 95–96
  new, 96
  overloading, 94
  prefix, 94–95
optional parameters, 75
Option Infer (Visual Basic), 39
option keyword, 41
Option<'T> type, 41, 75, 122
  defined, 122
  introduced, 41
  None, 41, 122–123
  Some<'T>, 41, 122–123
optional parameter prefix (?), 41, 75
optional parameters, 41, 75
OR patterns, 172
out parameters, 116
overloading
  methods, 75
  operators, 94
Overridable modifier (Visual Basic), 87
override keyword (inheritance), 83–84
overriding members, 83–84

P

pairwise function (Event module), 78
Parallel class
  ForEach method, 231
  For method, 231–232, 233
  Invoke method, 234–235
Parallel LINQ, 231
parallel loops
  cancelling, 233–234
  short-circuiting, 233
ParallelLoopState class, 233
  Break method, 233
  Stop method, 233
ParallelOptions class, 234
parallel programming, 230
parameters, optional, 75
partial active patterns, 174–175
partial application, 106
pattern matching
  active patterns, 173–174
  AND patterns, 171–172
  Array patterns, 168
  As patterns, 171
  Cons patterns, 169
  Constant patterns, 164
  Dynamic Type-Test patterns, 170
  and exception handling, 53–55
  Identifier patterns, 128, 129, 163, 164–165
  List patterns, 168–169
  Literal patterns, 165
  Null patterns, 165–166
  OR patterns, 172
  parentheses, use of, 172–173
  partial active patterns, 174–175
  Record patterns, 167–168
  Singleton pattern, 66
  Tuple patterns, 114, 166–167
  Type-Annotated patterns, 169–170
  Union Case patterns, 164–165
  Variable patterns, 163
  Wildcard patterns, 115, 163
pattern matching delimiter (|), 11, 160
pattern-matching functions, 161–162
Pattern module (quoted expressions), 194
pipelining, 107–108
  backward, 108
  defined, 107
  forward, 107–108
  noncurried functions, 108
PLINQ, 231
potential parallelism, 231
prefix operator (~), 94
primary constructor, 65–66
printf function, 58
printfn function, 58
private access modifier, 66, 69, 70
ProjectionParameterAttribute, 219–220
project templates, 2–4
properties, 69–73
  automatic, 70–71
  explicit, 69–70
  implicit, 70–71
  indexed
    one-dimensional, 71
    multidimensional, 72
initializing, 72–73
PropertyGet active pattern, 198
protected access modifier, 66
public access modifier, 66, 70
Publish property (events), 79
purity, functional, 27–28

Q
query expressions
  aggregating data, 213–214
  defined, 201–202
  detecting items, 214–215
  distinct values, 206
  extending, 219–221
  filtering data, 204–206
  finding arbitrary items, 207–208
  first or last item, 207
  grouping, 210–211
  joining data sources, 215–219
  pagination, 211–213
  projecting data, 203–204
  sorting, 209–210
quoted expressions
  creating through reflection, 191–192
  decomposing, 194–199
  defined, 187
  manual composition, 192–193
  quoted literals, 190–191
  splicing, 194
  strongly typed, 190
  weakly typed, 190
quoted identifier delimiter (`...`), 33

R
raise function, 55, 56
raising exceptions, 55–56
range expression operator (..), 135, 136
range expressions, 135–136
read-evaluate-print loop (REPL), 13
readonly keyword (C#), 28
rec keyword (recursive functions), 109
record expressions
  copy and update, 120
  defined, 118
  new records, 118–119
Record patterns, 167–168
record types, 118–122
  additional members, 121–122
  copying, 120
  creating, 118–119
  defined, 118
  mutability, 120–121
  naming conflicts, 119–120
recursion
  defined, 109
  tail-call, 110–111
reference cell assignment operator (:=), 30
reference cell dereferencing operator (!), 30
reference cells, 29–30
referential transparency, 104
ReflectedDefinitionAttribute, 191, 192, 194
ref operator, 29
REPL (read-evaluate-print loop), 13
reraise function, 54
ResizeArray<'T> type abbreviation, 149
ReturnFrom method (computation expressions), 259
return! keyword
  asynchronous workflows, 242
  computation expressions, 259
return keyword, 104
Return method (computation expressions), 259
return values, 9–10
Run method (computation expressions), 260

S
sbyte data type, 34
scripting
  command-line arguments, 22
  with F# Interactive, 18–19
SealedAttribute, 87, 88
sealed classes, 87–88
Select Case statement (Visual Basic), 127, 160
select operator (query expressions), 203–204
self-identifiers in constructors, 67–68
Seq module
- averageBy function, 142
- average function, 141
- empty function, 136–137
- filter function, 140
- fold function, 140–141
- isEmpty function, 138
- iter function, 139
- length property, 138
- map function, 139
- reduce function, 141
- sortBy function, 140
- sort function, 139
- sumBy function, 142
- sum function, 141

seq<'T> type abbreviation, 134
sequence expressions, 134–135
defined, 134
yielding results, 135

sequences, 46, 134–142
- aggregating, 140–142
- defined, 134
- empty, 136–137
- filtering, 140
- initializing, 137
- iterating over, 139
- length of, 138–139
- sorting, 139–140
- transforming, 139

set difference operator (-), 154
Set module
- difference function, 154
- intersect function, 154
- isProperSubset function, 154
- isProperSuperset function, 154
- isSubset function, 154
- isSuperset function, 154
- union function, 153

sets, 152–155
- creating, 152–153
- defined, 152
- differences, 154
- intersections, 154
- subsets and supersets, 154
- unions, 153

Set<'T> class, 153
set union operator (+), 153
ShapeCombination active pattern, 195, 196
ShapeLambda active pattern, 195, 196
ShapeVar active pattern, 195, 196

side effects, 26–27

single data type, 35
Singleton pattern, 66
SI units, 178
Skip extension method, 76
skip operator (query expressions), 211
skipWhile operator (query expressions), 211
slice expressions, 76–77, 145, 147–149
snd function, 114
sortByDescending operator (query expressions), 209
sortByNullableDescending operator (query expressions), 209
sortByNullable operator (query expressions), 209
sortBy operator (query expressions), 209
SpecificCall active pattern, 196
sprintf function, 58
statically resolved type parameters (^), 49, 52
static cast operator (>::), 82
static class, 88
static keyword, 88, 89
static members
- constructors, 88–89
- fields, 89
- initializers, 88–89
- methods, 90–91
- properties, 89–90
string concatenation operator (+), 38
string concatenation operator, ML style (^), 38
String class
- Concat method, 38
- Format method, 58
- Join method, 38
- Split method, 71

string data type, 37
strings, 37
- concatenation, 38
- formatting, 58
- literal, 37
- triple-quoted, 38
- verbatim, 38
StructAttribute, 80
structs, 80–81
structures, 80–81
subtraction operator (-), 35
sumByNullable operator (query expressions), 214
sumBy operator (query expressions), 213
switch statement (C#), 127, 160
symbols
  COMPILED, 21
  DEBUG, 21
  INTERACTIVE, 21
  RELEASE, 21
SyncLock statement (Visual Basic), 232
syntactic tuples, 115–116
T
  tail-call recursion, 110–111
Take extension method, 76
take operator (query expressions), 211–212
takeWhile operator (query expressions), 211
Task class
  constructor, 235
  ContinueWith method, 237, 238
  Factory property, 235
  Start method, 235
  WaitAll method, 237
  WaitAny method, 237
TaskFactory class, 235, 236, 238
  ContinueWhenAll method, 238, 239
  ContinueWhenAny method, 238, 239
  StartNew method, 235
  StartNew<'T> method, 236
  Wait method, 236
task parallelism, 230–231, 234–241
Task Parallel Library (TPL), 230–241, 249
tasks
  cancelling, 239–240
  continuations, 237–239
  creating and starting, 234–235
  exception handling, 240–241
  returning values from, 235–236
  waiting for completion, 236–237
Task<'T> class, 235, 236, 238
Task<'T>.Result property, 236
templates, project, 2–4
thenByDescending operator (query expressions), 210
thenByNullableDescending operator (query expressions), 210
thenByNullable operator (query expressions), 210
thenBy operator (query expressions), 210
then keyword, constructors, 67
timing (F# Interactive), 17
TPL (Task Parallel Library), 230–241, 249
ToFSharpFunc method, 105
tokens, string formatting, 58
to keyword (simple for loops), 46
Trigger function (events), 79
triple-quoted strings, 38
try...finally expressions, 53, 55
TryFinally method (computation expressions), 259
TryGetReflectedDefinition method, 191
try...with expressions, 53
TryWith method (computation expressions), 259
tuple delimiter (•), 113
Tuples patterns, 114, 166–167
  for out parameters, 116–117
  syntactic, 115–116
type abbreviations, 33, 59
Type-Annotated patterns, 169–170
type annotations
  defined, 40
  with units of measure, 181
type augmentations. See type extensions
type extensions, 99–100
  intrinsic, 99
  optional, 99
type functions, 137
type inference, 34, 39
type keyword, 59, 118, 123
  classes, 64
  interfaces, 93
type providers, 221–228
  available providers, 222–223
  defined, 221
  security warning, 224
U
uint16 data type, 34
uint32 data type, 34
uint64 data type, 34
uint data type, 34
unary negative operator (-), 35
unary positive operator (+), 35
unativeint data type, 35
Unicode, 37
union case delimiter (•), 127
Union Case patterns, 164–165
unit data type, 42, 104
unit value (()), 42
units of measure
  applying, 179–180
  conversions, 182–183
  defined, 178
  enforcing, 181
  formulas, 178–179
  generic, 184
  measure annotations, 179–180
  measure-aware types, 184–185
  ranges, 182
  stripping, 180–181
upcasting, 82–83
use bindings
  defined, 30
  within modules, 31
use! keyword
  asynchronous workflows, 242
  computation expressions, 259
use keyword
  computation expressions, 259
  defined, 30
using directive (C#), 59
using function
  C# implementation, 32
  defined, 31
Using method (computation expressions), 259
using statement (C#), 30

V
val
  F# Interactive, 15
  keyword (explicit fields), 67, 68, 81
Value active pattern, 196, 198
value types, 81
Variable patterns, 163
variables, 27
var keyword (C#), 39
var type, 190
verbatim string prefix (@), 38
verbatim strings, 38
verbose syntax, 6
virtual members, 84, 87
virtual modifier (C#), 87
void type (C#), 42

W
where operator (query expressions), 204
while loops, 46
While method (computation expressions), 259
whitespace, significance of, 5–6
Wildcard patterns (_)
  exception handling, 53
  generics, 52
  defined, 115, 163
with keyword
  object expressions, 98
  property accessors, 70
  type extensions, 100
workflows. See asynchronous workflows;
  computation expressions

X
XML comments (///), 60–61

Y
YieldFrom method (computation expressions), 259, 266
yield! keyword (computation expressions), 259, 266
yield keyword
  computation expressions, 259
  defined, 135
Yield method (computation expressions), 259, 261, 263, 266

Z
Zero method (computation expressions), 260, 266