Profiling and inspection with Blackfire

Emir Beganović • April 19th 2018

Agenda

- Introduction
- Performance as a feature
- Profiling and analysis
- Metrics
- Continuous Integration
- Q&A

About me

- Backend Developer @ Shpock
- Zend Certified PHP Engineer
- 7+ years of development career
- open source contributor

GitHub: @emirb
emir@php.net

Disclaimer

40% of users abandon a website that takes more than 3 seconds to load

100€

to solve a performance issue in **development**

1 500€

to solve the same issue on **staging and test environments**

15 000€

to solve the same issue **once it reached production**

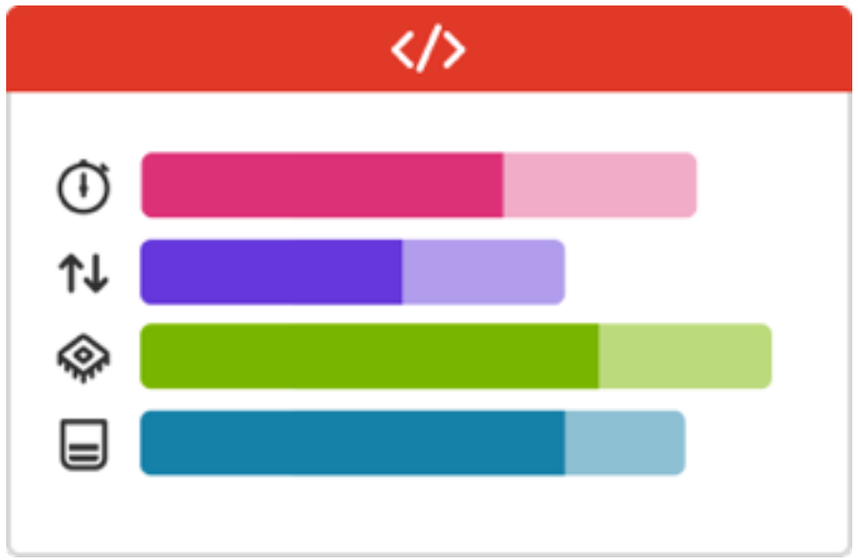
Solving issues in **production costs** a lot **more than** solving them in **development**

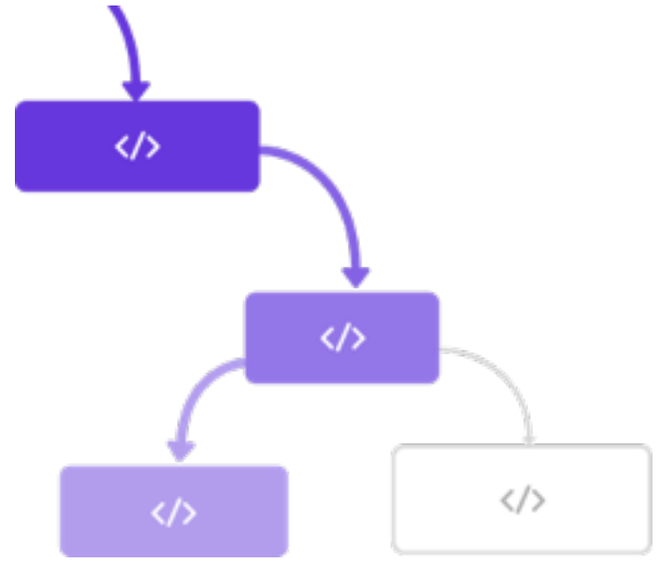

GOOD
HIGH
VERY HOT
DANGER
LOW

Old-school don'ts

- `microtime()`
- `memory_get_usage()`
- `memory_get_peak_usage()`

In-depth analysis with Blackfire





Wall time | I/O | CPU | Memory | Network | SQL

History

- built by SensioLabs, people behind Symfony
- available since spring 2015
- supports all platforms
- concentrated on PHP

Installation

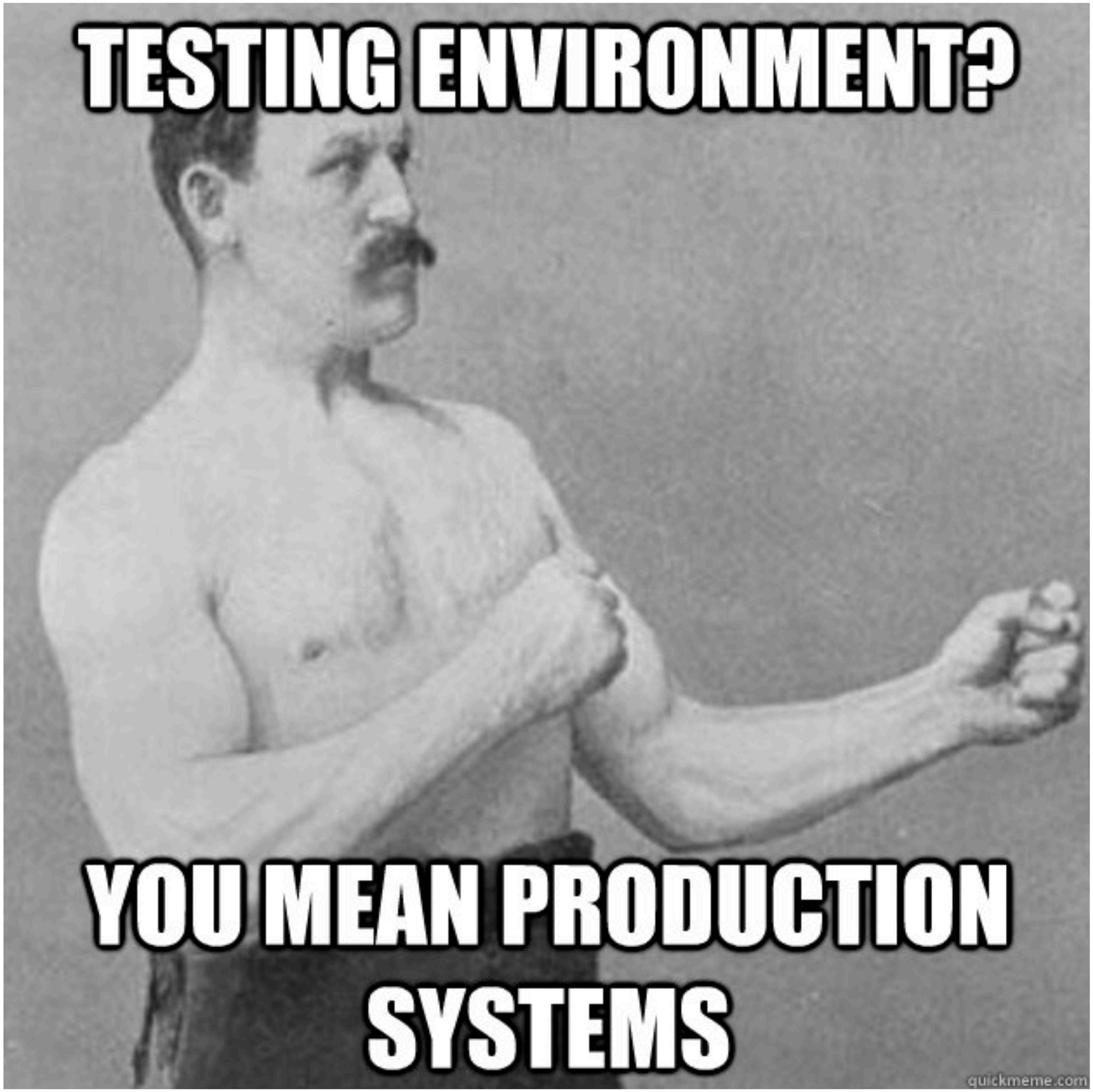
- consists of agent and extension
- optionally (recommended): Chrome extension

- platform support:

heroku platform.sh fortrabbit FORGE

Profiling possibilities

- Extension: browser requests within Chrome
- CLI with curl
- manually inside the code (PHP SDK)
- daemons using signals
- profiling in production: yes


TESTING ENVIRONMENT?
YOU MEAN PRODUCTION SYSTEMS
quickmeme.com

Let's profile!

Analysis

Let's optimise & verify

References

Metrics

- hundreds of built-in metrics:
  - Symfony
  - Laravel
  - Drupal
  - Doctrine
  - Memcache
  - Redis
  - internal calls

Custom metrics

```
metrics:
  cache.write: # metric name
    label: "Cache::write() calls"
    matching_calls:
      php:
        - callee:
            selector: '=Cache::write'
```

Testing custom metrics

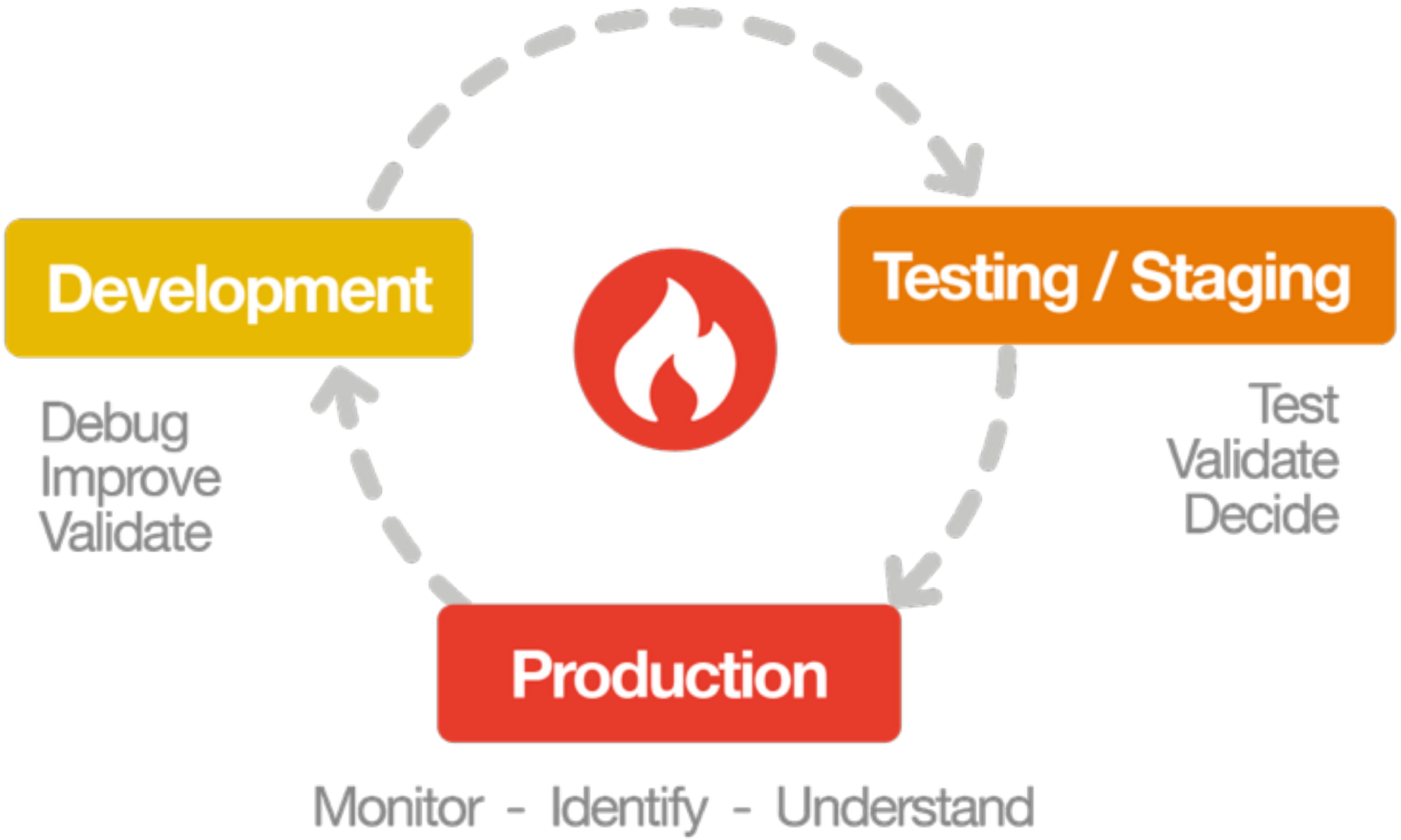
```
tests:
    "Ensure that Cache::write() calls do not
consume too much memory":
        path: "/.*"
        assertions:
            - "metrics.cache.write.peak_memory
< 10mb"
```

Assertions

Privacy

- anonymisation of:
  - function arguments
  - SQL queries
  - HTTP queries
  - file locations
- calls verifiable through Blackfire proxy (SDK)
- on-premise version for large clients

Continuous Performance Testing



Continuous Integration

- Travis, Jenkins, anywhere - simple webhook

```
curl -X POST https://blackfire.io/api/v1/build/
env/ENV-UUID/webhook \
    --user "userid" \
    -d "endpoint=http://myapp.com/" \
    -d "title=Deployed c589252"
```

Pricing

Profiler
SaaS

19,90€ / month, billed yearly

Profile your application in dev & prod and get optimization recommendations.

Features

- Profiling on production servers
- Unlimited profiles
- Timeline visualization
- Wall Time, CPU, I/O, Memory and Network dimensions
- Profiling Web Pages, CLI scripts, API Calls
- HTTP calls, SQL requests
- Recommendations
- Share your profiles publicly
- PHP SDK

Premium
SaaS

82,50€ / month, billed yearly

Automate performance testing and collaborate with a small team.

Profiler Features +

- Performance assertions
- Scheduled builds
- E-mail build notification channel

Collaborative usage

- 1 environment, 3 collaborators
- 30 days data retention

Additional users from 19,90€ per month
Additional environments from

Enterprise
SaaS / On premises

Custom

Integrate continuous performance management with all of your tools, for all of your developers

Premium Features +

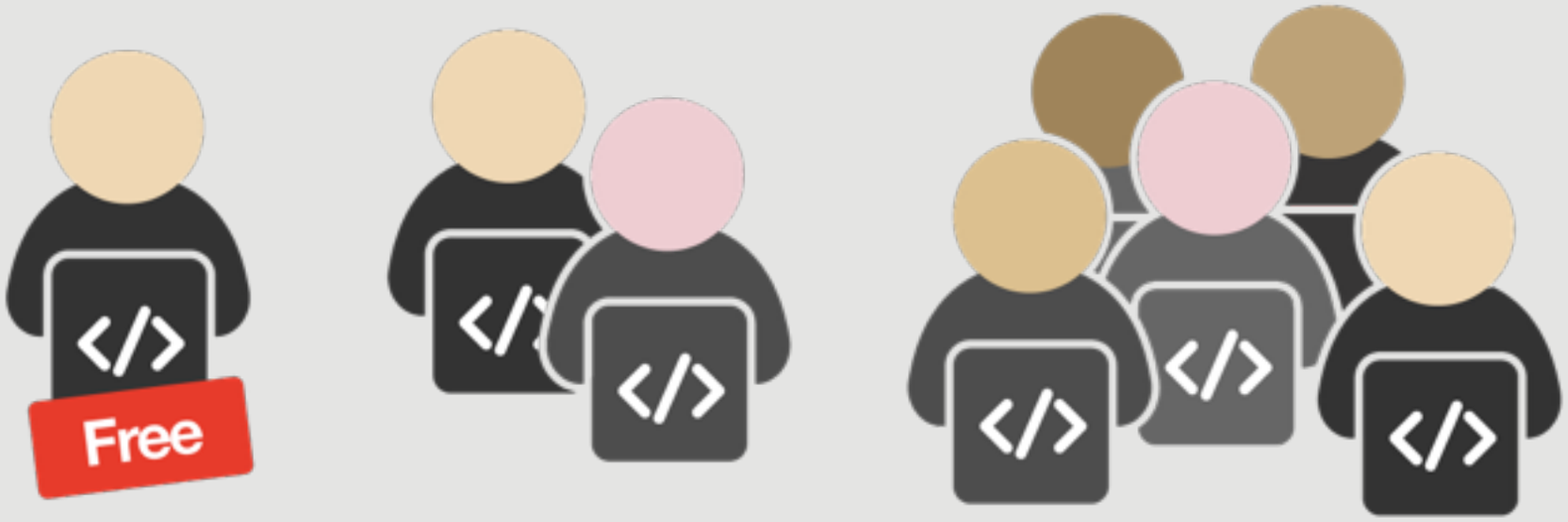
- Webhook and native integrations to trigger test scenarios and receive build notifications
- Priority support

Collaborative usage

- Custom # of environments and collaborators
- 90 days data retention

Special **20% discount** for

ViennaPHP members



ViennaPHP2018

Questions?

Thanks.