AOE

the open web company

Agile Programming Practices

Tales from ~~One Thousand and One~~ 1.092 Code Reviews

About me

I’m a

➔ software developing

➔ photography loving

➔ pipe organ playing

➔ dad of 2 wonderful kids

happily working @ AOE in Wiesbaden (Germany)

Introduction

Agile Programming Practices

Pair Programming

Code Review

Fine-scale feedback	Continuous process	Shared understanding	Programmer welfare
→ Pair programming	→ Continuous integration	→ Coding standards	→ Sustainable pace
→ Test-driven development	→ Refactoring	→ Collective code ownership	
→ Planning game	→ Small releases	→ Simple design	
→ Whole team		→ System metaphor	

Source: *"Extreme Programming", Computerworld (online), December 2001*

Pair Programming

Pair Programming: Why

- Increased code quality
  - Significant reduction of defects
  - Higher confidence in the code
- Easier team-building and communication
  - Constant sharing of knowledge
  - Better transfer of skills
- Improved resiliency of a pair to interruptions

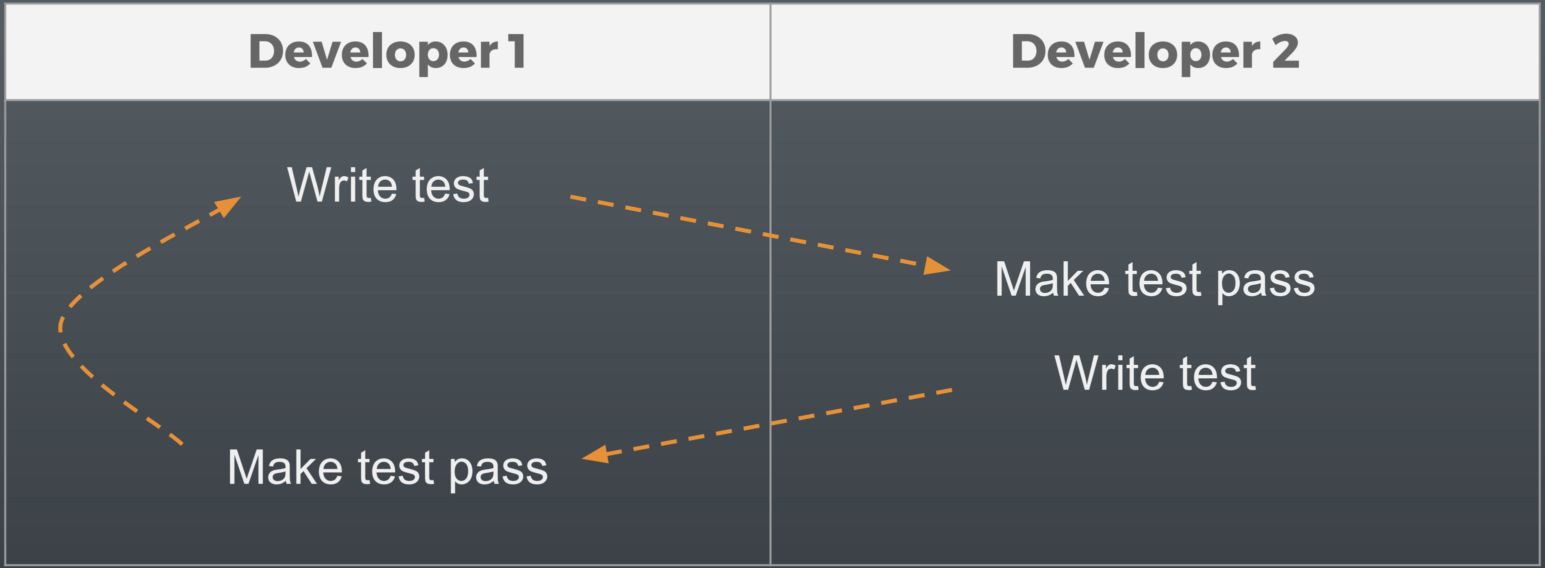
Pair Programming: Economics

Potential Cost	Expected Benefits
15%	**15%**
Overhead for pairing	Fewer defects

Pattern: Driver/Navigator

Driver	Navigator
Focused on tactics	**Focused on strategy**
→ clean code	→ how to design
→ code compiles and runs	→ what to test
→ automated tests pass	→ what to refactor

Pattern: Ping/Pong



Pair Programming: Best Practices

- Switch roles frequently (use a timer)
- Avoid the “watch the master” phenomenon
- Be actively engaged
- Keep talking
- Create a pairing friendly environment

Remote Pair Programming

Driver/Navigator	Ping/Pong
Sharing tool must support ➔ sharing of driver's screen (unscaled, high quality) ➔ audio call *Example: Skype*	**Sharing tool must support** ➔ screen sharing ➔ audio call ➔ simultaneous access to keyboard and mouse *Example: Screenhero*

» The adjustment period from solo programming to collaborative programming was like eating a hot pepper. The first time you try it, you may not like it because you are not used to it. However the more you eat it, the more you like it.
— *Anonymous*

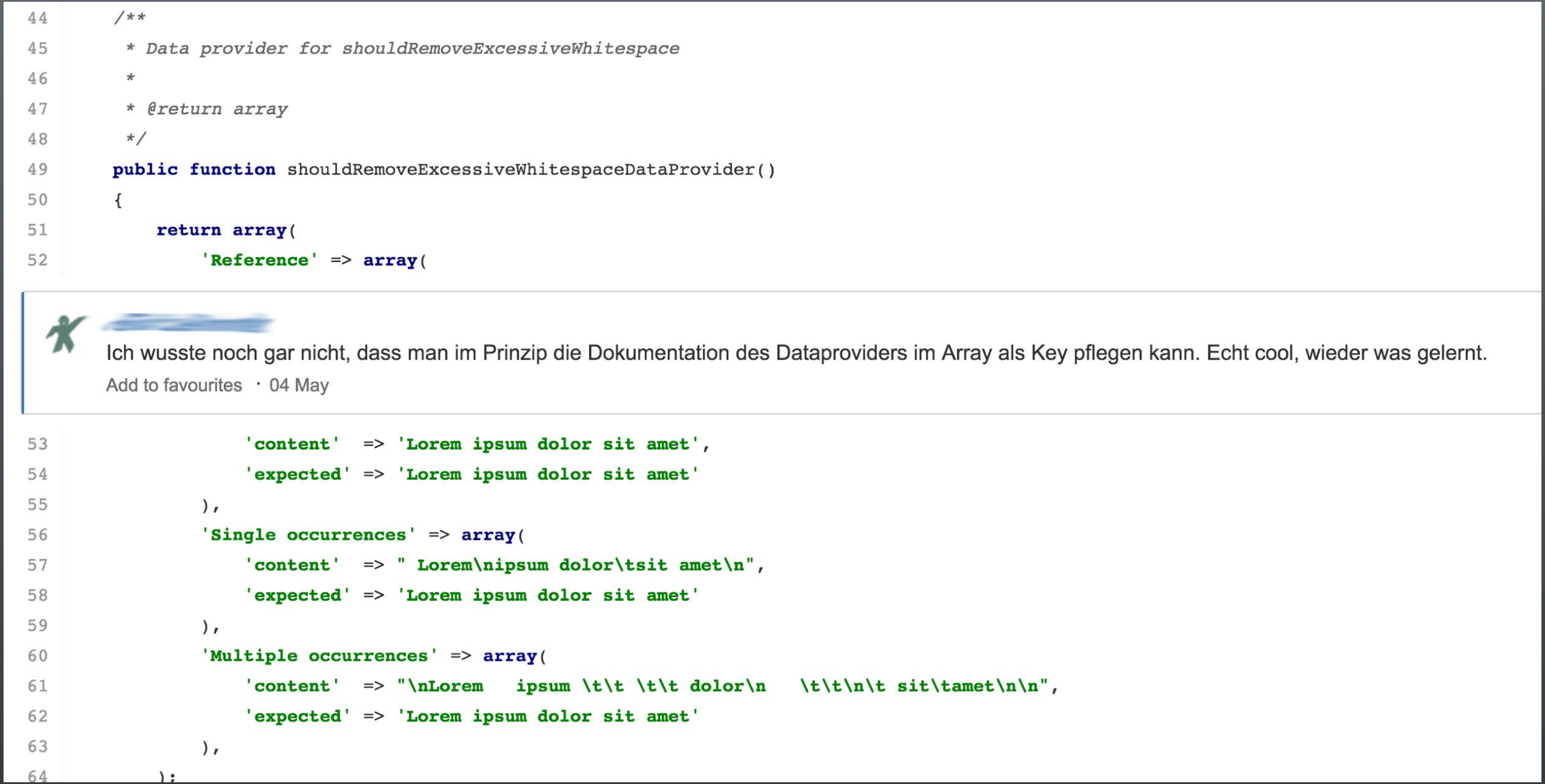
Code Reviews

Developers should pair program often with other developers, learning their coding habits and styles. This provides context to comments received on proposed changes.

— *Brandon Savage*

Code Reviews: Why

- Put a second set of eyes on a particular bit of code
- Force a developer to explain what his/her code does
- Find risky / wrong code and offer quality improvement suggestions
- Time to think about consequences of a change throughout the system
- Knowledge sharing
  - Learn new tricks
  - Correct bad habits



```
    /**
     * Data provider for shouldRemoveExcessiveWhitespace
     *
     * @return array
     */
    public function shouldRemoveExcessiveWhitespaceDataProvider()
    {
        return array(
            'Reference' => array(
```

[redacted]

Ich wusste noch gar nicht, dass man im Prinzip die Dokumentation des Dataproviders im Array als Key pflegen kann. Echt cool, wieder was gelernt.

Add to favourites · 04 May

```
                'content'  => 'Lorem ipsum dolor sit amet',
                'expected' => 'Lorem ipsum dolor sit amet'
            ),
            'Single occurrences' => array(
                'content'  => " Lorem\nipsum dolor\tsit amet\n",
                'expected' => 'Lorem ipsum dolor sit amet'
            ),
            'Multiple occurrences' => array(
                'content'  => "\nLorem   ipsum \t\t \t\t dolor\n   \t\t\n\t sit\tamet\n\n",
                'expected' => 'Lorem ipsum dolor sit amet'
            ),
        );
```

Code Reviews: When

Pre Commit	Post Commit
➔ No unreviewed code will be merged ➔ Code changes can be squashed ➔ Review blocking for moving on ➔ Risk of “pushing” code reviews	➔ Asynchronous ➔ Potentially wrong code might go into master ➔ Additional commits to fix findings ➔ Risk of ignoring the review outcome

Code Reviews: How

- Agree on
  - what kind of commits require a review
  - review criteria
  - how many reviewers are required
- Eliminate as many defects as possible, regardless who “caused” the error
- Not all suggested changes have to be incorporated in the code

Code Reviews: Do

- Smaller commits lead to more manageable code reviews
- Outline objectives to understand the what and why
- Use comment tiers, e.g.
    - Suggestion: *“You might…”*
    - Disagreement: *“You should…”*
    - Defect (blocking): *“You must…”*
- Take your time: faster is not better

Code Reviews: Don’t

- Don’t take it personal
  - Code reviews are highly subjective
- Don’t push reviews
  - Address open reviews e.g. in the team’s daily stand up
- Don’t use metrics to single out developers
  - Number of defects is related to complexity
- Don’t try to substitute code with the reviewer’s logic

Example

Code Review Checklist

- ❏ Obvious errors, risks, incompatibilities
- ❏ Compliance with team/community code style
- ❏ Following best practices of team/community
- ❏ Test coverage
- ❏ Code smells
- ❏ DRY – Don’t repeat yourself
- ❏ SRP – Single Responsibility Principle
- ❏ KISS – Keep it simple, stupid
- ❏ YAGNI – You aren’t gonna need it

» Code reviews are opportunities to spot things that are dangerous, bad, mistaken or wrong in the code, and to offer quality improvement suggestions.
— *Brandon Savage*

Facts & Figures

In ~18 month, the team created

Ø 14/week

1.092 Code Reviews

The biggest review contained

Too many

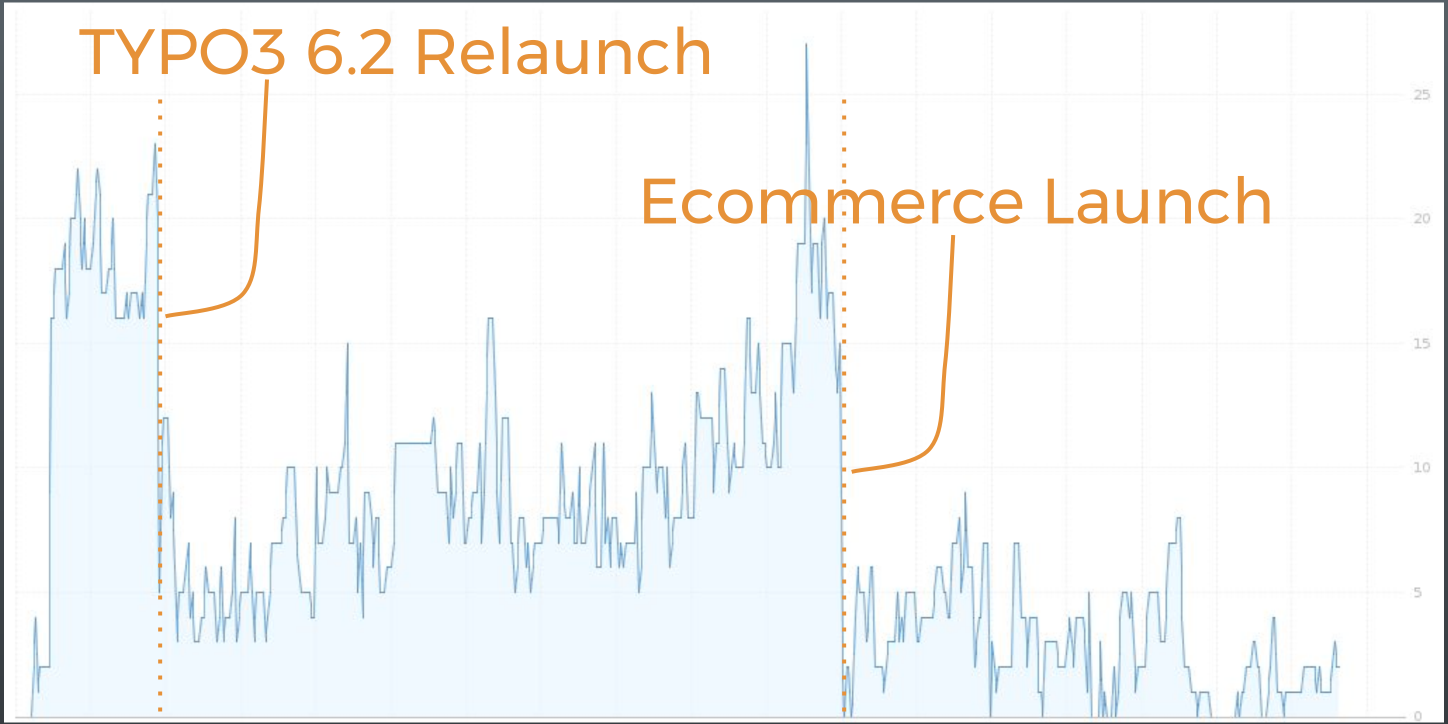
655 Files

The most controversial review led to

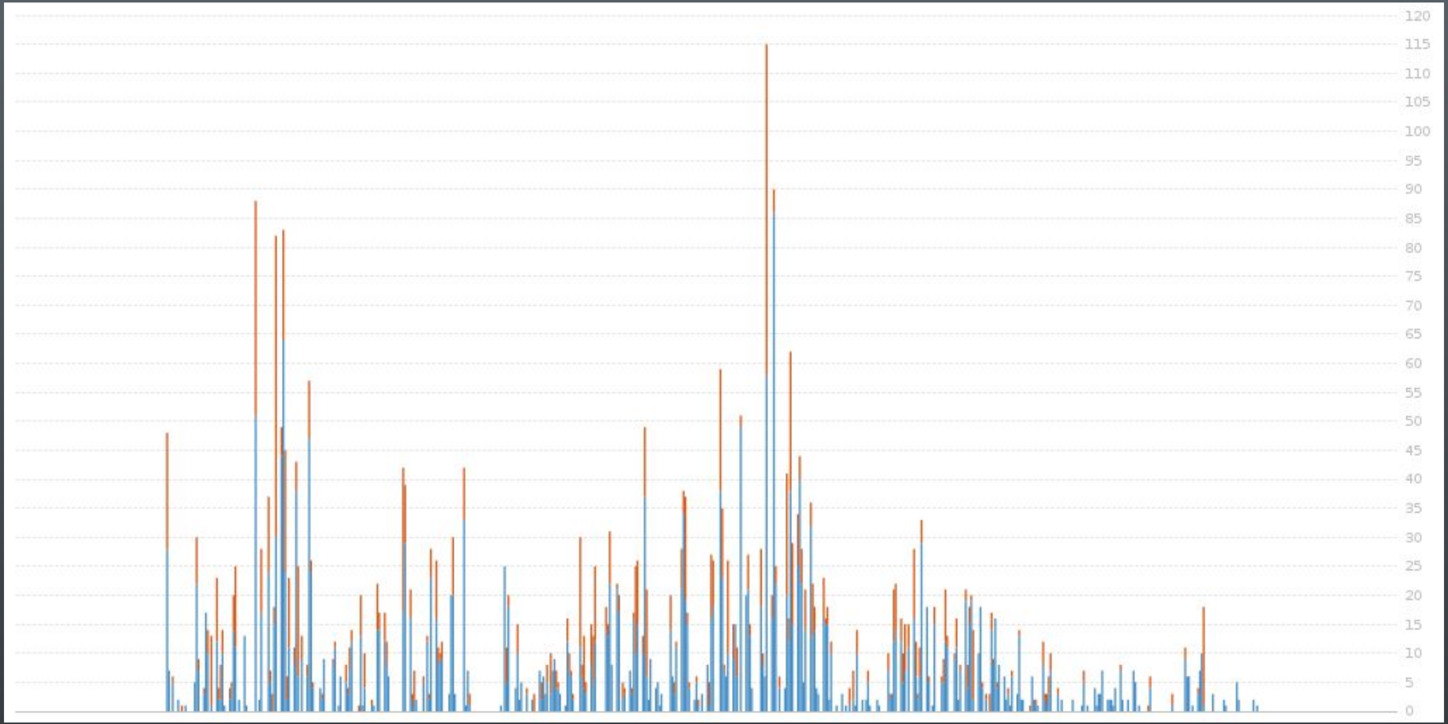
which should have
been addressed offline

144 Comments

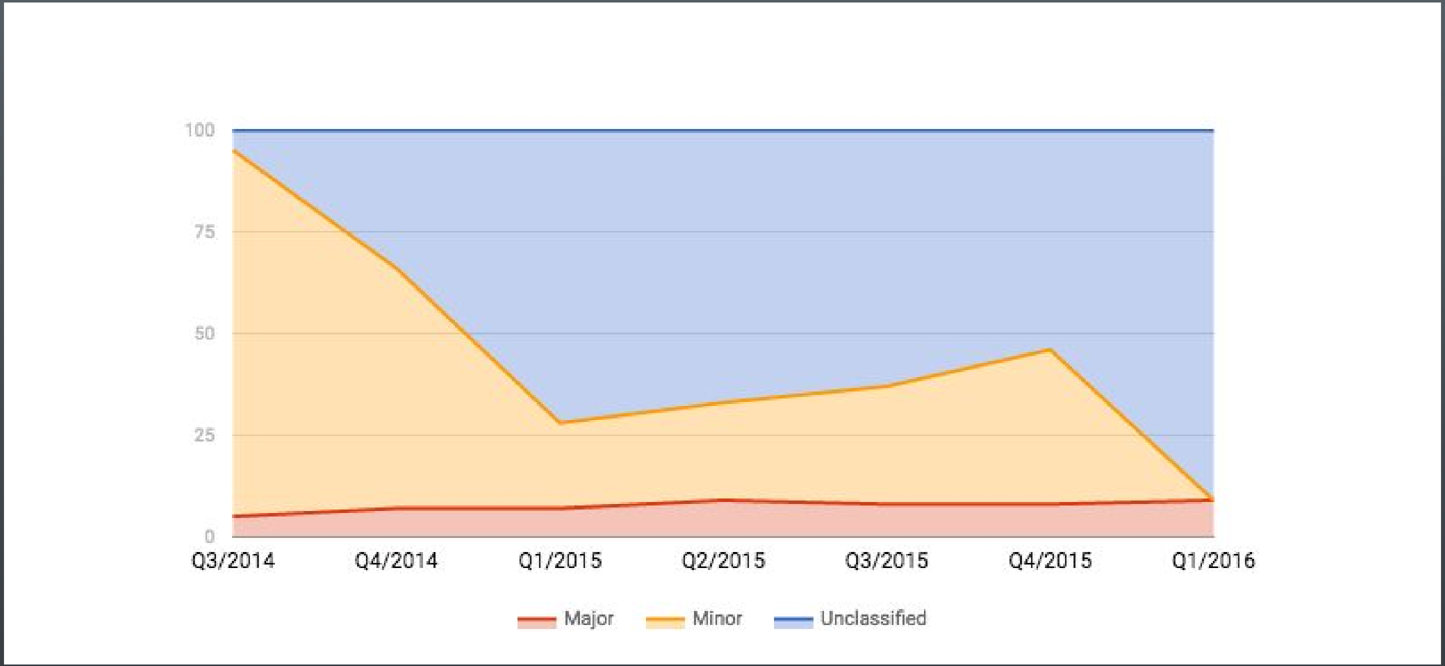
Open review count over time



Comment volume over time



Defect ranking over time



Final Thoughts

Agree on **guidelines and standards** in your team

Program in pairs as often as possible

Keep commits and code reviews small

Any stupid can write the program that computer understands but only good programmers write code that humans understand.

— *Martin Fowler*

@ritschie

Resources

- 11 proven practices for more effective, efficient peer code review
  http://www.ibm.com/developerworks/rational/library/11-proven-practices-for-peer-review/
- 10 ways to be a faster code reviewer
  http://blog.codacy.com/top-10-faster-code-reviews/
- The Pitfalls of Code Review (And How To Fix Them)
  http://www.brandonsavage.net/the-pitfalls-of-code-review-and-how-to-fix-them/
- Screenhero
  https://screenhero.com/

AOE | in Germany

AOE GmbH
LuisenForum, Kirchgasse 6
65185 Wiesbaden
Germany

Telefon: +49 6122 70 70 7 - 0
Fax: +49 6122 70 70 7 - 199
E-Mail: sales-de@aoe.com
Web: www.aoe.com