

## Symbols

.NET, 349

## A

adaptability, 10  
adapter pattern, 78  
adaptors, 255  
additive instruments, 379, 381  
additive monotonic instruments, 379  
administrative processes, 188  
Amazon, 5, 265  
Amazon EC2 (Elastic Computer Cloud), 5  
Amazon Elastic Computer Cloud (EC2), 5  
Amazon S3, 217  
anonymous functions, 56  
anonymous struct, 122  
Apache Thrift, 225, 230  
API gateway, 216  
append built-in function, 39  
append-only log file, 126  
application state, 7, 124, 196  
arrays, 36-37  
asynchronous instruments, 379, 383-384  
attributes, 352, 358  
Auth0, 216  
autoinstrumentation, 359-362  
autoscaling, 191, 196, 293-294  
availability, 170, 305  
AWS Lambda, 213

## B

backing services, 182  
backoff, 78  
(see also exponential backoff)

backoff algorithms, 277-280  
Berners-Lee, Timothy John, 287  
Blackbox Exporter, 378  
blank identifier, 35, 46  
Booleans, 30  
bottlenecks, 194  
bugs, 9, 24, 206  
(see also faults, memory leaks)  
build stage, 183  
built-in functions  
    append built-in function, 39  
    cap built-in function, 37  
    close built-in function, 66  
    delete built-in function, 41  
    len built-in function, 36-41  
    make built-in function, 38, 41  
    panic built-in function, 51  
byte, 31, 40-41

## C

C (language), 22, 135  
C++, 22-23, 24, 135, 206  
cache, 205  
caching, 198-201  
callback function, 384  
Canonical, 321  
cap built-in function, 37  
cardinality, 370  
cascading failures, 269  
case expressions, 48  
certificate authority, 147  
chan keyword, 65  
channels  
    buffered, 65, 67, 204

- close operation, 66
- features of, 201
- looping, 67
- timeouts, 68
- unbuffered, 65
- Cheney, Dave, 207
- circuit breaker pattern, 77-80, 81, 280
- close built-in function, 66
- closure, 56
- Cloud Computing, 5
- cloud native
  - attributes, 6
  - definition of, 3-4, 6
  - evolution of, 12-13, 343
  - purpose of, 168
- Cloud Native Computing Foundation, 6, 230, 343, 347, 355, 373
- CNCF (see Cloud Native Computing Foundation)
- COBRA, 230
- Cobra package, 310-313
- CockroachDB, 310
- code divergence, 186
- codebase, 179
- Codewalk "Share Memory by Communicating", 203
- cold start, 214
- command-line flags, 314
- Communicating Sequential Processes (see CSP)
- complex numbers, 31
- complexity, 205
- composition, 17, 61
- comprehensibility, 18-19
- concurrency
  - computational, 20
  - CSP-style, 19
  - levels of, 200
  - patterns, 92
    - (see also fan-in pattern, fan-out pattern, future pattern, sharding)
  - primitives, 201
    - (see also channels, goroutine)
  - safe, 121-123
- configurable flag, 336-337
- configuration, 180-182, 305, 306-333, 334
- configuration files, 314-317
- Console Exporter, 354, 368
- constants, 35
- Consul, 182, 329
- container ID, 154
- container image, 151, 154
- container orchestration system, 7, 24
  - (see also Kubernetes)
- containers, 36-42, 150-151, 156, 162, 186
  - (see also multiple containers)
- context deadlines, 74
- context package, 72-76
- context timeouts, 74, 281
- context values, 75
- contract, 221, 224
- coupling, 219-221
  - (see also loose coupling, tight coupling)
- CPU, 194
- CSP, 19, 167
- Cummins, Holly, 168

## D

- DAG (directed acyclic graph), 350
- data isolation, 184, 185
- data labels, 382
- database driver, 139, 142
- debounce pattern, 80-84, 87
- declarative methods, 109
- deep health checks, 296, 298-300
- defer keyword, 52-53
- degrees of freedom, 177
- Delays (see future pattern)
- delete built-in function, 41
- dependability, 169-174, 305
- dependability procurement, 173
- dependability validation, 173
- dependencies, 179-180, 298
  - (see also deep health checks)
- dependency relationship, 5
- dereferencing, 43, 58
- derived context, 74
- development/production parity, 186
- digital certificate, 146
- directed acyclic graph (DAG), 350
- disk I/O, 195
- disposability, 186
- distributed monolith, 8, 175, 219, 222
- distributed tracing, 346
- Docker, 150, 151-162, 310
- Docker Hub, 153
- Dockerfile, 151-162
- dot com gold rush, 4
- double-quote style string literals, 32

- downstream dependencies, 5
  - (see also transitive downstream dependencies)
- durability, 197
- dynamic analysis, 177
- dynamic feature flag, 337-341
- dynamic sampling, 391, 397
- DynamoDB, 265

## E

- Effector function, 84
- Elasticsearch, 187, 346
- element type, 65
- ELK, 187, 389
- embedded pointers, 63
- empty interface, 61
- envfile, 329
- environment agnosticism, 185
- environment variable
  - configuration files and, 314
  - example of, 10
  - Go and, 308
  - uses of, 181, 307
  - Viper and, 329, 330
- Envoy, 223
- Erlang, 349
- error detection, 176
- error handling, 49-50
- error, definition of, 9
- errors, 49, 268
  - (see also error handling)
- etcd, 182, 314, 329, 332
- events, 352, 358, 387
- Executable Linkable Format, 245
- exponential backoff, 80, 277
- exporters, 353
  - (see also tracing exporters)
- exposing ports, 154, 156
- extending standard composite types, 59

## F

- FaaS, 185, 213, 214, 215, 217
- Failure Mode and Effects Analysis, 178
- failure rate, 170
- failures
  - causes of, 266, 345
  - definition of, 9, 267-268
  - types of, 269
    - (see also cascading failures)

- Fallacies of Distributed Computing, 90, 212
- fan-in pattern, 93-95
- fan-out pattern, 95-97
- fault forecasting, 173, 178
- fault masking, 290-291
- fault prevention, 172-174
- fault removal, 173, 176-177
- fault tolerance, 8, 173, 176
- faults, 9, 268
- feature flagging, 334-335
- feature flags, 177
- feature gating, 334
- feature toggling (see feature flagging)
- Fibonacci service, 363-365, 368
- field number, 233
- Fielding, Roy, 287
- first-class values, 55
- floating point, 31
- for statement (see loops)
- format string, 34
- fsnotify package, 328, 332
- function-first implementation, 81-82
- function-last implementation, 81-82
- functional partitioning, 193-194
- functions, 73-101
  - anonymous functions, 56
  - recursive functions, 51
  - variadic functions, 39, 50-55
- functions as a service (see FaaS)
- future pattern, 97-101

## G

- Ganglia, 371
- garbage-collected language, 22, 206
- GCP Cloud Functions, 213
- GET function, 226-229
- GetContext function, 284
- global tracer provider, 356
- Go
  - basic data types, 30-36
    - blank identifier, 35, 46
  - Booleans, 30
  - complex numbers, 31
  - constants, 35
  - short variable declarations, 33
  - simple numbers, 31
  - strings, 30, 32, 40-41
  - variables, 32, 42-44
  - zero values, 33

- channels, 64-67, 203
- composition approach, 57-64
- container types, 36-42
  - (see also arrays, maps, slices)
- control structures, 44-49
- design of, 15-16, 19, 20-25, 177
- features of, 17-18, 19, 64, 349
- formatting, 34, 135
- functions, 50-57
- libraries, 112, 137, 226
- log package, 391-394
- maps, 121-123
- modules, 115, 180
- plug-ins, 241-255, 260
- protocol buffers, 231-232
- proverbs, 19, 201, 307
- select statements, 67-68
- tools of, 181
- Go 1, 21
- Go 2, 21
- go keyword, 64
- Go language (see Go)
- Go-YAML, 321-323
- Google, 15, 20, 221, 230
- Gorilla, 114, 116-117
- goroutine, 64, 65, 92, 96, 197, 203, 206
  - (see also leaking goroutine)
- graceful degradation, 275
- Grafana, 386
- Graphite, 371
- GraphQL, 224, 225
- grouping instruments, 379
- gRPC, 221, 225, 230, 285, 361-362
  - (see also gRPC interceptors)
- gRPC interceptors, 361-362
- gRPC Remote Procedure Calls (see gRPC)

## H

- handler, 113
- handshake configuration, 250, 254
- hard-coded feature flag, 335-336
- hash table, 41
- HashiCorp, 198, 223, 247-249, 251-254
- HashiCorp Consul, 314, 332
- HCL, 329
- HEAD method, 226-229
- health checks, 294-300
  - deep health checks, 296, 298-300
  - liveness checks, 295-297

- shallow health checks, 296, 297-298
- Helm, 310
- Heroku, 178
- hexagonal architecture, 255-264
- Hoare Logic, 167
- Hoare, Tony, 19, 167
- Honeycomb, 349
- horizontal sharding, 101
- horizontally scaling, 7, 185, 193, 194
- host.docker.internal, 385
- HTTP, 224, 226-228
- HTTP/1.1 standard, 287

## I

- IaaS, 5, 13
- idempotence, 108-110, 286-289
- idempotency, 168
- IDL (Interface Definition Language), 231
- if Statement, 47
- imaginary literal, 31
- incrementor function, 56
- Infrastructure as a service (see IaaS)
- inheritance, 17-18
- INI, 329
- instruments (see metric instruments)
- integer types, 31
- Interface Definition Language (IDL), 231
- interface embedding, 61-62
- interfaces, 59-60
- interpreted string literals, 32
- inversion of control, 255
- iota, 128-129
- Istio, 223, 310

## J

- Jaeger, 349, 355, 366, 368
- Jaeger Exporter, 354-355
- Java, 16-18, 22-23, 24, 41, 135, 349
- Java Properties, 329
- Java RMI, 230
- JavaScript, 80, 81, 315, 349
- JavaScript Object Notation (see JSON)
- jitter, 279-280
- JMX Exporter, 378
- JSON, 231, 315-321, 329, 354, 367
- json.Marshal function, 316-321
- json.Unmarshal function, 317-318

## K

key, 125  
key pair, 146  
key-value store, 108, 256  
Kibana (see ELK)  
Kubernetes  
    applications for, 7, 314  
    features of, 24, 151, 162, 293, 306, 310

## L

labels, 369  
Laprie, Jean-Claude, 169  
latency, 205  
LaunchDarkly, 341  
leaking goroutine, 206-207  
Least Recently Used cache (see LRU cache)  
len built-in function, 36-41  
Lightstep, 349  
linguistic stability, 21  
Linkerd, 223  
Linux binaries, 159  
Linux Foundation, 6  
ListenAndServe, 113, 148  
liveness checks, 295-297  
load shedding, 270, 274  
lock contention, 101, 205  
locks, 102, 122  
    (see also read locks, write locks)  
log level, 390  
logging, 346, 387-399  
    (see also Zap logging)  
logging packages, 391  
    (see also Go log package, Zap logging)  
Logrus, 394  
logs, 187, 346, 387, 389  
    (see also stream of events)  
Logstash, 187  
looking up plug-ins, 243  
loops, 44-46  
loose coupling  
    definition of, 7-8, 175, 220-221  
    hexagonal architecture and, 255-264  
    plug-ins, 241-255  
LRU cache, 198-201

## M

maintainability, 11, 171, 304  
make built-in function, 38, 41

make function, 65  
manageability  
    categories of, 305  
        (see also configuration)  
    contributions to, 173  
    definition of, 10, 303-306  
    designs for, 177  
    maintainability and, 11, 304  
map literals, 42  
maps, 36, 41-42, 121-123  
matchers, 116  
mean time between failures (MTBF), 170  
Means of Dependability pyramid, 173  
memory, 195  
memory leaks, 206-209  
Message Transfer System (MTS), 221  
messaging redundancy, 290  
Meter, 374-379  
meter provider, 376  
methods, 58-59  
metric collection, 371  
    pull-based metric collection, 372  
    push-based metric collection, 371  
metric instruments, 379  
    additive instruments, 379, 381  
    additive monotonic instruments, 379  
    asynchronous instruments, 379, 383-384  
    grouping instruments, 379  
    synchronous instruments, 379, 381-383  
metrics, 346, 369-387  
microservice, 209-210  
    (see also microservices architecture)  
microservices architecture, 8, 209, 211-213  
monitoring, 345, 370  
monolith, 209, 213  
    (see also monolith architecture)  
monolith architecture, 210-211, 216  
MTBF (mean time between failures), 170  
MTS (Message Transfer System), 221  
multiple containers, 156  
multiple returns, 51  
multiplexer (mux), 113  
multitiered architecture, 4  
mutex, 121, 201, 205  
mux (multiplexer), 113

## N

network I/O, 195  
networked applications, history of, 4-6

- Node Exporter, 378
- nullpotence, 109
- numeric types, 30
  - (see also complex numbers, simple numbers)

## O

- object-oriented programming, 16-17
- observability
  - concepts of, 370
    - (see also cardinality)
  - definition of, 11-12, 344
  - pillars of, 346-347
    - (see also logging, metrics, tracing)
  - purpose of, 345
  - techniques, 178, 334
- observers (see asynchronous instruments)
- Okta, 216
- opening plug-ins, 242
- OpenTelemetry, 347-350, 352-356, 359-361, 374-377
- OpenZipkin, 355
- os.Getenv function, 308
- os.LookEnv function, 308
- OTel (see OpenTelemetry)

## P

- PaaS (platform as a service), 178
- panic built-in function, 51
- parallelism, 20
- patterns
  - circuit breaker pattern, 77-80, 81, 280
  - debounce pattern, 80-84, 87
  - fan-in pattern, 93-95
  - fan-out pattern, 95-97
  - future pattern, 97-101
  - ports and adapters pattern (see hexagonal architecture)
  - Retry function, 84
  - sharding, 101-106, 194, 205-206
  - stability patterns, 77-92
  - throttle pattern, 81, 86-89, 280
  - timeout, 90-92
- PEM, 147-149
- Perl, 135
- personnel divergence, 187
- PHP, 135, 349
- Plateau of Productivity, 213
- platform as a service (PaaS), 178

- plug-ins, 242-255
  - (see also Go plug-ins, looking up plug-ins, loose coupling plug-ins, opening plug-ins, symbol plug-ins)
- pointer arithmetic, 22
- pointers, 22, 42-44, 53-54
- polling, 326-327
- portability, 185
- ports, 255
- ports and adapters pattern (see hexagonal architecture)
- POST method, 226-229
- PostgreSQL Exporter, 378
- privacy enhanced mail (see PEM)
- private key, 146
- programming languages, 15, 16
- Prometheus, 369, 373, 375, 385-387
  - (see also Prometheus exporter)
- Prometheus exporter, 375-378
- Promises (see future pattern)
- promotion, 63
- PromQL, 373
- protocol buffer compiler installation, 232
- protocol buffers, 231-236
- public clouds, 13
- public key, 146
- public-key cryptography, 146
- publish-subscribe messaging pattern, 223, 224
- publishing ports, 154, 156
- pull-based metric collection, 372
- Push Gateway, 378
- push-based metric collection, 371
- Python, 22-23, 24, 41, 349

## Q

- quicksort, 167

## R

- range keyword, 45-46, 67
- read locks, 122-123
- receiver argument, 58
- receivers, 59
- recovery, 176
- recursive functions, 51
- Redis, 217
- Redis Exporter, 378
- redundancy, 176, 268, 290-293
  - (see also messaging redundancy)
- reference types, 54

- release stage, 183
- reliability, 170, 171, 267, 290, 305
  - (see also redundancy)
- remote procedure calls (see RPC)
- repository name component, 153
- representational state transfer (see REST)
- request-response messaging pattern, 223, 224-226
- resilience
  - building, 173, 268-269
  - definition of, 8-9, 266-268
  - importance of, 266
  - reliability and, 9, 125, 267
- resource state, 124, 196
- REST, 112, 221, 224, 225, 230, 287
- RESTful, 117
- retries, 275-280, 290
  - (see also messaging redundancy, retry storm)
- Retry function, 84-86
- retry storm, 276
- return statement, 51
- root command, 311
- root spans, 351, 368
- RPC, 225, 230
- Ruby, 23, 41, 135, 349
- run stage, 183
- rune, 31, 40-41
- Rust, 22-23, 349

## S

- SAAS (software as a service), 4
- sample, 369
- Scala, 135
- scalability
  - bottlenecks in, 194-195
  - definition of, 6-7
    - (see also horizontal scaling, vertical scaling)
  - efficiency and, 173, 197-209
  - forms of, 193-194
  - service architectures and, 209-217
  - states of, 195-197
- scalar operations, 289
- scalar values, 289
- Scan method, 135
- search space, 177
- secrets, 180
- security, 305
- sentinel error, 111
- sequence number, 125
- server affinity, 125
- serverless architecture, 213-217
- serverless computing, 213
- service architectures (see microservices architecture, monolith architecture, serverless architecture)
- service contract, 8
- service discovery, 223, 305
- service interface, 231
- service mesh, 223
- service processes, 184
- service-oriented architecture (SOA), 209
- shallow health checks, 296, 297-298
- ShardedMap, 104-106
- sharding, 101-106, 194, 205-206
- shards, 194
- short variable declarations, 33
- signed integer, 31
- simple numbers, 31
- simple object access protocol (see SOAP)
- simplicity, 197
- slice operator, 39-40
- slices, 36, 37-41
- Slope of Enlightenment, 213
- snowflakes, 188-189
- SOA (service-oriented architecture), 209
- SOAP, 221, 230
- software as a service (SAAS), 4
- SoundCloud, 373
- spans, 350, 357-358
  - (see also root spans)
- Split function, 95-96
- Splunk, 187
- SQL databases, 137-144
- stability patterns, 77-92
  - (see also circuit breaker pattern, debounce pattern, throttle pattern, timeout)
- stack divergence, 186
- stack trace, 350
- state, 124, 195
  - (see also application state, resource state)
- stateful application (see application state)
- stateless, 124, 196-197
- static analysis, 177
- static linking, 23
- static typing, 24-25
- StatsD, 371

- steam of events, 389
- Stream function, 75-76
- string literals, 32, 40-41
  - (see also double-quote style string literals, interpreted string literals)
- strings, 30, 32, 40-41
- struct embedding, 62
- struct field tags, 319-321, 323
- structs, 17, 57-58, 122, 315
  - (see also anonymous struct)
- structural typing mechanism, 18
- Stubby, 230
- subcommands, 312
- subsystem, 267
- swap function, 51
- Swift, 349
- switch Statement, 48-49, 67
- symbol plug-ins, 242
- synchronous instruments, 379, 381-383
- synchronous messaging pattern (see request-response messaging pattern)
- system, 267
- system, definition of, 8
- systems engineering, 169

## T

- tag component, 153
- testability, 176-177, 185
- Three Pillars of Observability, 346-347
- throttle pattern, 81, 86-89, 280
- throttling, 270-274
- Tickers, 208
- tight coupling
  - communication, 224
  - definition of, 220
  - forms of, 221-223
- time dimension, 207
- time series, 370
- timely function, 208
- timeout, 90-92
- timeouts, 229, 281, 285-286
- Timers, 208
- timestamp, 390
- TLS, 145-149
- token bucket, 88-89, 271
- TOML, 329
- trace, 346
- Tracer, 356
- tracer provider, 355-357

- (see also global tracer provider)
- traces, 350
- tracing, 346, 350-368
  - (see also distributed tracing)
- tracing exporters, 353
  - (see also Console Exporter, Jaeger Exporter)
- transaction log, 124-137, 258
- transitive downstream dependencies, 5
- transitive upstream dependencies, 5
- transport layer security (see TLS)
- Twelve-Factor App, 178-188, 306
- type assertion, 60
- type embedding, 61-63
  - (see also interface embedding, struct embedding)
- types
  - arrays, 36-37
  - Booleans, 30
  - byte, 31, 40-41
  - containers, 36-42
  - embedded pointers, 63
  - errors, 49
  - extending standard composite types, 59
  - floating point, 31
  - interfaces, 59-60
  - maps, 36, 41-42
  - rune, 31, 40-41
  - signed integer, 31
  - slices, 36, 37-41
  - strings, 30, 32, 40-41
  - structs, 17, 57-58
  - unsigned integer, 31

## U

- Uber Technologies, 355
- unary RPC, 236
- underscore operator, 35
  - (see also blank identifier)
- unsigned integer, 31
- upstream dependencies, 5
  - (see also transitive upstream dependencies)

## V

- value, 125
- variable declaration, 32-33
- variables, 32, 42-44
- variadic functions, 39, 50-55
- variadic operator, 54
- version control, 307



vertical scaling, [7](#), [185](#), [193](#), [194](#)  
vertical sharding, [101-103](#), [205](#)  
Viper, [181](#), [329-333](#)

## W

watchConfig function, [326](#)  
Windows Exporter, [378](#)  
write locks, [122](#)  
WriteDelete method, [127-130](#), [139](#)  
WritePut method, [127-130](#), [139](#), [204](#)

## Y

YAML, [315](#), [321-325](#), [329](#)  
yaml.Marshal function, [321-323](#)  
yaml.Unmarshal function, [322](#)

## Z

Zap logging, [394-399](#)  
zero values, [33](#)