INCREASING CODE QUALITY: STATIC ANALYSIS

EMIR BEGANOVIĆ

AGENDA

- introduction
- what, why, how?
- what's in it for me?
- tools: Phan, PHPStan

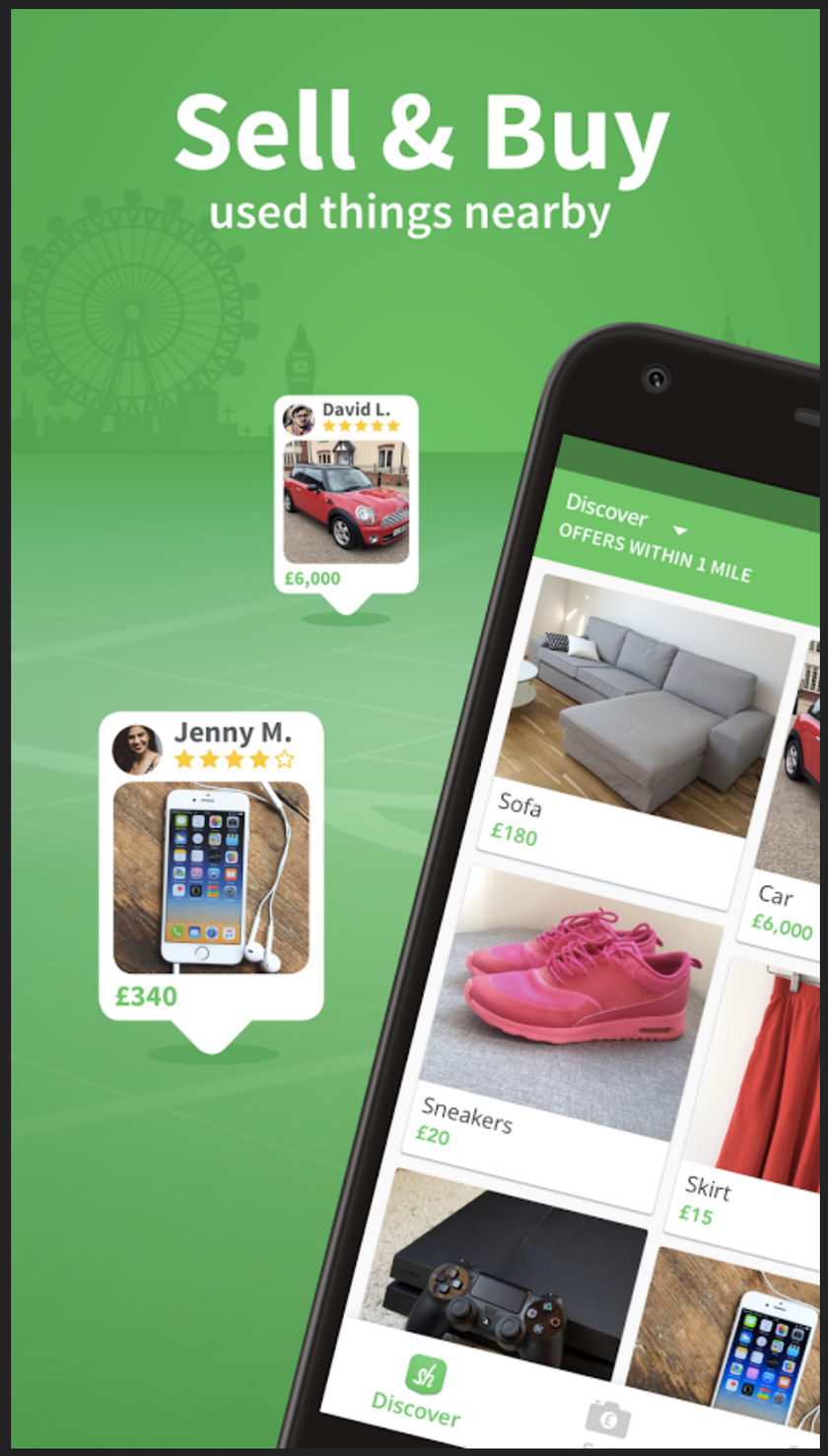
ABOUT ME

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

- https://github.com/emirb
- emir@php.net

WE'RE HIRING!

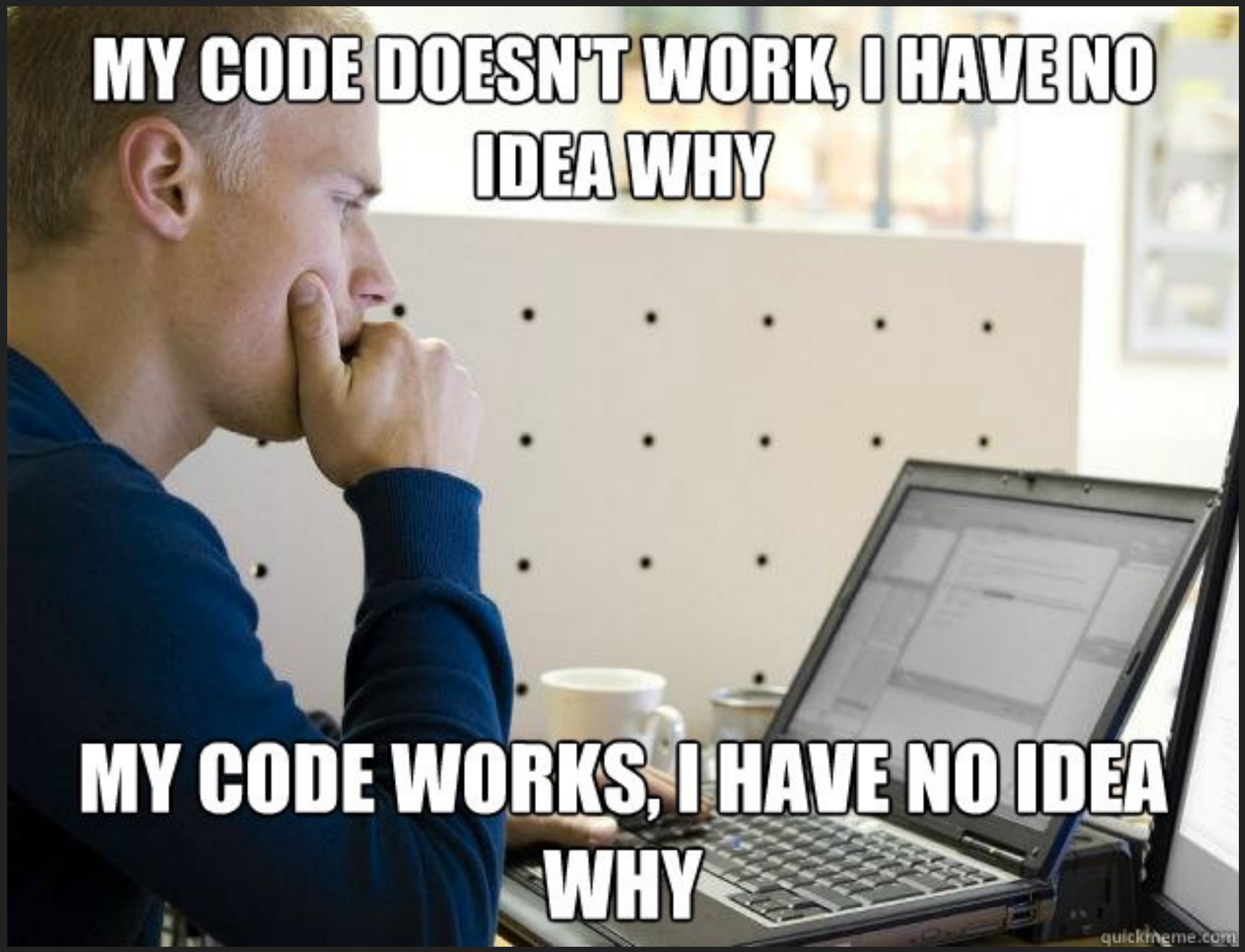
- ~140 team members from 30+ nations
- 10 million users
- 100 million monthly searches
- biggest PHP deployment in Austria (traffic, hits)
- **Frontend developers**
  ReactJS, SCSS
- **Backend developers**
  PHP 7, Symfony components,
  MongoDB, ELK
- Check out: jobs.shpock.com



STATIC CODE ANALYSIS

- static code analysis vs dynamic analysis (runtime)
- analysing code **without running it**
- get to know your codebase
- you’re (probably) **already** doing it in your IDE

WHY?


MY CODE DOESN'T WORK, I HAVE NO IDEA WHY
MY CODE WORKS, I HAVE NO IDEA WHY
quickmeme.com

WHY?

- PHP is an interpreted, dynamically typed language
- gradually typed (PHP7 type hints)
- check code upfront

WHAT DOES IT HELP?

- maintaining and increasing code quality
- finding lexical and syntax mistakes
- ensuring a ruleset across the team
- making pull request reviews **easier**

I Am Devloper

@iamdevloper

10 lines of code = 10 issues.

500 lines of code = "looks fine."

Code reviews.

BUT WHY?

- static analysis can be forced throughout CI pipeline (and pre-commit hooks!)
- rulesets can be shared across the team
- there are (still) people not using PhpStorm's inspections

POSITIVE IMPACTS

- Be a **defensive programmer**
- Easier comprehension of legacy codebases
- Catch problems as early as possible
- Ensure consistent quality assurance process
- Increase confidence in your development cycle

COST OF BUGS

- hotfix deployments still **take time**
- automated preventions **saves time**
- prevention is better than a cure

GIVE ME THE TOOLS

- the most basic: `php -l`
- coding standards: `phpcs`, `phpmd`
- metrics (`phpmetrics`, `phploc`)
- dependency graph (`pdepend`)
- analysis: **Phan, PHPStan**

PHAN

- Built at Etsy (home of Rasmus) since 2015
- 2500+ stars, a lot of activity
- built on php-ast extension
- philosophy: incrementally strengthening analysis
- extensible with custom *plugins*

INSTALL AND CONFIGURE

- `composer require --dev "phan/phan"`
- config file: .phan/config.php
- set up directories whitelist
- boolean options

MINIMUM CONFIGURATION

```php
<?php

return [

    'check_docblock_signature_return_type_match' => true,
    'processes'                                  => 1,
    'analyzed_file_extensions'                   => ['php'],
    'directory_list'                             => [
        'src',
    ],
    "exclude_analysis_directory_list"            => [
        'vendor/',
    ],
];
```

ANALYSIS EXAMPLE

```php
class Entity
{
    protected $timespan = false;

    public function __construct(string $timespan)
    {

        $this->timespan = $timespan;
    }

    /**
     * @return void
     */
    public function fooBar($foo): array
    {
        return $this;
    }

    /**
     * @param string $code
     */
    public function fooMethod(string $errorcode): int
    {
    }

    public function failingArgument()
    {
        return str_replace('foo', 'bar', $this->fooBar(false));
    }
```

```
./src/Acme/HelloWorld/Entity.php:20 PhanTypeMismatchReturn Returning type void but fooBar() is declared to return array
./src/Acme/HelloWorld/Entity.php:31 PhanTypeMismatchArgument Argument 1 (errcode) is false but fooMethod() takes string defined
 at ./src/Acme/HelloWorld/Entity.php(30)
./src/Acme/HelloWorld/Entity.php:44 PhanTypeMismatchArgumentInternal Argument 3 (subject) is int but \str_replace() takes array
|string
./src/Acme/HelloWorld/Entity.php:72 PhanDeprecatedFunction Call to deprecated function \Acme\HelloWorld\Entity::getRequest() de
fined at ./src/Acme/HelloWorld/Entity.php:215
./src/Acme/HelloWorld/Entity.php:131 PhanTypeMismatchProperty Assigning string to property but \Acme\HelloWorld\Entity::timespa
n is false
```

MORE USAGES

- backwards compatibility issues check (PHP 5 -> 7 , 7.1 -> 7.2)

PHPSTAN

- Exists since the middle of 2016
- Also ~2500 stars, but half a million downloads (packagist)
- PHP >= 7.0 as well
- the more typehints and annotations you use, the better
- extensible with easy-to-write rules!
- Different analysis levels

INSTALLATION AND CONFIG

- `composer require —dev phpstan/phpstan`
- `vendor/bin/phpstan analyse -l 4 -c phpstan.neon src`

ANALYSIS EXAMPLE

```
[phpstan]  ------ ------------------------------------------------------------------------------
[phpstan]   Line   src/Acme/Test/TestCase.php
[phpstan]  ------ ------------------------------------------------------------------------------
[phpstan]   452    Class TestCaseB does not have a constructor and must be
[phpstan]          instantiated without any parameters.
[phpstan]   531    Access to undefined constant
[phpstan]          TestCaseB::TEST_CONSTANT
[phpstan]  ------ ------------------------------------------------------------------------------
[phpstan]
[phpstan]  ------ ------------------------------------------------------------------------------
[phpstan]   Line   src/Acme/Test/TestCaseB.php
[phpstan]  ------ ------------------------------------------------------------------------------
[phpstan]   365    Class TestCaseB does not have a constructor and must be
[phpstan]          instantiated without any parameters.
[phpstan]  ------ ------------------------------------------------------------------------------
[phpstan]
[phpstan]  ------ ------------------------------------------------------------------------------
[phpstan]   Line   src/Acme/Test/TestCaseC.php
[phpstan]  ------ ------------------------------------------------------------------------------
[phpstan]   56     Access to an undefined property \Acme\TestCaseC::$manager.
[phpstan]   1308   Return typehint of method \Acme\TestCaseC::foo() has invalid type
[phpstan]          \Acme\TestCaseD.
[phpstan]   1322   Property \Acme\TestCaseC::$logger does not accept \Acme\LoggerInterface.
```

SOME ISSUES

- https://github.com/phpstan/phpstan/issues/67
- PHPStan is triggering class autoloading
- **static analyzer should never execute code**
- moving to Roave/BetterReflection soon

EXTENDING PHPSTAN

- https://github.com/thecodingmachine/phpstan-strict-rules
- add your own!
- real world use case: rule to check **Object** class names which are disallowed from PHP 7.2
- example https://wiki.php.net/rfc/object-typehint

INTEGRATION WITH CI

- Closing the cycle: integrate static analysis within CI (Jenkins/Travis/AWS CodeBuild)
- Make checks **required** in your GitHub/GitLab repository

CONCLUSION

- easy to integrate into your development flow
- start very tolerant
- increase strictness over time
- focus more on real problems

QUESTIONS?

THANKS!