



## Stichwortverzeichnis

Sebastian Bergmann, Stefan Priebsch

Softwarequalität in PHP-Projekten

ISBN (Buch): 978-3-446-43539-1

ISBN (E-Book): 978-3-446-43582-7

Weitere Informationen oder Bestellungen unter

<http://www.hanser-fachbuch.de/978-3-446-43539-1>

sowie im Buchhandel.

# Stichwortverzeichnis

## A

Akzeptanztest [4](#), [23](#), [407](#), [432](#), [455](#)  
Allgegenwärtige Sprache [282](#), [285](#)  
Alternative PHP Cache (APC) [213](#), [449](#)  
Alternative PHP Debugger (APD) [213](#), [217](#)  
– `apd_set_pprof_trace()` [218](#)  
– `pprofp` [217](#)  
Apache Bench [204](#), [207](#)  
Apache HTTP Server [207](#), [212](#), [215](#)  
Äquivalenzklasse [18](#)  
Aufwärmphase [215](#)  
Ausführungspfad [24](#)

## B

Backup [233](#)  
Barrierefreiheit [186](#)  
Benchmark [207](#)  
Build  
– automatisierter [113](#)  
– kontinuierlicher [110](#)  
– Management [109](#)  
– täglicher [110](#)  
Bytekit [13](#)

## C

Caching [449](#)  
– Cache Locking [207](#)  
– Cache Miss [205](#)  
– Cache Priming [205](#)  
– Cache Warming [205](#)  
Call Graph [213](#), [217](#)  
Callee [217](#)  
Caller [217](#)  
Callgrind [213](#)  
Capability Maturity Model Integration (CMMI) [7](#)  
Carica CacheGrind [219](#)  
Change Risk Anti-Patterns (CRAP) Index [10](#)

## Code Smell

– Duplizierter Code [9](#), [85](#), [111](#), [283](#)  
– Große Klasse [283](#)  
– Lange Methode [283](#)  
Code-Altlasten [129](#), [434](#)  
Code-Coverage [10](#), [23](#), [145](#), [286](#), [301](#), [312](#)  
– Path Coverage [23](#)  
Codeduplikat [111](#)  
Code-Review [113](#), [279](#)  
Coding Standard [13](#), [283](#)  
Collective Code Ownership → Kollektives Eigentum  
CPU-Metrik [204](#)  
CPU-Zeit [204](#)  
Cross Site Request Forgery [242](#)  
Cross Site Scripting [245](#)

## D

Data Set [151](#)  
Datenbanktest [141](#)  
Datenzugriffsschicht [450](#)  
DBUnit [146](#)  
Debug Build [206](#)  
Debugger [217](#), [219](#)  
Debugging-Symbole [215](#)  
Dependency Injection [9](#), [11](#), [28](#), [305](#), [445](#), [454](#)  
Dialog [183](#)  
Domain Specific Language (DSL) [426](#)  
Domain-Driven Design →  
– Domänengetriebenes Design  
Domänengetriebenes Design [285](#)

## E

Entwurfsmuster  
– Acceptor [446](#)  
– Chain of Responsibility [449](#)  
– Data Transfer Object [115](#)  
– Decorator [152](#), [160](#)  
– Multiton [304](#)

- Observer 306, 447
- Proxy 450
- Repository 99
- Singleton 304
- Table Data Gateway 63
- Visitor 446
- eXtreme Programming 386

**F**

- Fehlbedienung 199
- Fehlerbehandlung 235
- Fehlertoleranz 199
- Fixture 85
- Flaschenhals 142, 203
- Flood 211
- Fluent Interface 315, 428
- Funktion
  - aufgerufene 217
  - Aufrufer 217
- Funktionalität 4
- FURPS 3

**G**

- GCC 215
- Gebrauchstauglichkeit 4, 183
- Gleitpunktzahl 324, 325
- Global State 11, 91
- gprof 222

**H**

- Happy Path 18
- Hardening 231
- Heatmap 200
- HipHop 109
- HTTPerf 211
- HTTPLoad 204
- Hudson 280

**I**

- Instrumentierung 212, 215
- Integrationstest 458
- Iteration 391

**J**

- JavaScript 194
- Jenkins 14
- JMeter 211

**K**

- KCachegrind 216, 217, 288
- Keep It Simple, Stupid (KISS) 388
- Klickdummy 183, 200

- Kohäsion 12
- Kollektives Eigentum 394
- Konfiguration 108
- Kontinuierliche Integration 110, 303, 340, 387, 410, 456
- Kopplung 12, 235

**L**

- Lasttest 204, 206
- Lime 309
- Lines of Code 113
  - Comment 114
  - Executable 115
  - Non Comment 114
- Lock 213
- Logfile 200

**M**

- MacCallGrind 219
- Memcache 205, 449
- meminfo 204
- Migration 143
- Mock-Objekt 9, 11, 29, 30, 141, 146, 290, 323, 445
- Model-View-Controller (MVC) 384
- MySQL 143, 450

**N**

- Navigation 183
- Nebenläufigkeit 207
- Non-Mockable Total Recursive Cyclomatic Complexity 11
- NPath-Komplexität 10, 24

**O**

- OProfile 213, 222
- Optimierung 203
- OWASP 238

**P**

- Pair Programming → Programmieren in Paaren
- Partitionierung 385, 450
- PEAR 436
- Performanz 192, 203
- Performanztest 203, 206
- Persistenz 141
- PHP\_CodeBrowser 14
- PHP\_CodeSniffer 13, 281
- phpcpd 13
- PHP\_Depend 13
- php.ini 218

phploc 12  
phpmd 13  
PHPT 437  
PHPUnit 12, 146, 286, 437  
– Selenium-Erweiterung 410  
Planning Game 390  
Port  
– privilegiertes 212  
Profiler 204, 206, 212, 217, 219, 222  
Profiling 204, 212  
Programmieren in Paaren 393  
Propel 109  
Pylot 204, 209

## Q

Qualität  
– externe 4, 31  
– interne 4, 31  
Qualitätsmanagement 5  
Qualitätssicherung  
– konstruktive 7

## R

Reaktionsfreudigkeit 4  
Rechteverwaltung 231  
Redirect 352  
Refaktorisierung 5, 111, 283, 287, 308, 382  
Regression 27  
Regressionstest 179  
Remote Procedure Call (RPC) 340  
Representational State Transfer (REST) 340  
Response Time 209  
Root-Server 232  
Runkit 435

## S

sar 204  
Scrum 281, 388, 408, 454  
Secure by Design 232  
Security by Obscurity 234  
Seiteneffekt 29  
Selenium 21, 314, 384, 405  
Selenium Grid 410  
Selenium IDE 410  
Selenium RC 409, 457  
Separation of Concerns 235, 384  
Server API (SAPI) 212  
Serviceorientierte Architektur (SOA) 385, 450  
Sharding 385, 450  
Shared Hosting 232  
Shared Memory 213  
Sicherheit 4, 231

Siege 211  
Single Responsibility Principle 9, 29  
sismo 303  
SOAP 340  
Software Process Improvement and Capability  
Determination (SPICE) 7  
Software-Artefakt 113  
Software-Metrik 10, 112  
Softwarequalitätsmodell 3  
Softwaretestpyramide 31  
Spaghetti-Code 434  
Spike Solution 390  
Sprint 454  
Standup-Meeting 388  
Story Point 391  
Stub 29, 30, 141, 290  
Suhosin 233  
Symfony 109  
Systemaufruf 215  
Systemmetrik 204, 207  
Systemtest 19

## T

Technische Schulden 5  
Test  
– automatisierter 109  
– Black-Box 17, 407, 434  
– Browserkompatibilitätstest 432  
– Capture&Replay 410  
– Edge-to-Edge 36, 40  
– End-to-End 31, 405  
– End-to-Test 4  
– Integration 36, 432  
– White-Box 17  
Testautomatisierung 425  
Testbarkeit 10, 29, 444, 451  
Testdatenbank 166, 181  
Test-First Programmierung 7  
Testgetriebene Entwicklung 8, 26, 278, 388, 440  
Testinventar 22, 142, 151, 166, 179, 181, 316, 415  
Testisolation 22  
Testplan 20  
Test-Smell 84  
– Begieriger Test 86  
– Duplizierter Testcode 85  
– Fragiler Test 21, 89, 415, 417, 419  
– Indirekter Test 94  
– Konditionale Testlogik 100  
– Langsamer Test 98  
– Lügenger Test 97  
– Mock-Overkill 103

- Obskurer Test [91](#)
- Selbstvalidierender Test [101](#)
- Skip-Epidemie [105](#)
- Undurchsichtiger Testname [95](#)
- Websurfender Test [102](#)
- Zusicherungsroulette [86](#)
- Testumgebung [205](#)
- Throughput [209](#)
- Timeboxing [458](#)
- Toleranzintervall [324](#)
- top [204](#)
- Trait [447](#)

## U

- Überwachung [231](#)
- Ubiquitous Language → Allgegenwärtige Sprache
- Unit-Test [24](#)
- Usability [183](#)
- User Story [390](#)
- User-Test [200](#)

## V

- Valgrind [213](#), [222](#)
- Velocity [391](#), [458](#)
- Verfügbarkeit [4](#)
- Versionsmanagement [109](#)
- vfsStream [291](#)
- Virtualisierung [232](#)

## W

- Wasserfallmodell [457](#)
- WebDAV [357](#)
- Webdienst [195](#)
- Webgrind [219](#), [288](#)
- Webservice → Webdienst
- White-Box-Test [24](#)

## X

- Xdebug [213](#), [217](#), [219](#), [286](#)
- xdebug.profiler\_enable [219](#)
- xdebug.profiler\_enable\_trigger [219](#)
- xdebug.profiler\_output\_dir [219](#)
- XHProf [213](#), [219](#), [288](#)
  - xhprof\_disable() [221](#)
  - xhprof\_enable() [221](#)
  - xhprof\_html [221](#)
- xhprof\_lib [221](#)
- xhprof.output\_dir [221](#)
- XML-RPC [340](#)

## Y

- You Ain't Gonna Need It (YAGNI) [388](#)
- YSlow [192](#)

## Z

- Zend Extension [218](#)
- Zusicherung [27](#)
- Zuverlässigkeit [4](#)
- Zyklomatische Komplexität [10](#), [113](#)