IPC International PHP Conference

by devmio

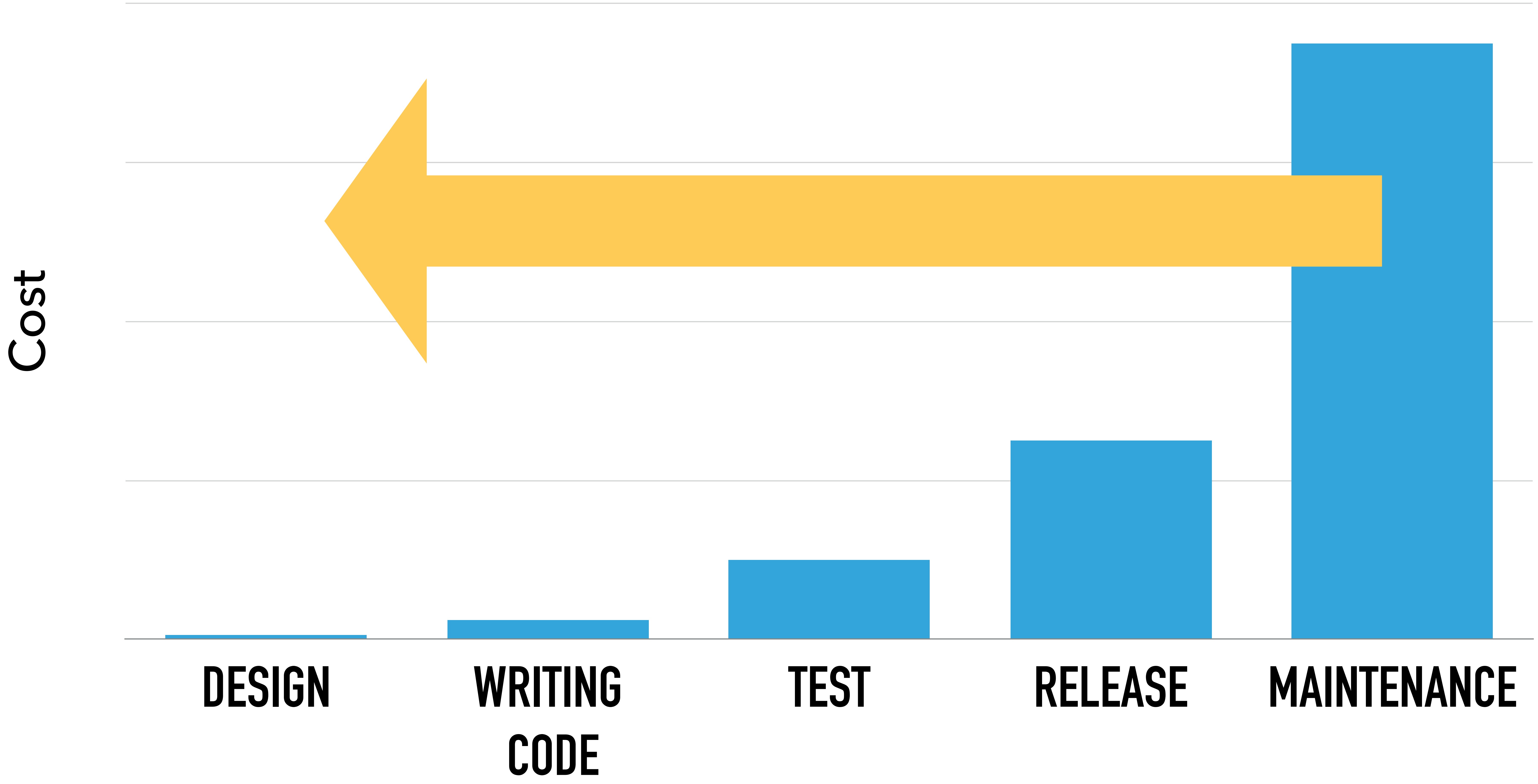
Effective Code Review

Dave Liddament | Lamp Bristol

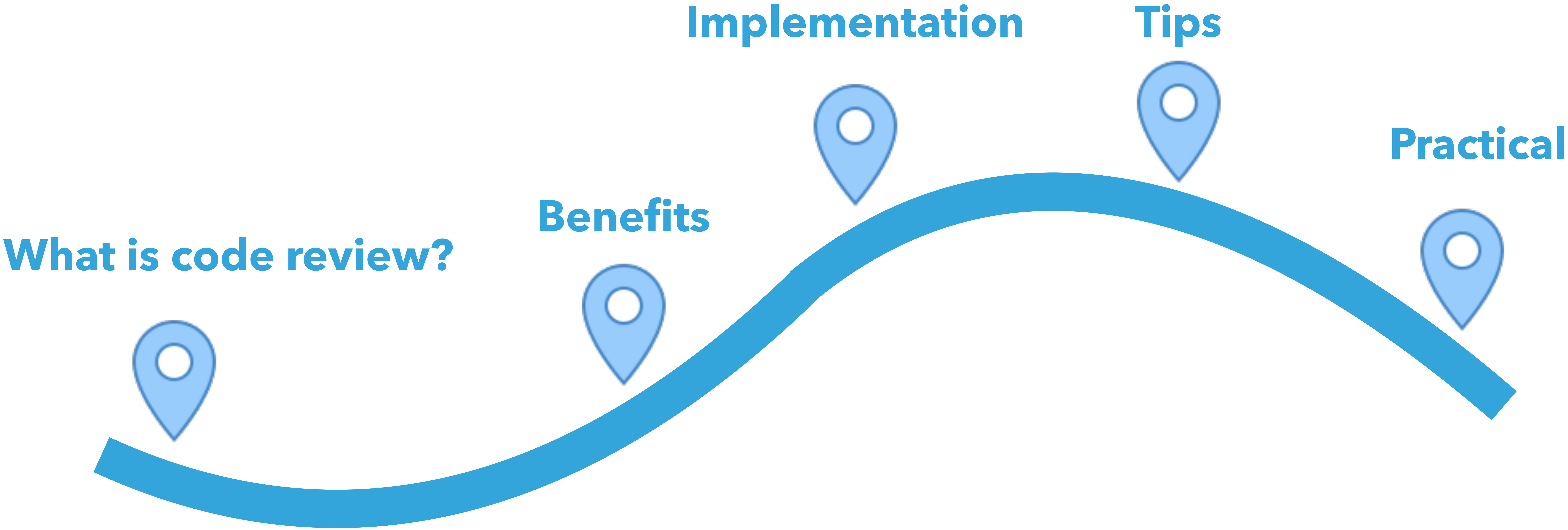
@daveliddament

EFFECTIVE CODE REVIEW REDUCES OVERALL COST OF SOFTWARE DEVELOPMENT

REDUCE COST OF DEFECTS * - FIND IT SOONER



@daveliddament



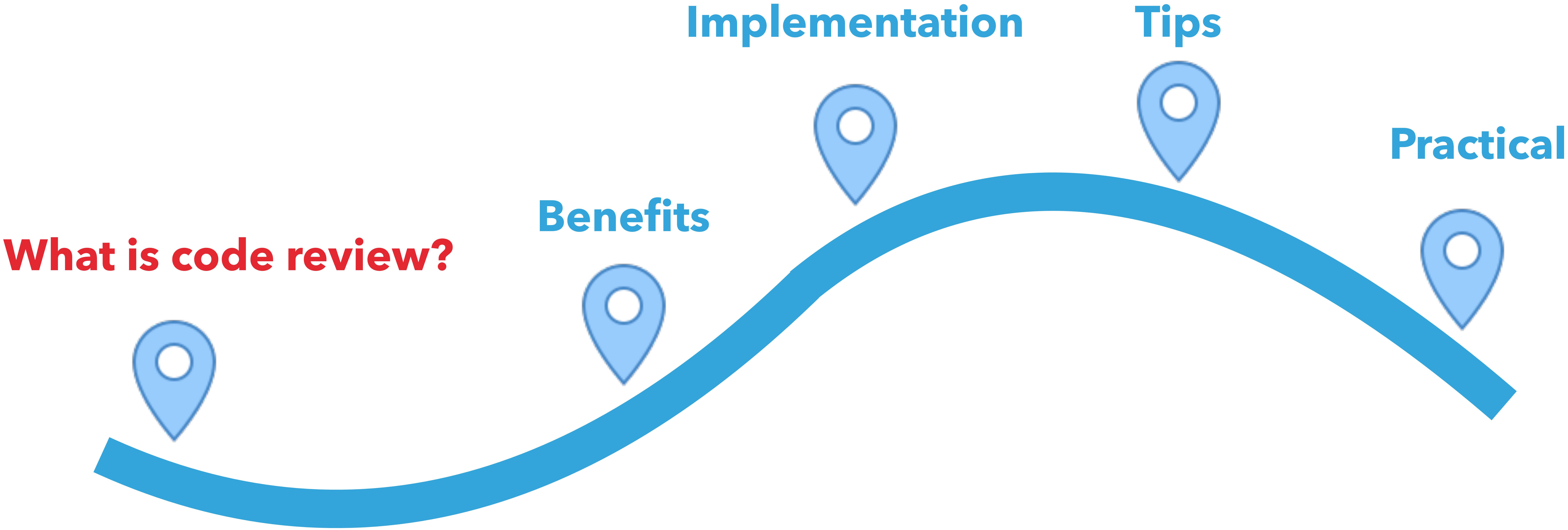
What is code review?

Benefits

Implementation

Tips

Practical



What is code review?

Benefits

Implementation

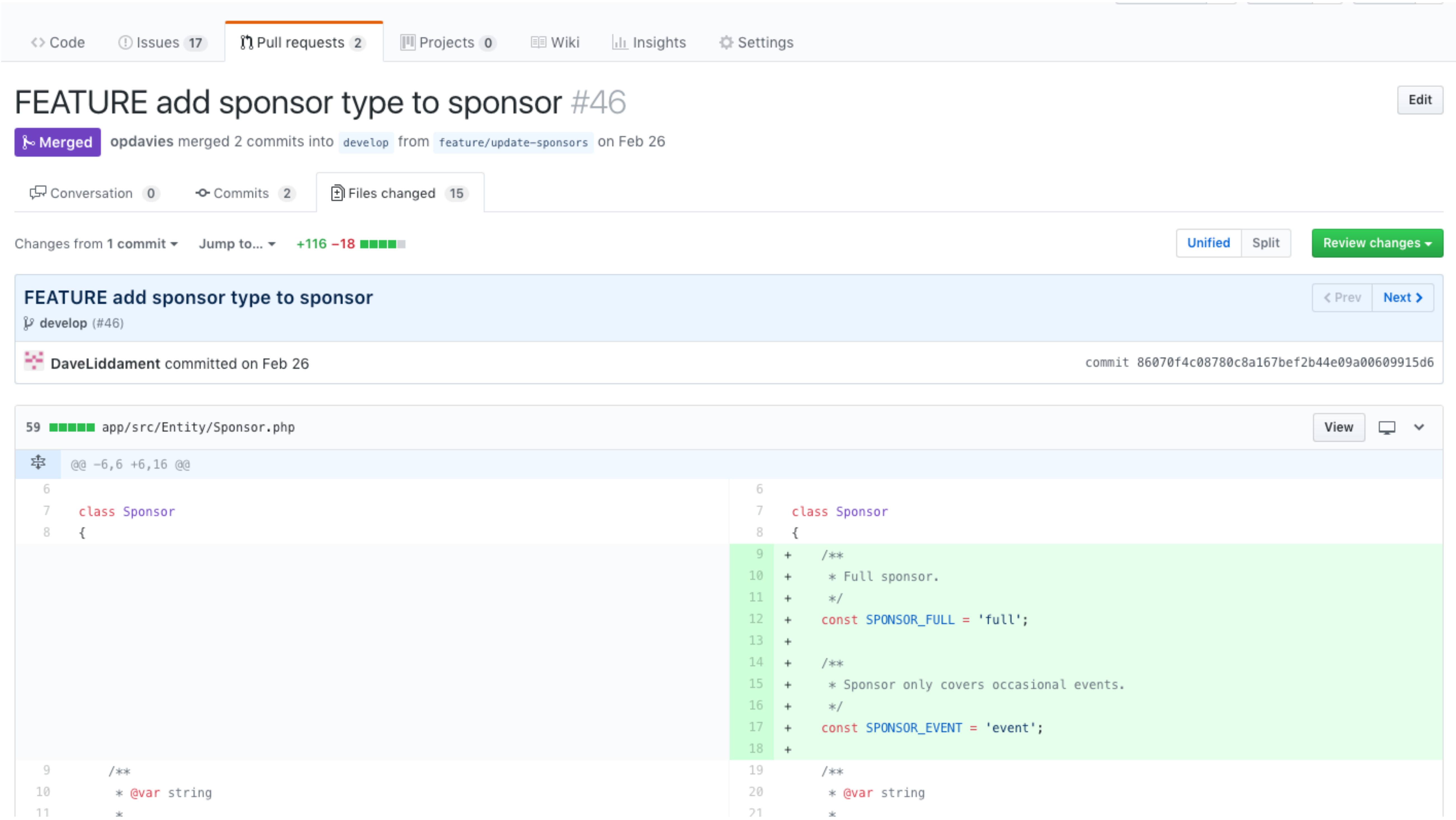
Tips

Practical

> Code review is the systematic examination of source code.
>
> It is intended to find mistakes overlooked in software development, improving the overall quality of software.

Wikipedia

HOW IS IT DONE?



FEATURE add sponsor type to sponsor #46

Merged opdavies merged 2 commits into `develop` from `feature/update-sponsors` on Feb 26

+116 −18

FEATURE add sponsor type to sponsor

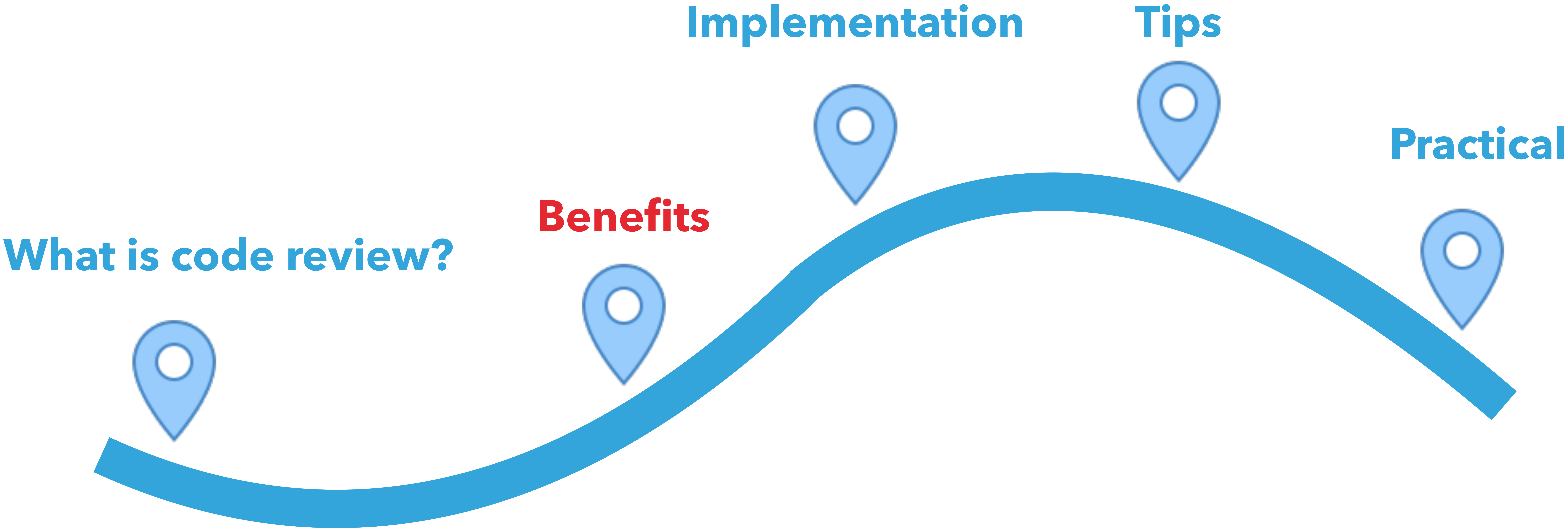
develop (#46)

DaveLiddament committed on Feb 26 `commit 86070f4c08780c8a167bef2b44e09a00609915d6`

59 app/src/Entity/Sponsor.php

```
@@ -6,6 +6,16 @@

class Sponsor                          class Sponsor
{                                      {
                                       +    /**
                                       +     * Full sponsor.
                                       +     */
                                       +    const SPONSOR_FULL = 'full';
                                       +
                                       +    /**
                                       +     * Sponsor only covers occasional events.
                                       +     */
                                       +    const SPONSOR_EVENT = 'event';
                                       +
    /**                                    /**
     * @var string                          * @var string
     *                                      *
```

@daveliddament



What is code review?

Benefits

Implementation

Tips

Practical

@daveliddament

WHAT ARE DEFECTS?

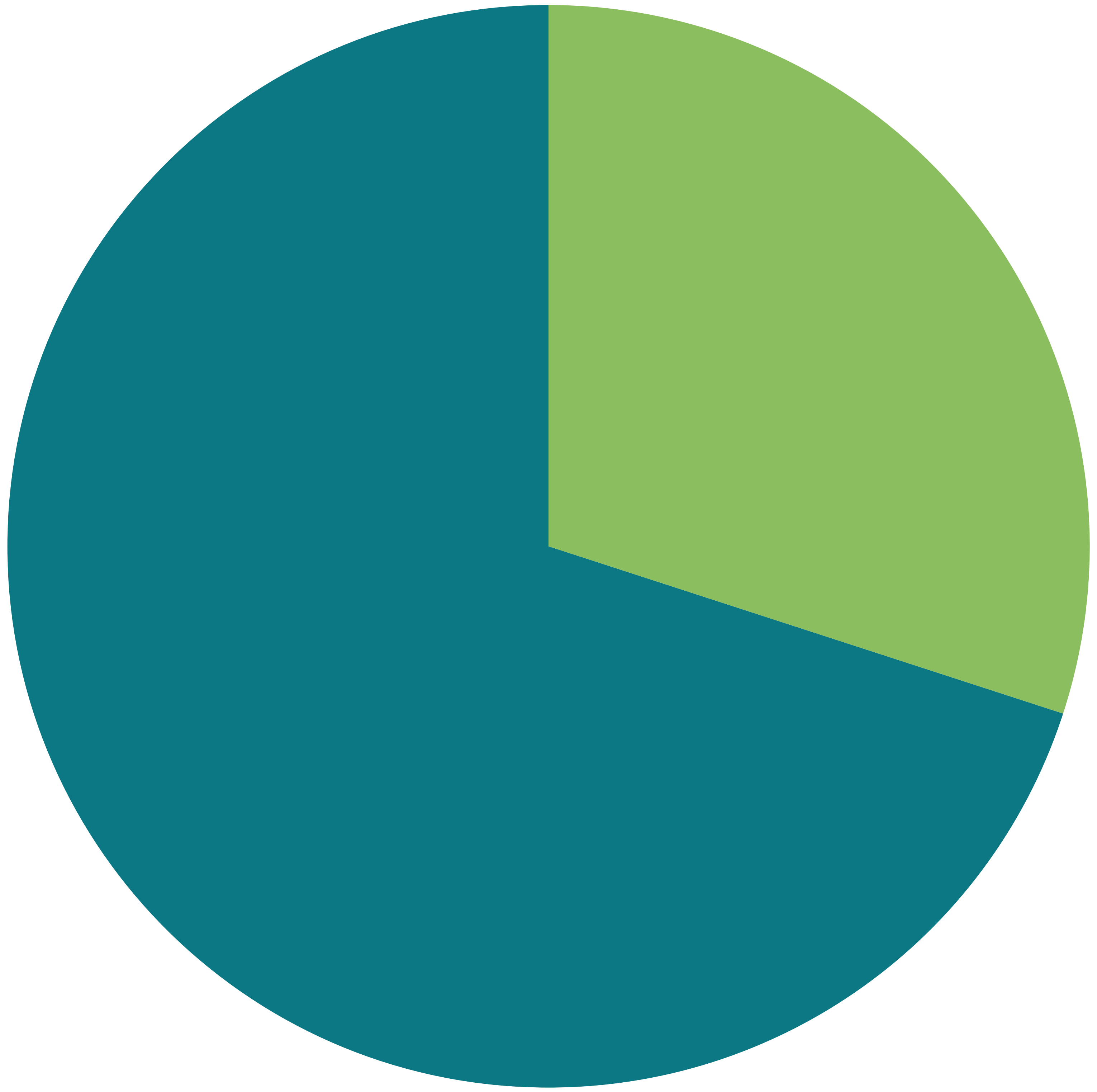
Bugs

Evolvability

AN EVOLVABILITY DEFECT IS…

Code that makes codebase less compliant with standards, more error prone, or more difficult to modify, extend or understand.

WHAT ARE DEFECTS?



Bugs

Evolvability

[1, 2]

@daveliddament

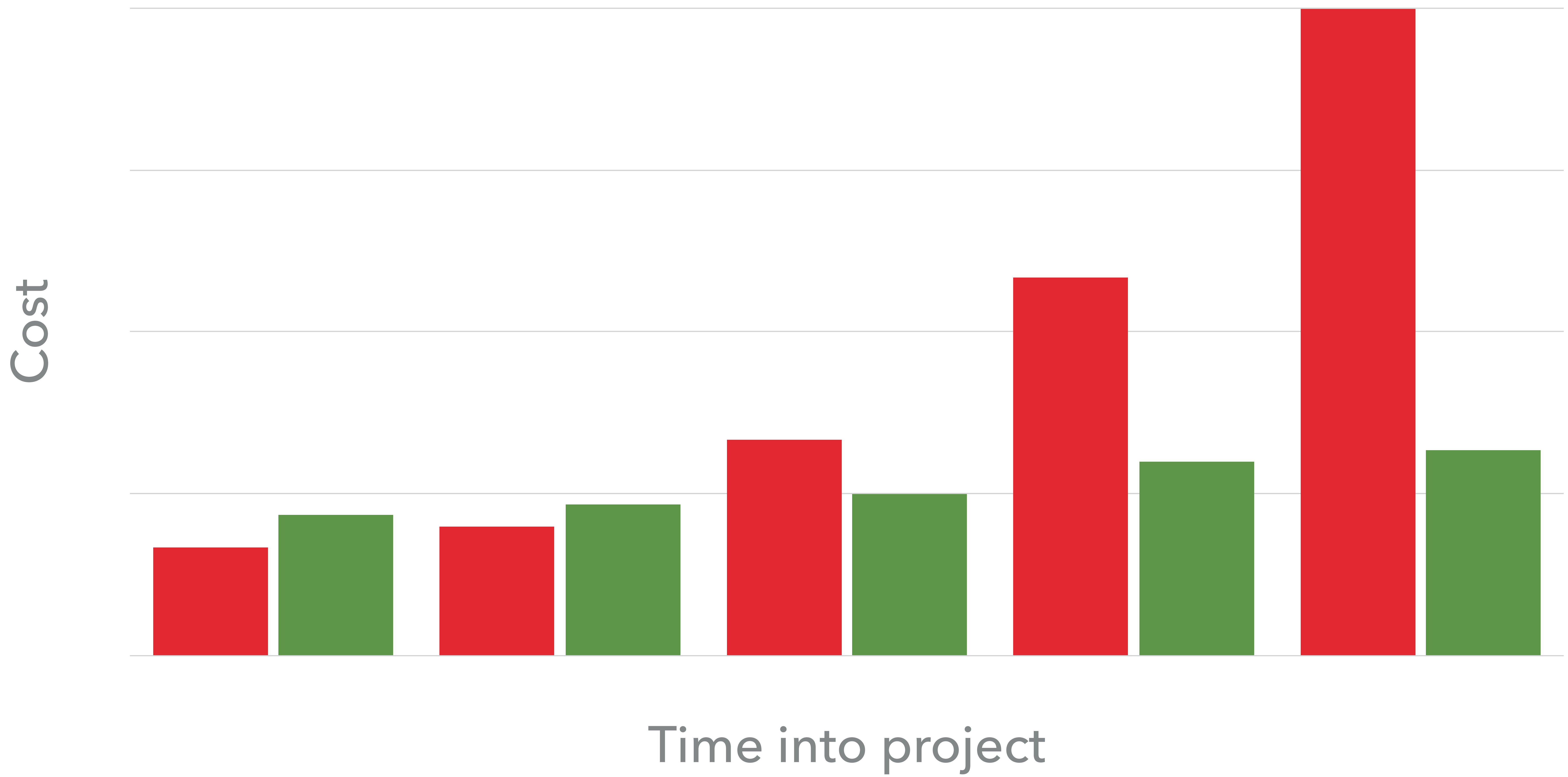
LOW EVOLVABILITY COSTS MONEY

⬆ 28% longer to implement new features [3]

⬆ 36% longer to fix bugs [3]

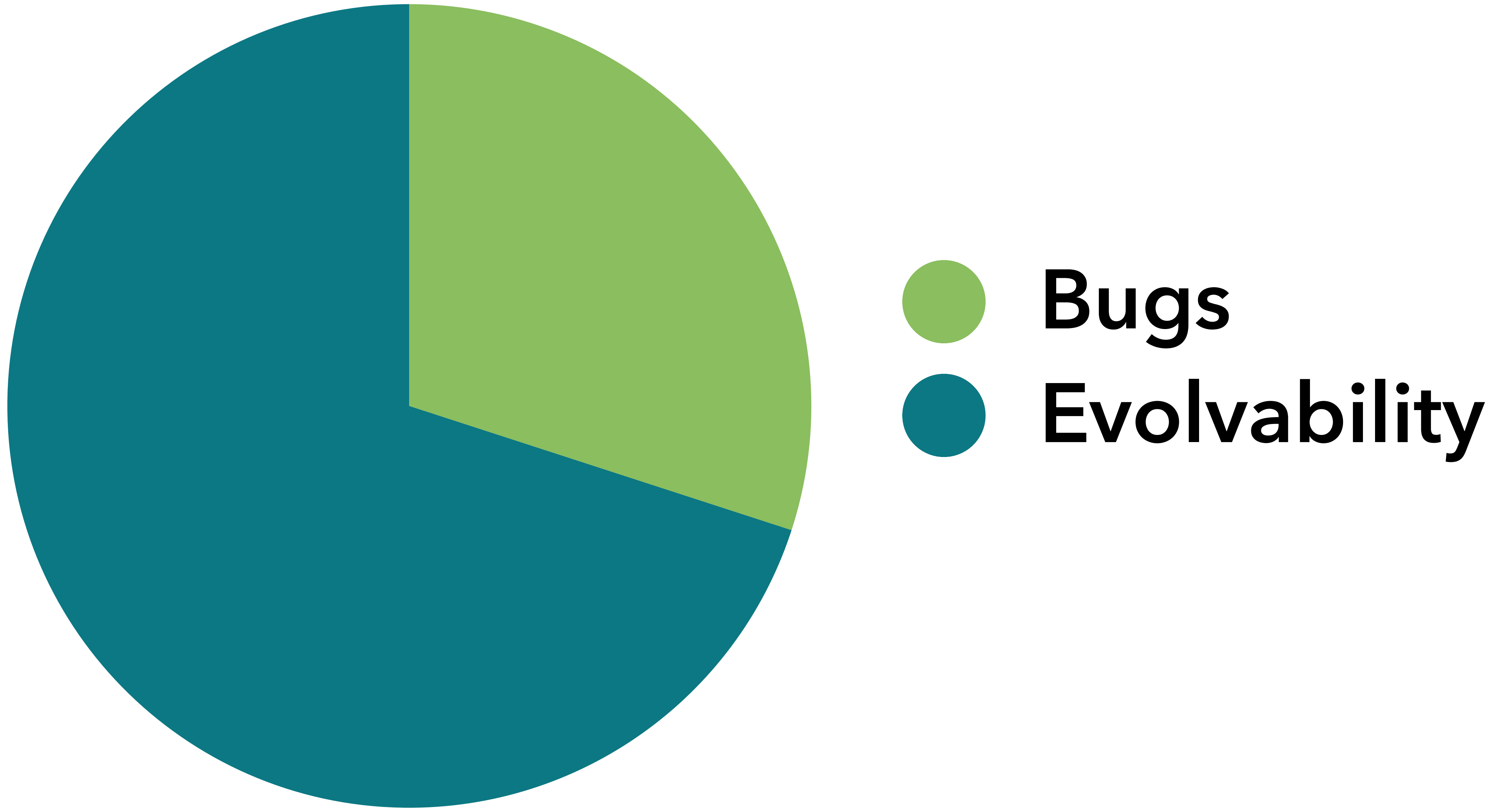
⬆ Software structure may account for 25% of total maintenance costs [4]

COST TO DEVELOP SIMILAR SIZED FEATURE OVER TIME



@davelidament

CODE REVIEW BENEFIT 1: FEWER DEFECTS



- OK not to find lots of “bugs”
- Remember to sell the right metric to management.

⬇ Reduce evolvability defects = ⬇ Reduce costs [1, 2, 3]

CODE REVIEW BENEFIT 2: SECURITY REVIEW

- Writing sensitive data to logs (e.g. password)
  - #[SensitiveParameter]
- Files that shouldn’t be there? (e.g. malware)
- OWASP top 10
  - OWASP top 10 cheat sheet
- Rolled your own authentication / hashing / encryption algorithms

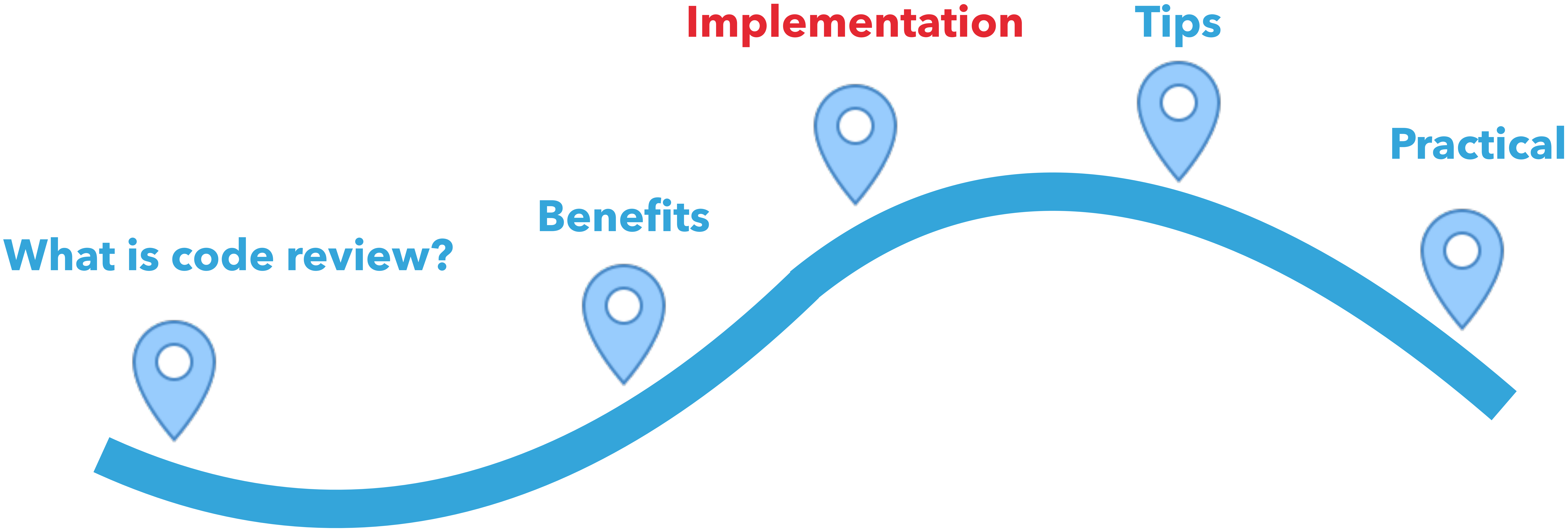
CODE BENEFIT 3: SPREAD THE KNOWLEDGE

CODE REVIEW BENEFIT 4: MENTORING

CODE REVIEW BENEFITS

- Reduce defects
  - Mainly evolvability defects
- Find security vulnerabilities
- Spread knowledge
- Mentoring

EFFECTIVE CODE REVIEW REDUCES OVERALL COST OF SOFTWARE DEVELOPMENT



What is code review?

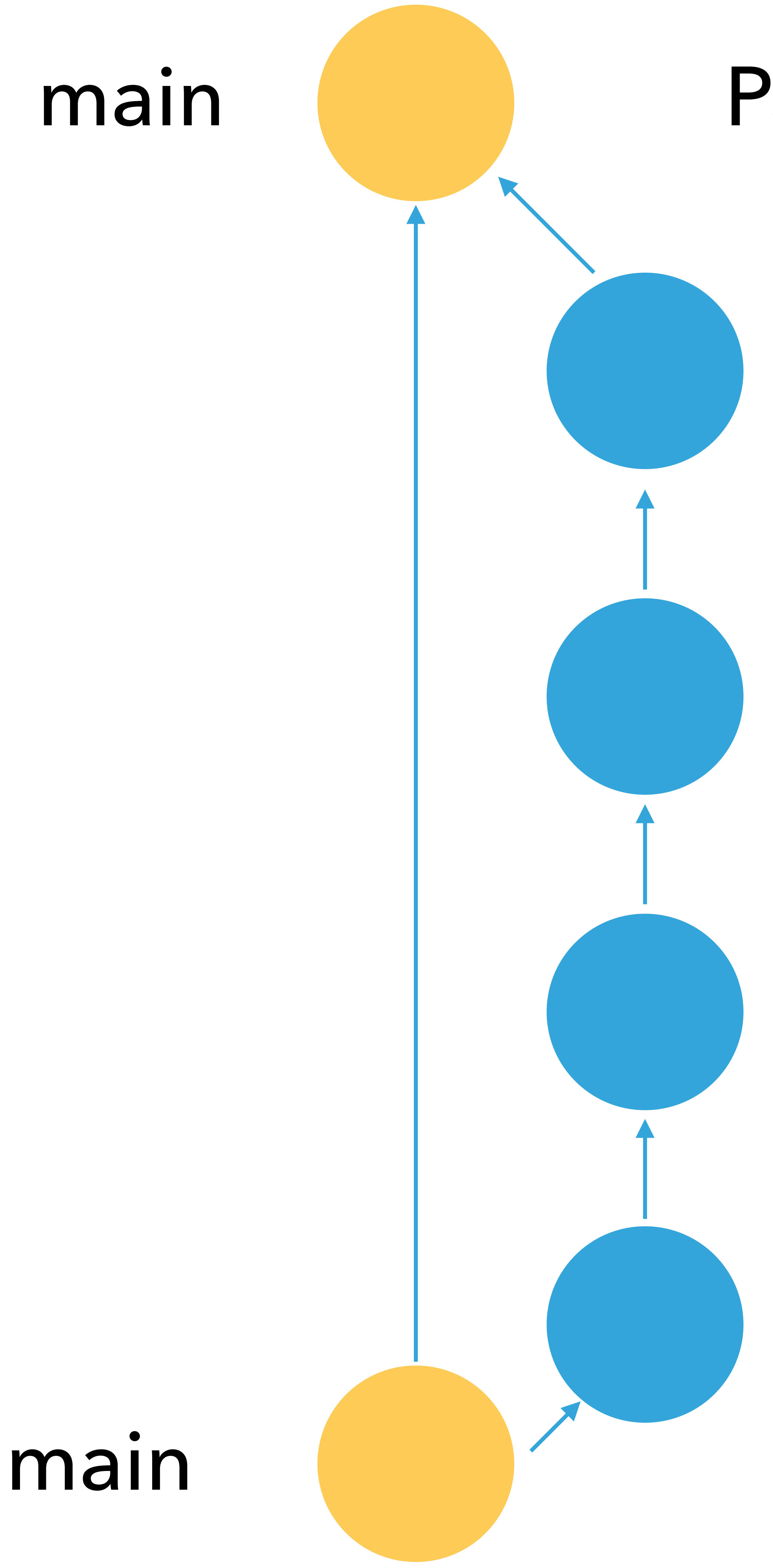
Benefits

Implementation

Tips

Practical

FEATURE BASED DEVELOPMENT

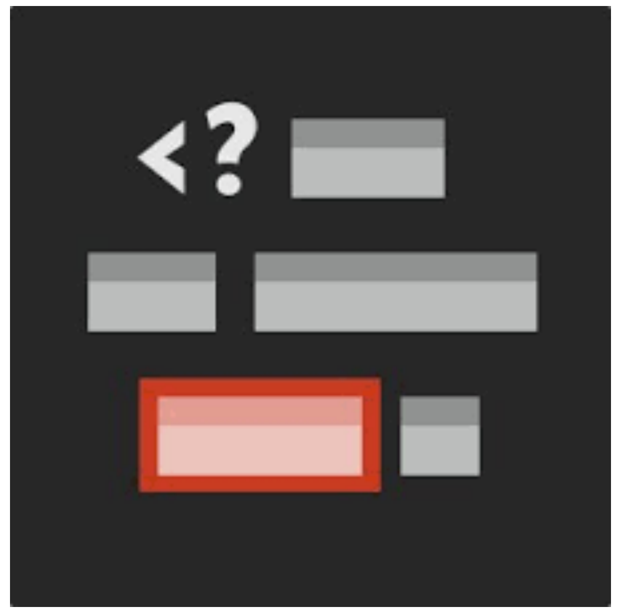


Passes code review and is merged

@daveliddament

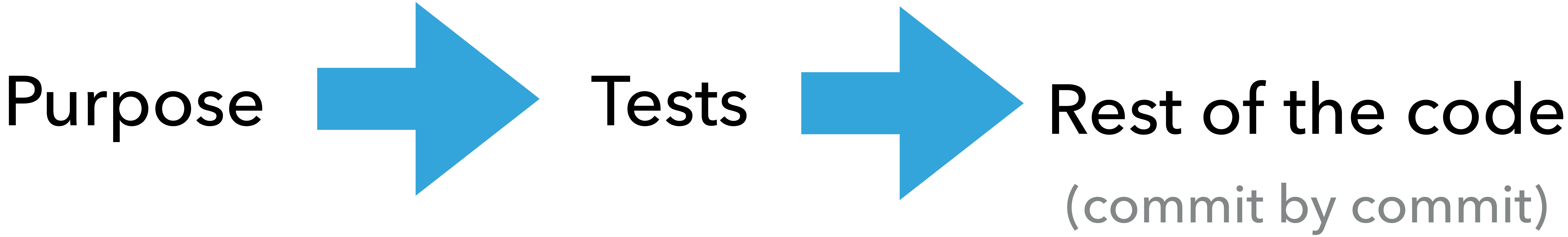
PHPUnit

GitHub Actions



@daveliddament

CODE REVIEW STEPS



Purpose → Tests → Rest of the code

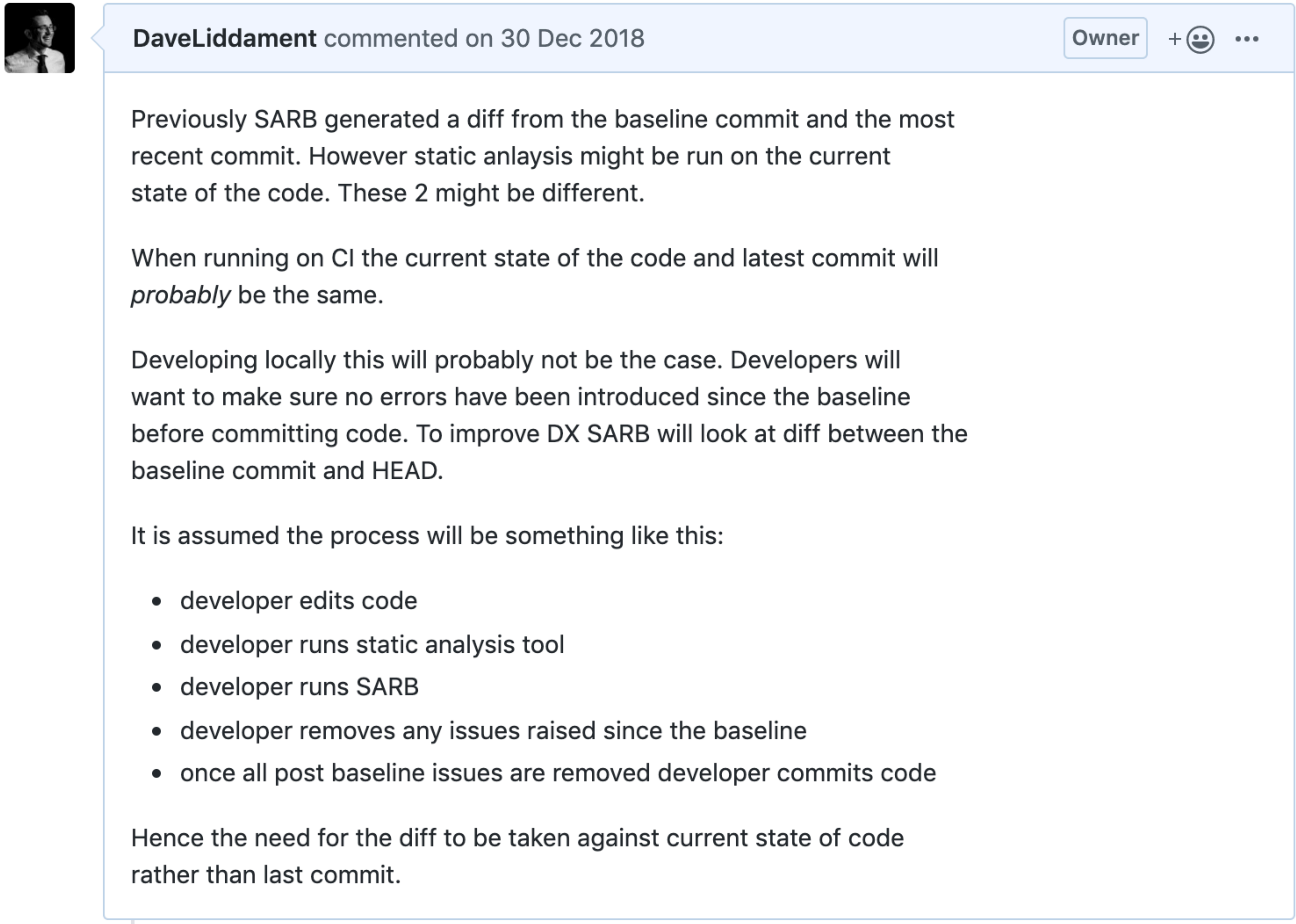
(commit by commit)

Bugs

Evolvability

Bugs

WHAT ARE WE REVIEWING?



DaveLiddament commented on 30 Dec 2018

Previously SARB generated a diff from the baseline commit and the most recent commit. However static anlaysis might be run on the current state of the code. These 2 might be different.

When running on CI the current state of the code and latest commit will *probably* be the same.

Developing locally this will probably not be the case. Developers will want to make sure no errors have been introduced since the baseline before committing code. To improve DX SARB will look at diff between the baseline commit and HEAD.

It is assumed the process will be something like this:

- developer edits code
- developer runs static analysis tool
- developer runs SARB
- developer removes any issues raised since the baseline
- once all post baseline issues are removed developer commits code

Hence the need for the diff to be taken against current state of code rather than last commit.

ARE THE TESTS TESTING THE RIGHT THING?

Input	Expected Output
my blog	my-blog
hello Dave	hello-Dave

ARE ALL TEST CASES COVERED?

Input	Expected Output
my blog	my-blog
it’s Saturday	its-saturday

WILL I UNDERSTAND THIS CODE IN 6 MONTHS?

WILL I UNDERSTAND THIS CODE IN 6 MONTHS?

✗

```
$userFields = [
    'Username',
    'Email',
    'FirstName',
    'LastName',
    'Phone',
];

foreach ($userFields as $key) {
    if ($userDetails->{'get'.$key}()) {
        $user->{'set'.$key}($userDetails->{'get'.$key}());
    }
}
```

WILL I UNDERSTAND THIS CODE IN 6 MONTHS? (2)

```
if ($userDetails->getUsername()) {
  $user->setUsername($userDetails->getUsername());
}
if ($userDetails->getEmail()) {
  $user->setEmail($userDetails->getEmail());
}
if ($userDetails->getFirstName()) {
  $user->setFirstName($userDetails->getFirstName());
}
if ($userDetails->getLastName()) {
  $user->setLastName($userDetails->getLastName());
}
if ($userDetails->getPhone()) {
  $user->setPhone($userDetails->getPhone());
}
```

✅

CAN WE REMOVE COMMENTS?

```
if ($this->messageSender->sendMessage($message) === true) {
    // 3 means message sent
    $message->setStatus(3);
}
```

CAN WE REMOVE COMMENTS?

```
class Message
{
  public const CREATED = 1;
  public const PENDING = 2;
  public const SENT = 3;

  … rest of class …
}


if ($this->messageSender->sendMessage($message) === true) {
  $message->setStatus(Message::SENT);
}
```

DO WE NEED COMMENTS?

```
/**
 * Populates template string containing placeholders with placeHolderValues
 *
 * Inputs of
 *    $template: “Hello {name}. Prepare to play {game}”
 *    $values: [“name” => “Jane”, “game” => “monopoly”]
 * Returns: “Hello Jane. Prepare to play monopoly”
 *
 * Optional values are marked with a ?
 * E.g. inputs of
 *    $template: “Hello {name}. Prepare to play {game?}”
 *    $values: [“name” => “Bob”]
 * Returns: “Hello Bob. Prepare to play ”
 *
 * @param array<string,string> $placeHolderValues
 * @throws MissingPlaceHolderValue
 */
function populateTemplate(  string $template,
                            array $placeHolderValues,
):string { … }
```

ARE COMMENTS UP TO DATE?

HOW DO WE MAKE THIS MORE OBVIOUS?

```
class MarketingCampaign
{

  public function addAddress(
    string $address
  ): void {
    .. some implementation ..
  }
}
```

HOW DO WE MAKE THIS MORE OBVIOUS (2)

```
class MarketingCampaign
{

  public function addEmailAddress(
    string $emailAddress
  ): void {
    .. some implementation ..
  }

}
```

✗

HOW DO WE MAKE THIS MORE OBVIOUS (3)

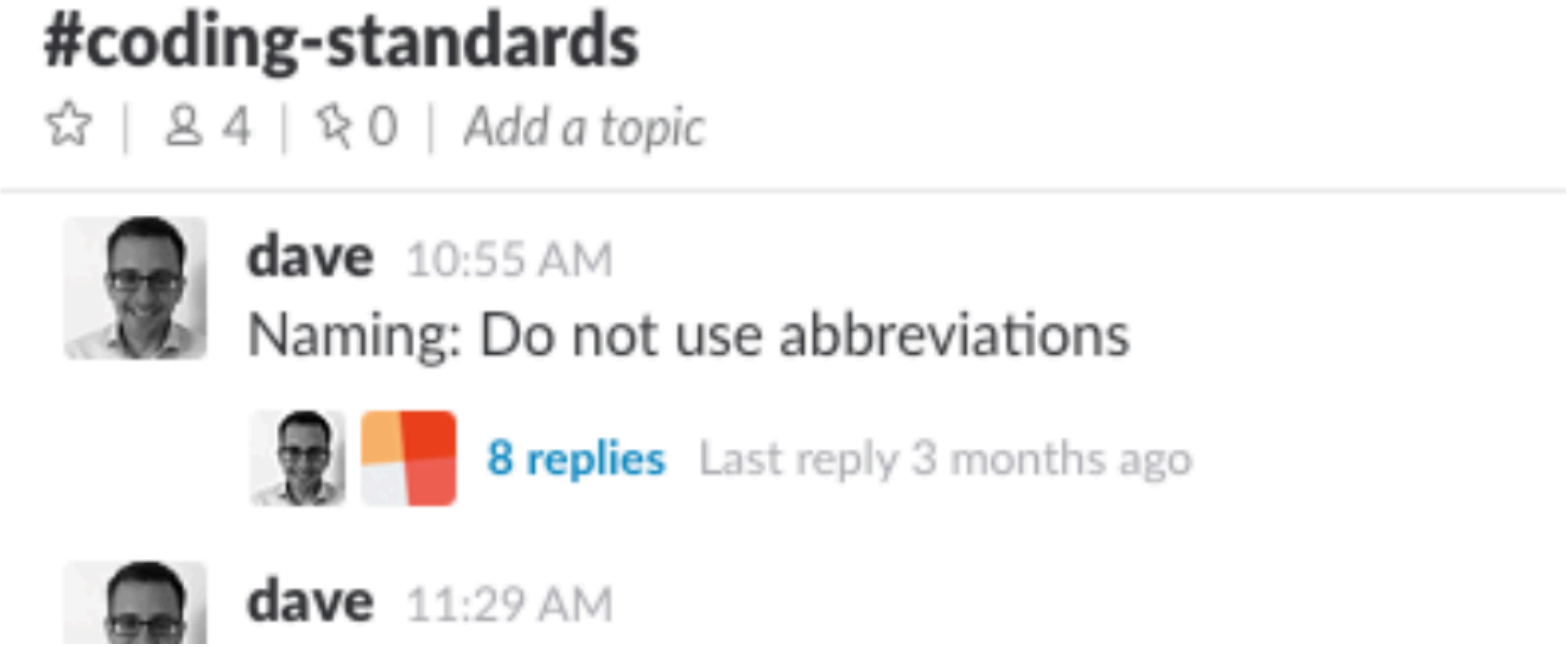
```
class MarketingCampaign
{

  public function addEmailAddress(
    EmailAddress $emailAddress
  ): void {
    .. some implementation ..
  }

}
```

✅

ARE WE FOLLOWING PROJECT CONVENTIONS?

```
interface LocationRepository
{

  public function findClosestTo($point);

  public function findByName($name);

  public function findBySlug($slug);

  public function searchForLocation($name, $type);

  public function findAllByType($type);

}
```

DOCUMENT PROJECT CONVENTIONS



#coding-standards

☆ | 4 | 0 | *Add a topic*

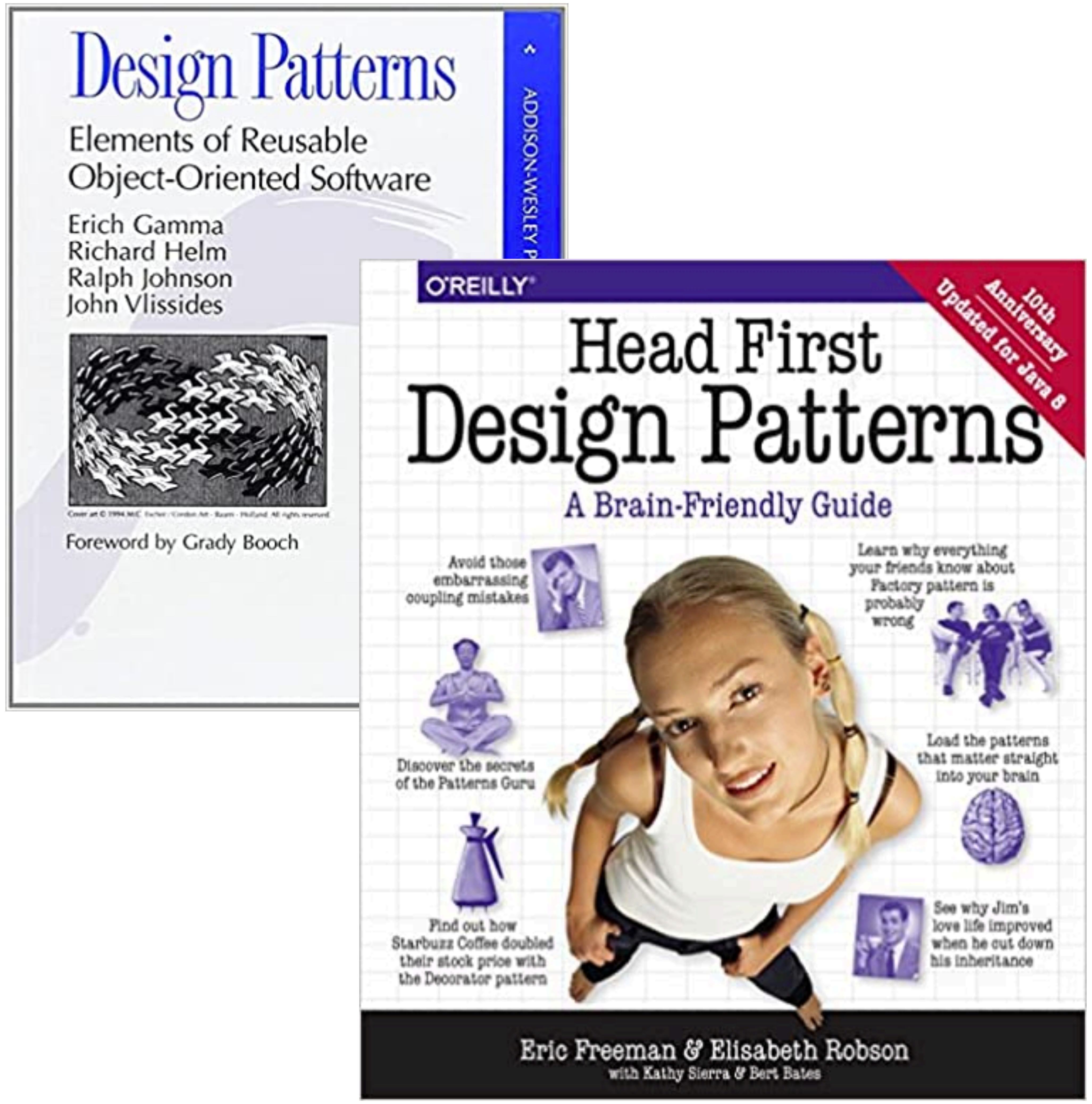
dave 10:55 AM

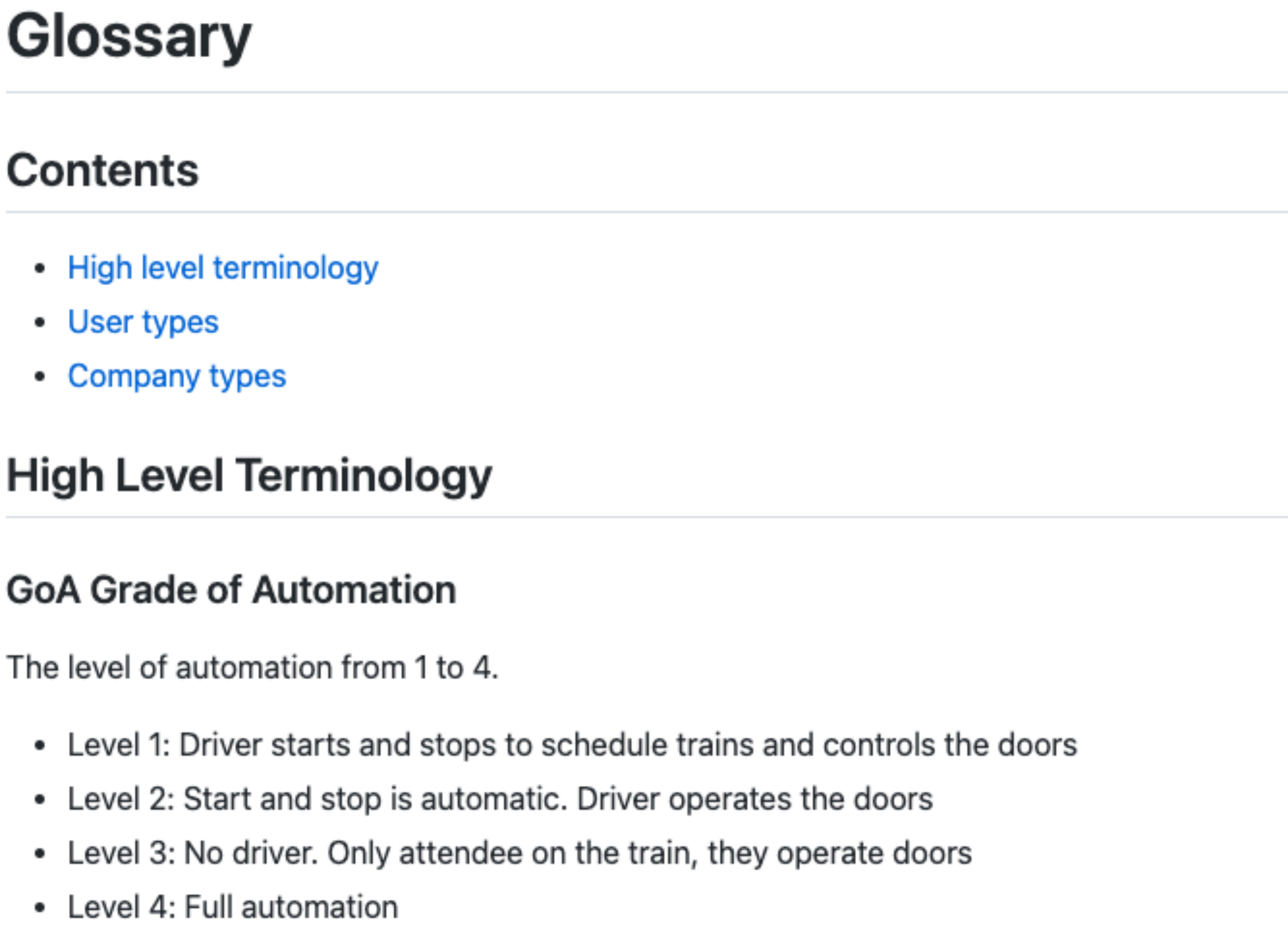
Naming: Do not use abbreviations

8 replies Last reply 3 months ago

dave 11:29 AM

CORRECT NAMING


Design Patterns
Elements of Reusable Object-Oriented Software
Erich Gamma
Richard Helm
Ralph Johnson
John Vlissides
Foreword by Grady Booch
ADDISON-WESLEY
O'REILLY
10th Anniversary
Updated for Java 8
Head First Design Patterns
A Brain-Friendly Guide
Avoid those embarrassing coupling mistakes
Learn why everything your friends know about Factory pattern is probably wrong
Discover the secrets of the Patterns Guru
Load the patterns that matter straight into your brain
Find out how Starbuzz Coffee doubled their stock price with the Decorator pattern
See why Jim's love life improved when he cut down his inheritance
Eric Freeman & Elisabeth Robson
with Kathy Sierra & Bert Bates



Glossary

Contents

- High level terminology
- User types
- Company types

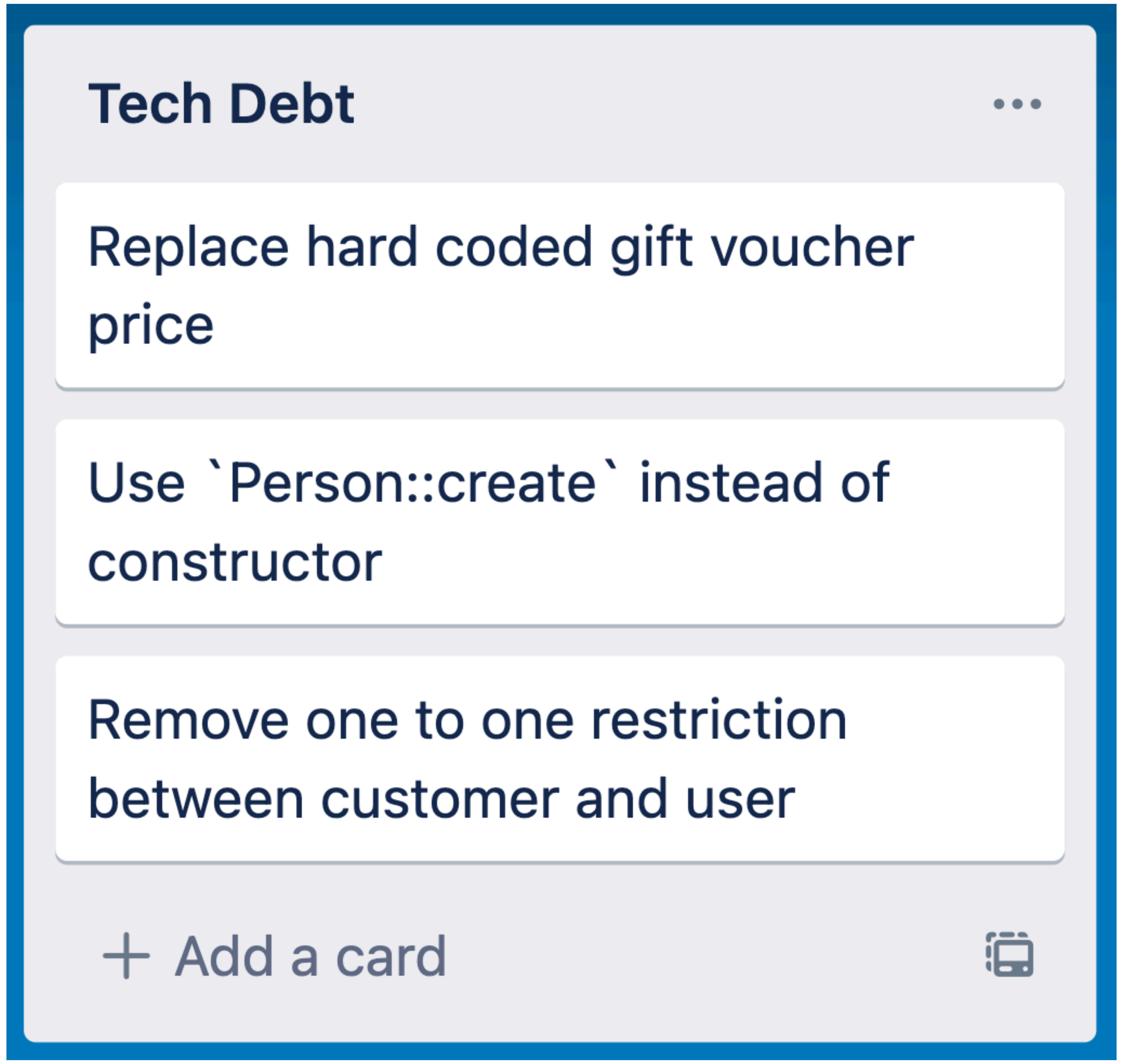
High Level Terminology

GoA Grade of Automation

The level of automation from 1 to 4.

- Level 1: Driver starts and stops to schedule trains and controls the doors
- Level 2: Start and stop is automatic. Driver operates the doors
- Level 3: No driver. Only attendee on the train, they operate doors
- Level 4: Full automation

HAS TECHNICAL DEBT BEEN DOCUMENTED?



Tech Debt

Replace hard coded gift voucher price

Use `Person::create` instead of constructor

Remove one to one restriction between customer and user

```
// TODO https://trello.com/c/Aaa123
// Refactor to method

… some hacky code …
```

@daveliddament

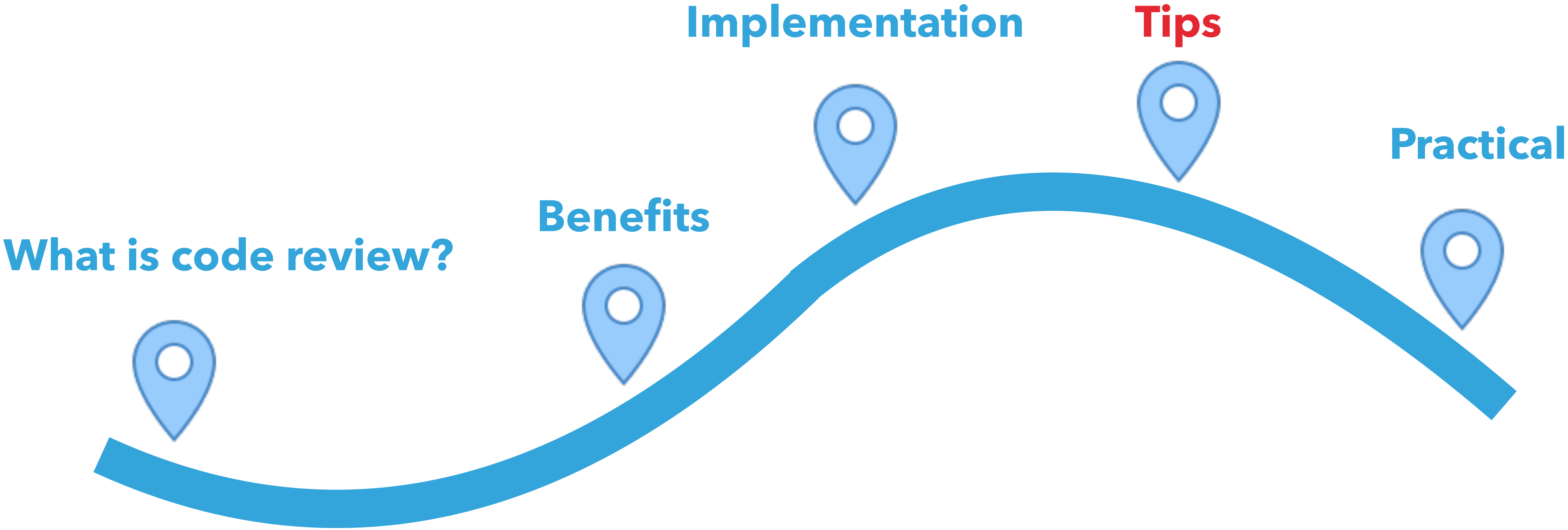
IS THE ARCHITECTURE GOOD?

ARE THERE ANY BUGS?!

CHECK LIST

- What are we reviewing?
- Do the tests fully test the required functionality?
- Will I understand this code in 6 months?
- Do comments match the code?
- Is the code obvious and explicit?
- Does the code follow project conventions?
- Has technical debt been documented?
- Can architecture be improved?
- Are there any bugs?

EVERYONE SHOULD CODE REVIEW



What is code review?

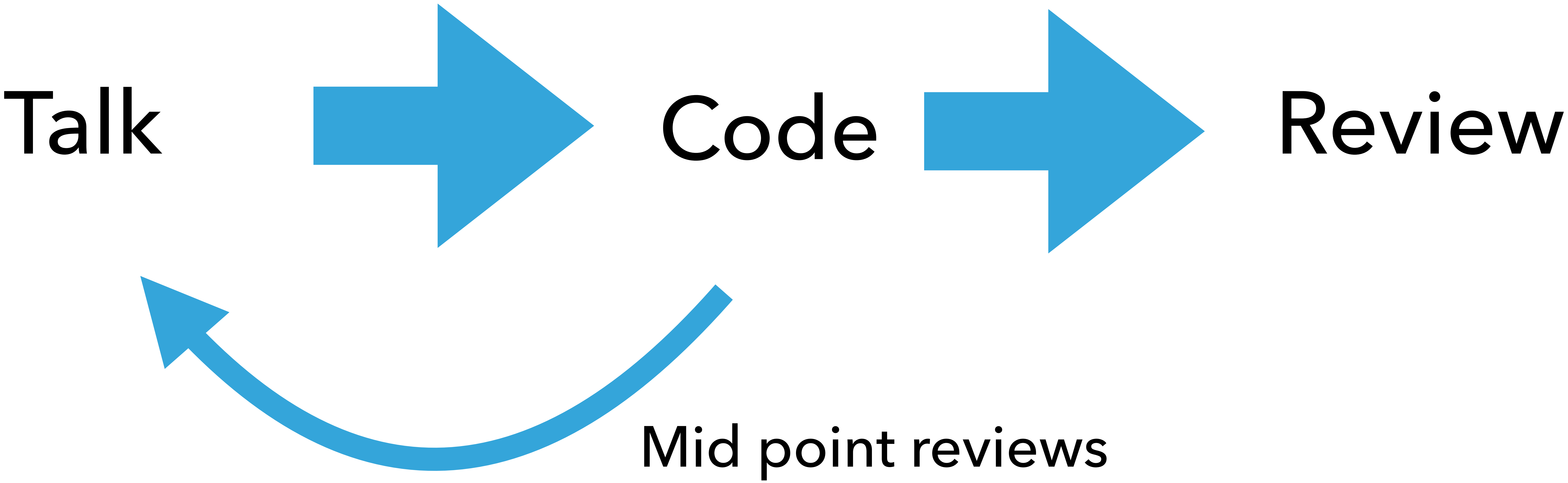
Benefits

Implementation

Tips

Practical

CODE REVIEW



Talk

Code

Review

Mid point reviews

> ASK PROGRAMMERS TO REVIEW 10 LINES OF CODE THEY’LL FIND 10 ISSUES…
>
> ASK THEM TO DO 500 LINES THEY’LL SAY IT’S GOOD TO GO

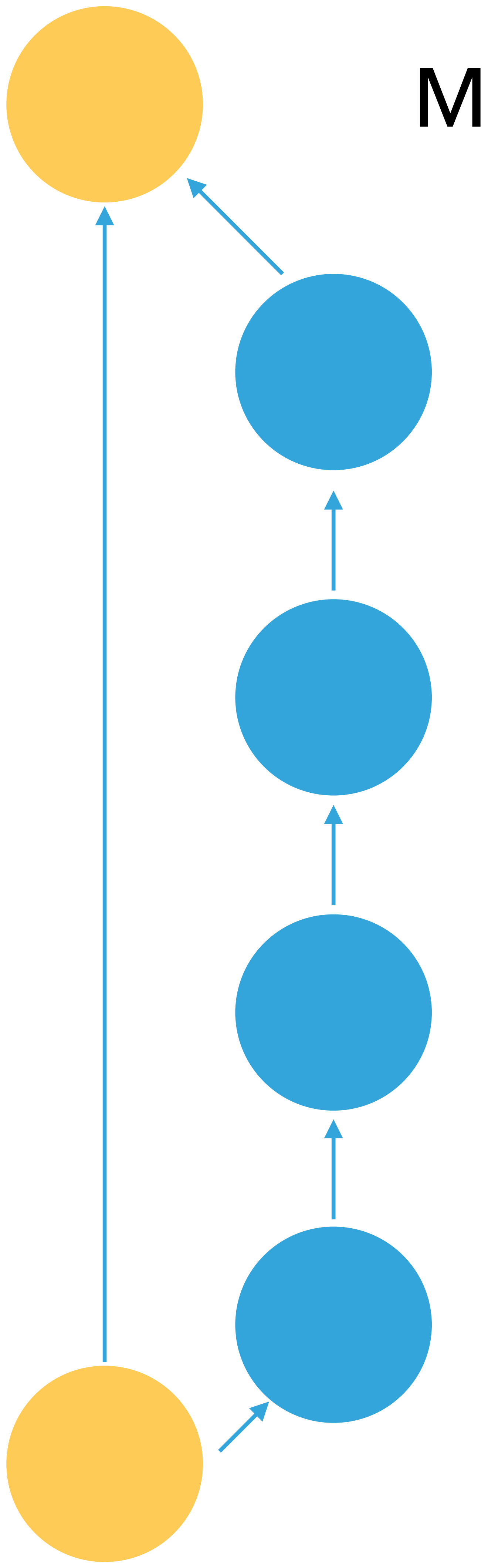
Anyone who’s done code review

HOW MUCH SHOULD YOU REVIEW IN ONE GO?

Max 1 hour review at a time [5]

Fewer than 400 lines of code at a time [5]

STORY OF ATOMIC COMMITS



MERGE: Use new calculation service: https://trello.com/a/1234

REMOVE: Deprecated price calculation service

UPDATE: Use new price calculator code

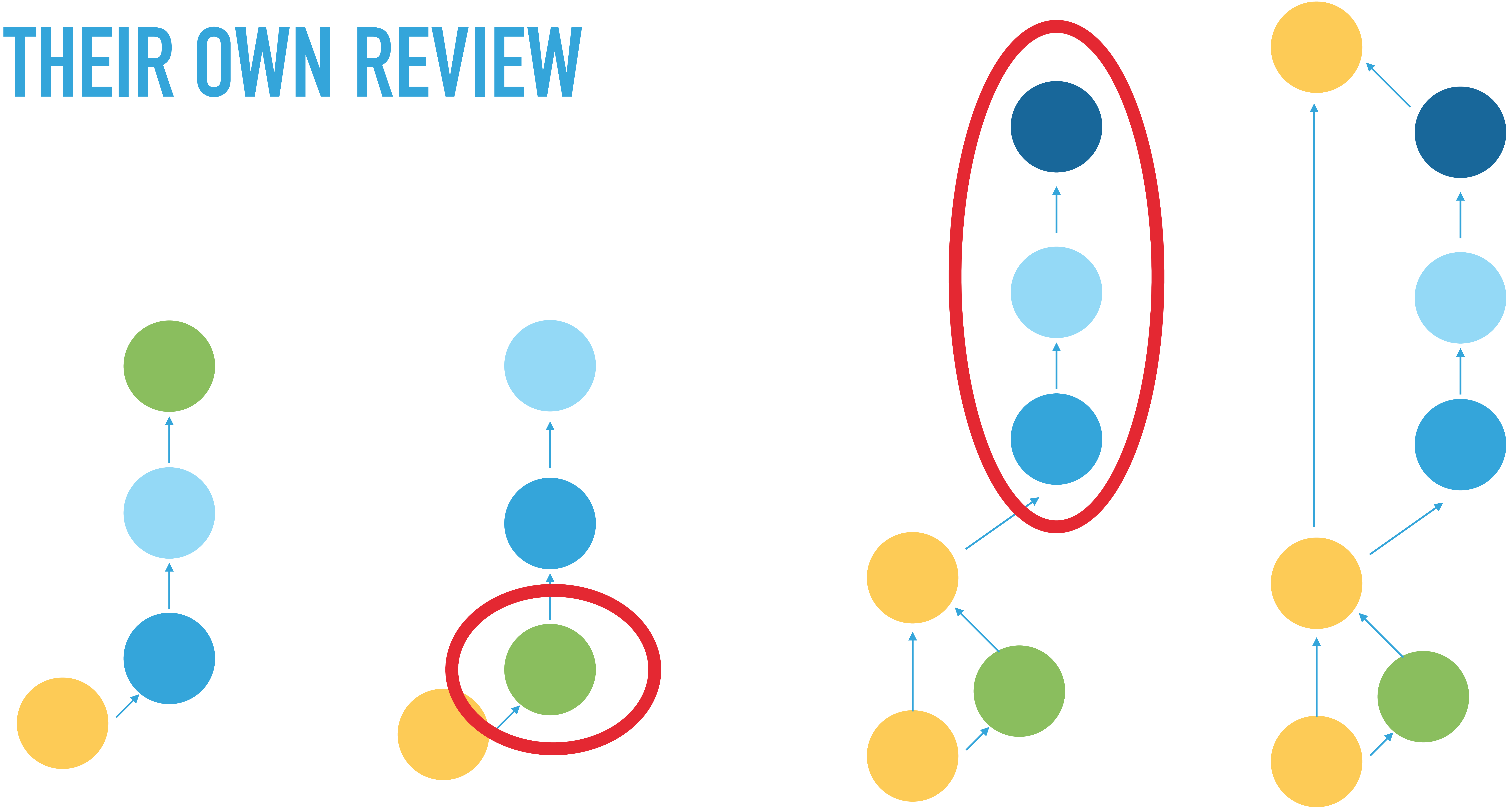
ADD: Facade to 3rd party price calculation service

DEPRECATE: The price calculation service

ATOMIC COMMITS (2)

- Use individual commits for:
  - whitespace changes
  - class renames or moves
  - method or property renames
  - composer updates

PULL OUT REFACTORS TO THEIR OWN REVIEW



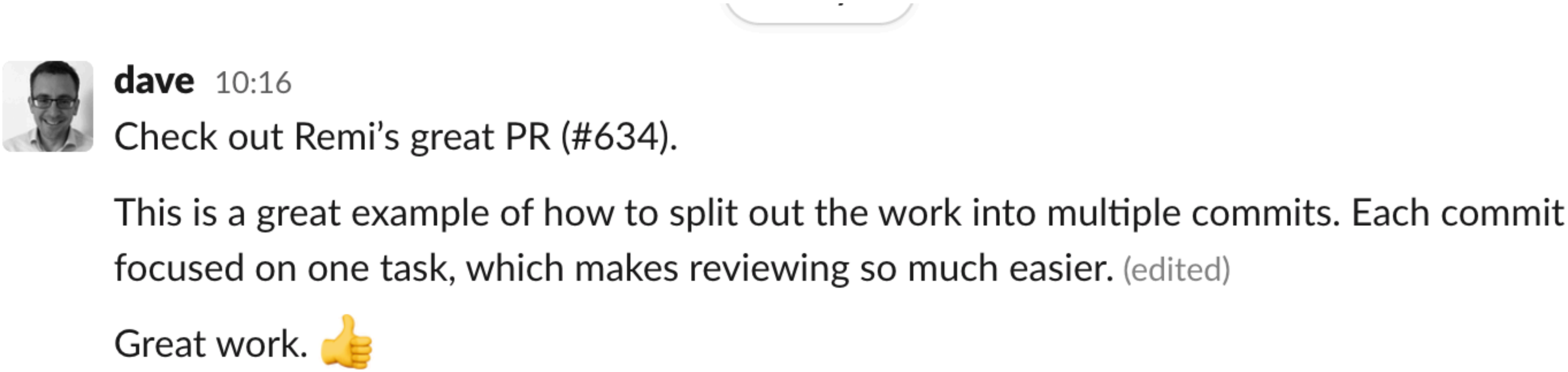
@daveliddament

GOOD REVIEW COMMENTS

- Don’t be an idiot. Don’t be rude.
- Not critical of the author. (Use we rather than you).
- State problem and solution. (Maybe as a commit).
- Link to Stack Overflow, blog, etc.
- Use: “Let’s chat”.
- Use: “Question”.
- Compliment.

RECEIVING REVIEW COMMENTS

- Don't be an idiot. Don't be rude
- Don't take offence
- Do say if you disagree
- Compliment



dave 10:16

Check out Remi's great PR (#634).

This is a great example of how to split out the work into multiple commits. Each commit focused on one task, which makes reviewing so much easier. (edited)

Great work. 👍

KEEP ON TOP OF CODE REVIEWS

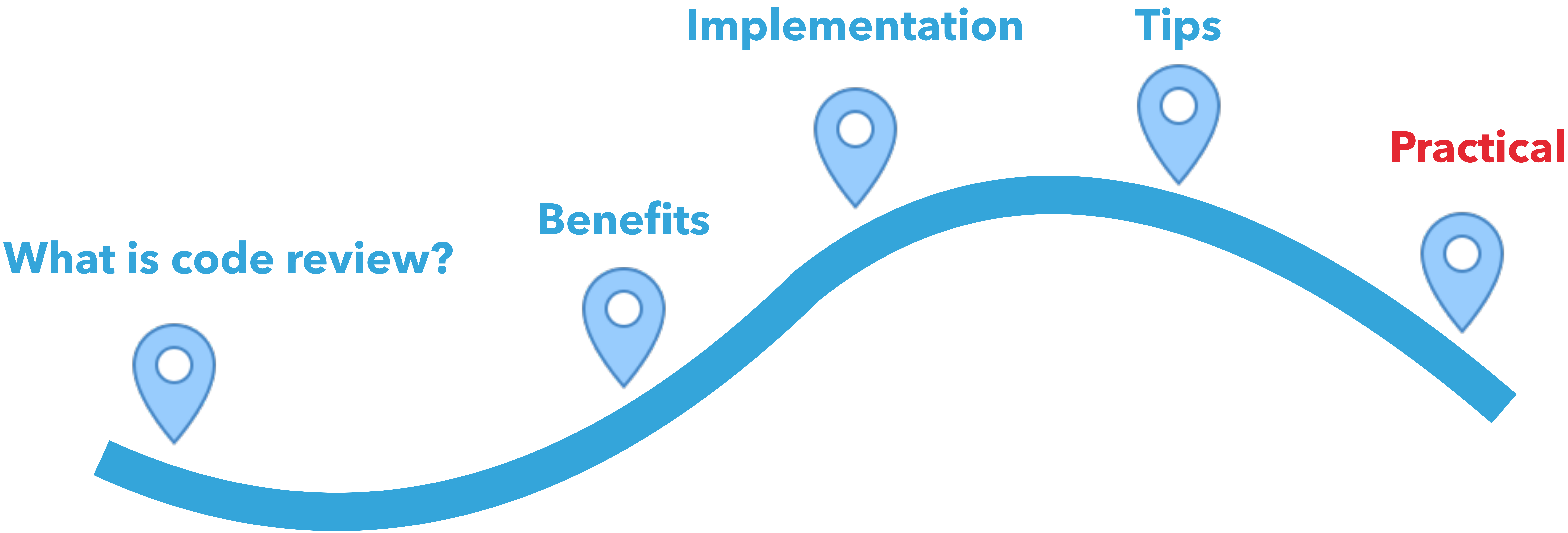
- Code review > new code
- 30 to 60min focus blocks

PRAGMATIC USE OF CODE REVIEW

- Default: Review everything
- Don’t review:
  - Small changes
  - Common refactors

CODE REVIEW TIPS

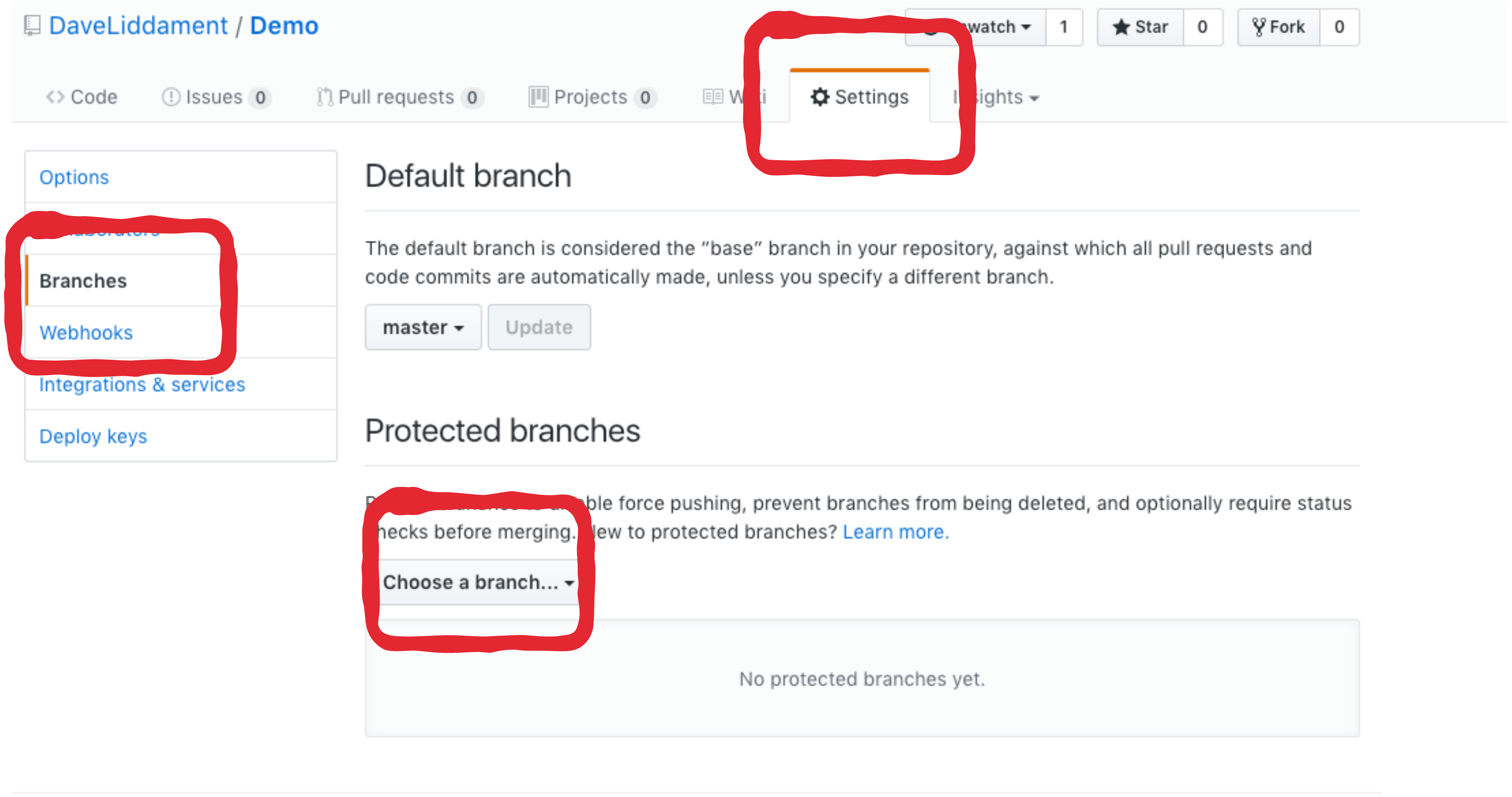
- Plan and communicate before you code
- Keep code to be reviewed small in quantity
- Be constructive in code review comments
- Prioritise code review over new work
- Be pragmatic



What is code review?
Benefits
Implementation
Tips
Practical

CODE CAN ONLY BE DEPLOYED IF:

- CI passes
- Code review passes



Default branch

The default branch is considered the "base" branch in your repository, against which all pull requests and code commits are automatically made, unless you specify a different branch.

master ▾ Update

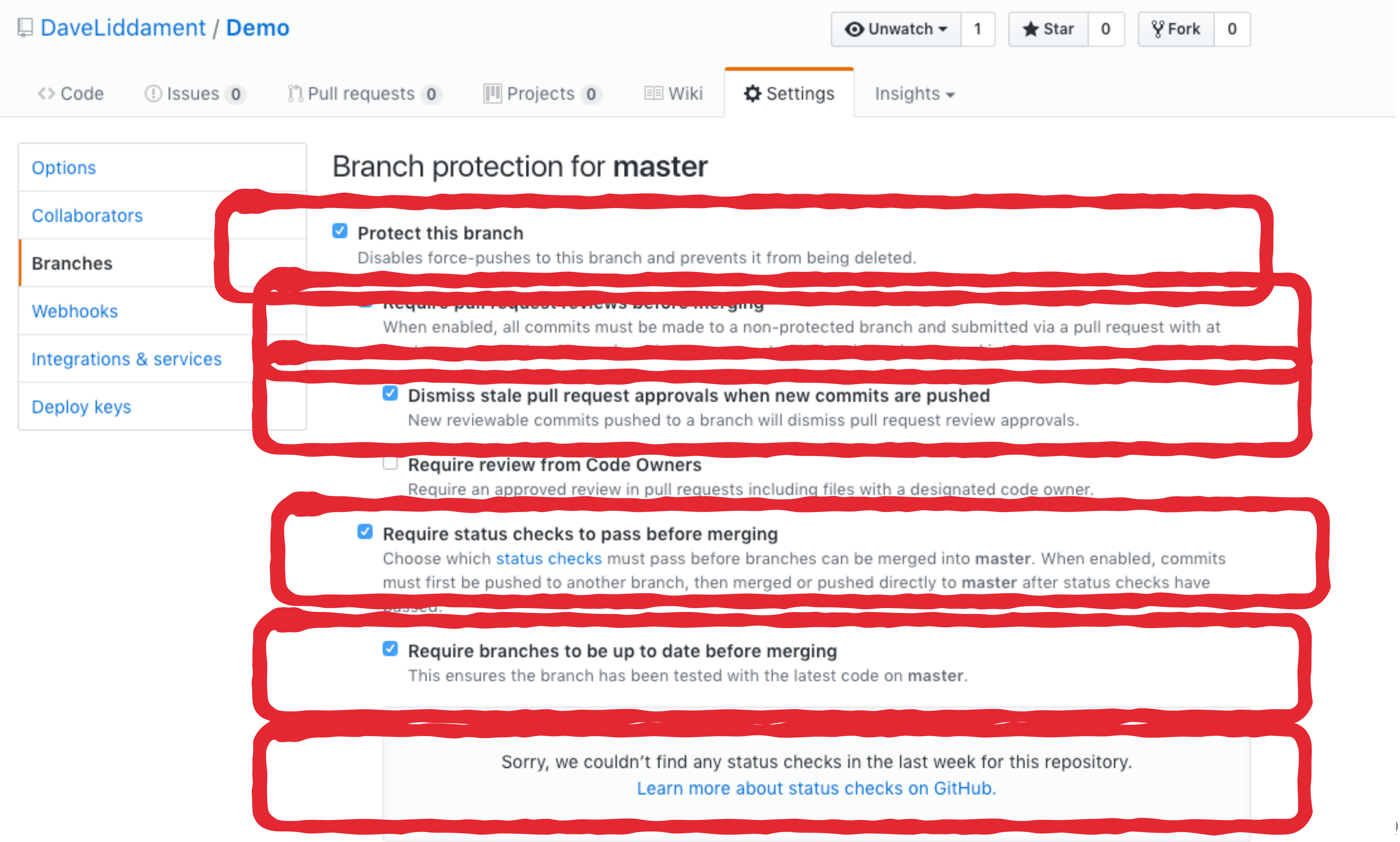
Protected branches

P... ble force pushing, prevent branches from being deleted, and optionally require status hecks before merging. lew to protected branches? Learn more.

Choose a branch... ▾

No protected branches yet.

@daveliddament



DaveLiddament / **Demo**

Branch protection for **master**

☑ **Protect this branch**
Disables force-pushes to this branch and prevents it from being deleted.

☑ **Require pull request reviews before merging**
When enabled, all commits must be made to a non-protected branch and submitted via a pull request with at

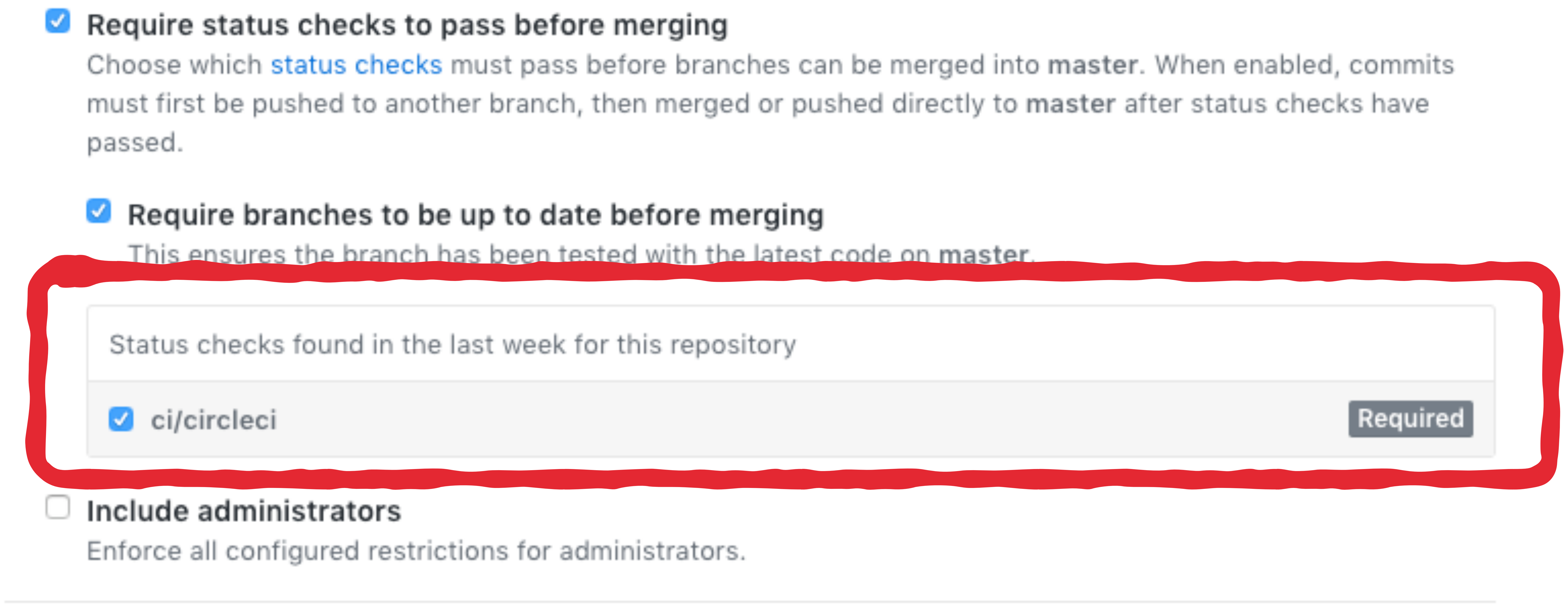
☑ **Dismiss stale pull request approvals when new commits are pushed**
New reviewable commits pushed to a branch will dismiss pull request review approvals.

□ **Require review from Code Owners**
Require an approved review in pull requests including files with a designated code owner.

☑ **Require status checks to pass before merging**
Choose which status checks must pass before branches can be merged into **master**. When enabled, commits must first be pushed to another branch, then merged or pushed directly to **master** after status checks have passed.

☑ **Require branches to be up to date before merging**
This ensures the branch has been tested with the latest code on **master**.

Sorry, we couldn't find any status checks in the last week for this repository.
Learn more about status checks on GitHub.



☑ **Require status checks to pass before merging**

Choose which status checks must pass before branches can be merged into **master**. When enabled, commits must first be pushed to another branch, then merged or pushed directly to **master** after status checks have passed.

☑ **Require branches to be up to date before merging**

This ensures the branch has been tested with the latest code on **master**.

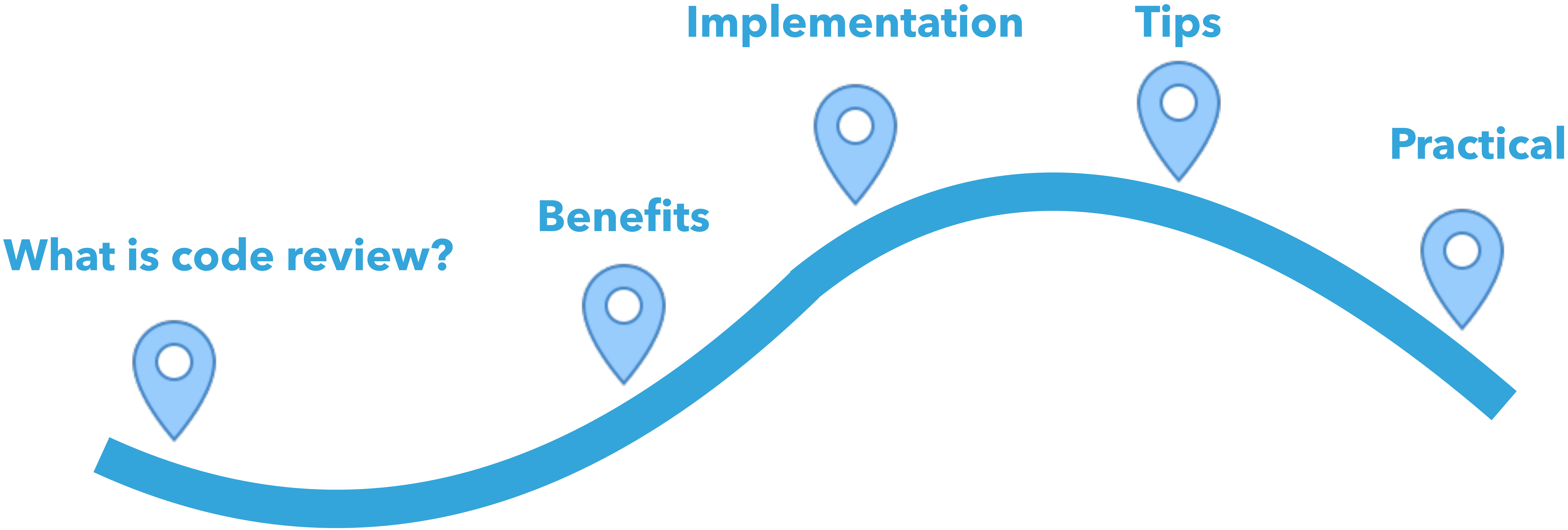
Status checks found in the last week for this repository

☑ ci/circleci Required

☐ **Include administrators**

Enforce all configured restrictions for administrators.

INTEGRATING CODE REVIEW INTO PROJECT WORKFLOW IS EASY!



What is code review?

Benefits

Implementation

Tips

Practical

EFFECTIVE CODE REVIEW REDUCES OVERALL COST OF SOFTWARE DEVELOPMENT

Dave Liddament

Lamp Bristol

Thank you for listening

Organise PHP-SW meetup

Author of Static Analysis Results Baseliner (SARB)

And PHP Language Extensions library

20 years of writing software (C, Java, Python, PHP)

@daveliddament

REFERENCES

- [1] Mika V. Mantyla and Casper Lassenius “What Types of Defects Are Really Discovered in Code Reviews?” IEEE Transactions on Software Engineering
- [2] Harvey Siy, Lawrence Votta “Does The Modern Code Inspection Have Value?”
- [3] R.K. Bandi, V.K. Vaishnavi, and D.E. Turk, “Predicting Maintenance Performance Using Object-Orientated Design Complexity Metrics”
- [4] R.D. Banker, S.M. Datar, C.F. Kemerer, and D. Zweig, “Software Complexity and Maintenance Costs,”
- [5] https://smartbear.com/learn/code-review/best-practices-for-peer-code-review/