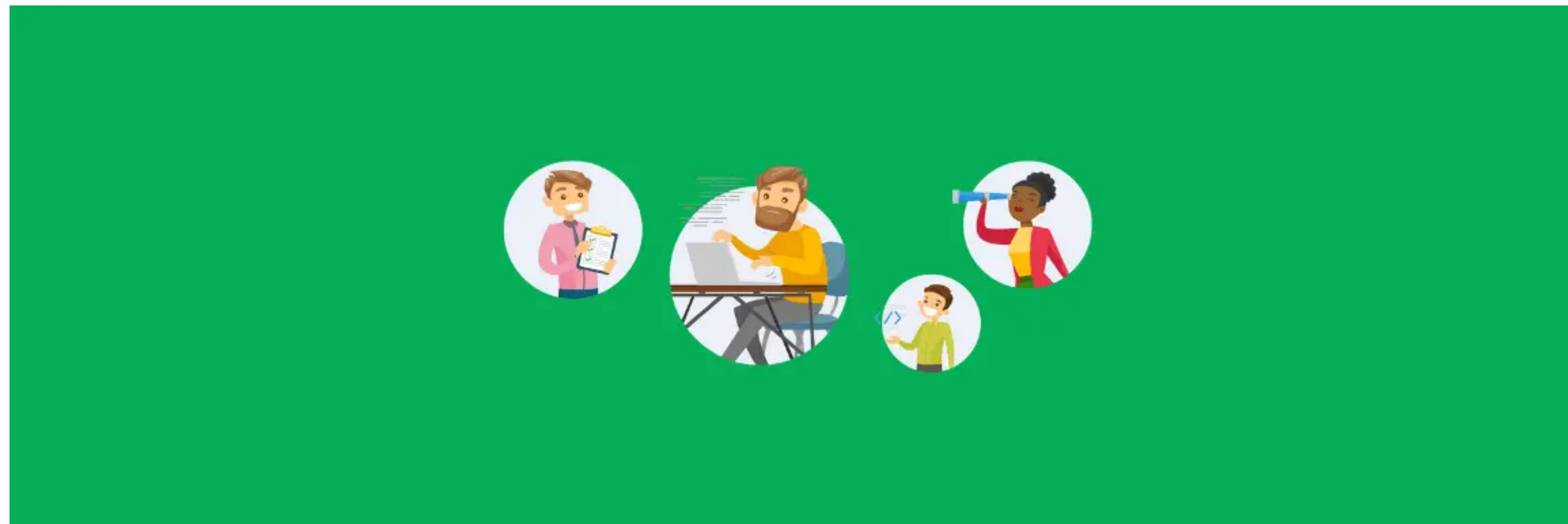


Automating Code Reviews on GitHub

by **Julien Delange** · January 19, 2020



Code reviews are part of the daily activities of software engineers and a key process in [release management](#). With engineers spending 10% to 20% of their time on code reviews, automating code reviews (at least part of) allows them to focus on other tasks.

In addition, [automating](#) code reviews guarantee consistency across reviews and unblocks developers waiting for a review. It significantly increases developer velocity while reducing engineering costs.

Automate Code Reviews

In this article, we will explain step by step how to automate code reviews on [Github](#) using Code Inspector, a code analysis platform that empowers developers to write better software. Code Inspector offers a function to automate code reviews that detect design, security, safety, good practice enforcement issues in code, as well as duplicates of complex functions.

Step 1: Install the GitHub App on your repository

[Marketplace](#) [Apps](#) [Code Inspector](#)

ci

Verified by GitHub

GitHub confirms that this app meets the requirements for verification.

Categories

Code quality

Code review

Free

Paid

Supported languages

C, C++, Java
and 7 other languages supported

Developer

ci

xcodinglabs

Developer links

Support

Documentation

Privacy Policy

Terms of Service

Report abuse

Application

Code Inspector

Set up a plan

Code Inspector is a code analysis platform that does automated code reviews, technical debt management and analysis of code quality trends over time. The platform aggregates multiple quality metrics (violations, duplicates, readability, complexity). The platform reports the \$ figure of the technical debt and show trends of your code base. Code Inspector reports the most critical issues by distinguishing them according to their category, severity and location.

Read more...

stray(+, "blabla");

Check failure on line 6 in toto.c

Code Inspector / Code Inspector - Pull Request Code Review (Beta)

Security issue

Uninitialized variable: s

Check warning on line 6 in toto.c

Code Inspector / Code Inspector - Pull Request Code Review (Beta)

Security issue

Does not check for buffer overflows when copying to destination [MS-banned] (CWE-129)

Integrated Code Review on GitHub Pull Requests

Pricing and setup

Basic

Most popular plan: code analysis and technical debt management in one platform

\$0

Silver

Ideal for small to medium companies

\$10
per user / month

Gold

Large organizations with extended historical data and premium support

\$18
per user / month

Code Inspector

Basic

Most popular plan: code analysis and technical debt management in one platform

5 repositories with 100Kslocs/repositories

10 analyses per day

Automated code reviews on your github pull request

Code Quality Score and Technical Debt Evaluation

Install it for free

Next: Confirm your installation location.

Code Inspector is provided by a third-party and is governed by separate [terms of service](#), [privacy policy](#), and [support documentation](#).

Then, click on “Complete order and begin installation” as shown below.

Search or jump to...

[Pull requests](#) [Issues](#) [Marketplace](#) [Explore](#)

Review your order

Code Inspector

Basic

Most popular plan: code analysis and technical debt management in one platform

5 repositories with 100Kslocs/repositories

10 analyses per day

Automated code reviews on your github pull request

Code Quality Score and Technical Debt Evaluation

\$0 / month

Order total

Basic

\$0.00 / month

Due today

Prorated for Dec 15th-Jan 14th

\$0.00

Billing information

codeinspectordemo

Personal account

By clicking "Complete order and begin installation", you are agreeing to the [Marketplace Terms of Service](#), [Code Inspector's Terms of Service](#) and the [Privacy Policy](#).


Complete order and begin installation

Next: Authorize Code Inspector to access your account.

© 2019 GitHub, Inc. [Terms](#) [Privacy](#) [Security](#) [Status](#) [Help](#)

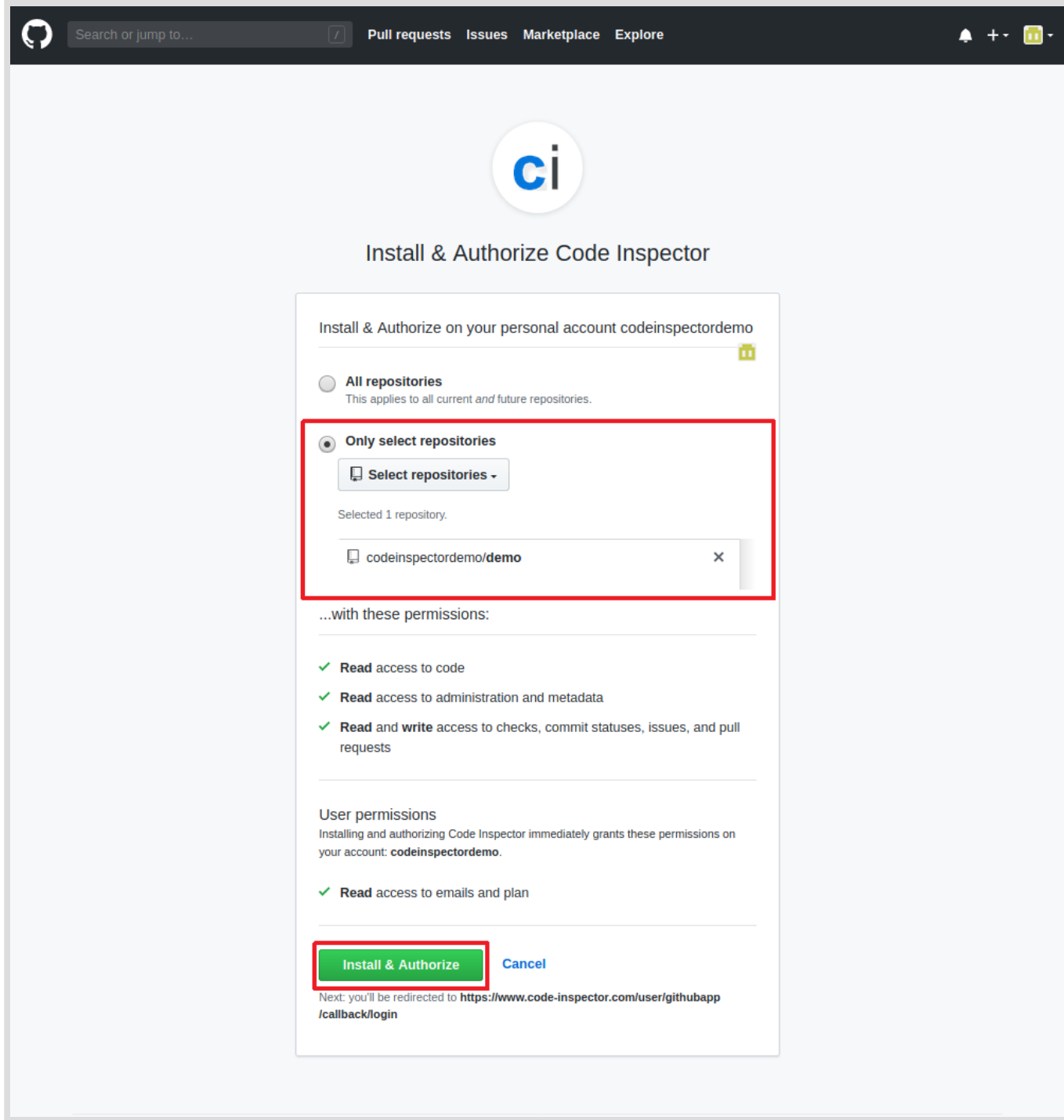
[Contact GitHub](#) [Pricing](#) [API](#) [Training](#) [Blog](#) [About](#)

Finally, choose the repository you want to enable the automated code repository and

 devops

[Learn](#) [Resources](#) [News](#) [Newsletter](#) [Certification Guides](#)

START HERE



Step 2: Push a Pull Request

To demonstrate the capabilities, we will start with a small Python project that has just a few lines of code. We will voluntarily put some errors.

In the terminal, go in an empty repository. We will assume you have a repository, all the commands below must be typed in the directory that contains the repository.

You Might Like: [Jenkins Shared Library Tutorial For Beginners](#)

Before we start to write any code, let's switch to a new branch, called code-review-demo

```
git checkout -b code-review-demo
```

Let's write a very small Python program that sums two numbers. We write the following code in the file main.py.

```
print("2 + 2 = {}".format(sumTwoTerms(2, 2)))
```

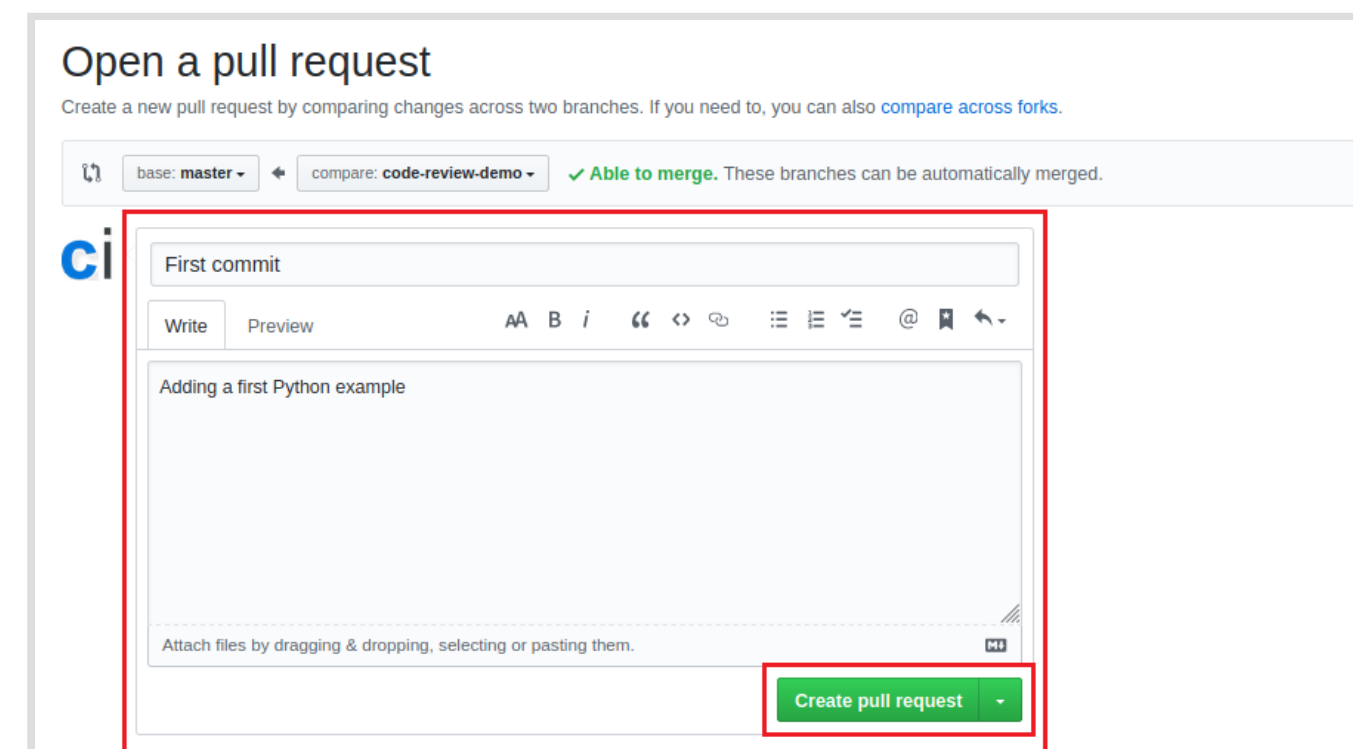
Then commit and push our changes to our Github repository.

```
$ git add main.py
$ git commit -a
$ git push --set-upstream origin code-review-demo

Counting objects: 3, done.
Delta compression using up to 8 threads.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 349 bytes | 349.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
remote:
remote: Create a pull request for 'code-review-demo' on GitHub by visiting:
remote:      https://github.com/codeinspectordemo/demo/pull/new/code-review-
demo
remote:
To https://github.com/codeinspectordemo/demo.git
 * [new branch]      code-review-demo -> code-review-demo
Branch 'code-review-demo' set up to track remote branch 'code-review-demo' from
'origin'.
```

We pushed the branch to the remote repository on GitHub. Now, we need to create a pull request that will formally ask to push the branch on the master. The URL to create the pull request is provided when we pushed the branch and we just need to visit it: <https://github.com/codeinspectordemo/demo/pull/new/code-review-demo>

When you open the link, you need to put a title and message for the Pull Request. Click on “Create pull request” below to create it.



Open a pull request

Create a new pull request by comparing changes across two branches. If you need to, you can also [compare across forks](#).

base: **master** compare: **code-review-demo** ✓ Able to merge. These branches can be automatically merged.

ci

First commit

Write Preview AA B i “ <> ☰ ☷ ☹ @ 📎 ↶

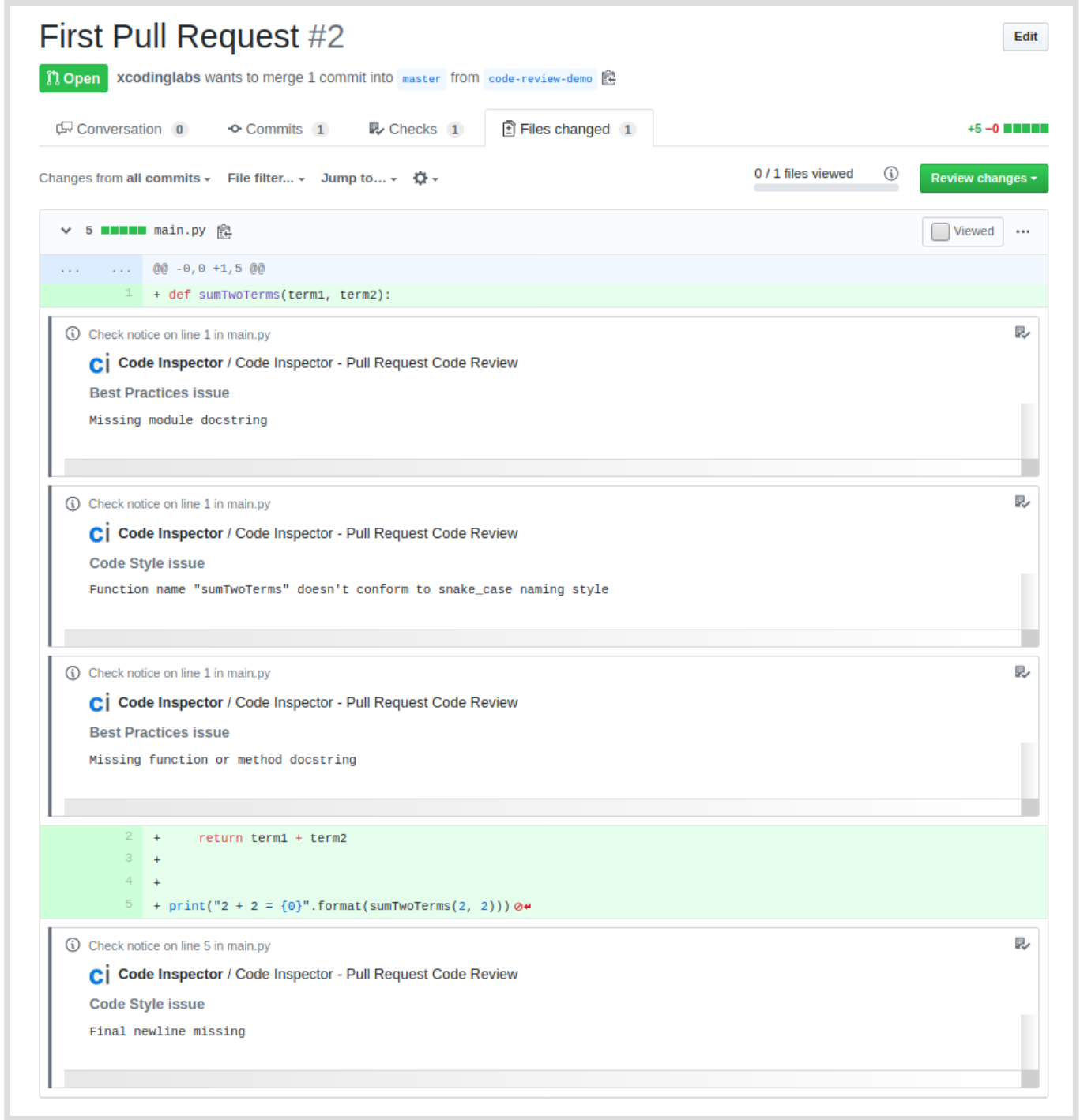
Adding a first Python example

Attach files by dragging & dropping, selecting or pasting them.

Create pull request

Step 3: Check the results in the Pull Request

The pull request will then be analyzed. Once the analysis is finished, you will see the summary of the analysis in the pull request. To see the result for each analyzed file, click on the File tab as shown below.



Step 4: Fixing and Validating Code does not have an issue

We can fix and address the issue reported in the automated review.

In the present case, according to the review, we need to:

- add documentation for the module
- Add documentation for the function
- make sure the function uses the snake_case rule naming
- add a final newline at the end of the file

In the present case, to fix the issues reported by the Code Inspector, we added documentation for the module to make sure the function uses the snake_case rule. We also added a final newline after the print statement.

This is how the correct code looks.

```
""" My first Python module """
```

```
:param term2: the second term
:return:
"""
    return term1 + term2

print("2 + 2 = {}".format(sum_two_terms(2, 2)))
```

Once you modified the code, update it on the remote repository.



```
$ git commit -a
$ git push





Username for 'https://github.com': <username>
Password for 'https://<username>@github.com':




Counting objects: 3, done.
Delta compression using up to 8 threads.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 429 bytes | 429.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/codeinspectordemo/demo.git
  79594a2..c0bbd8a  code-review-demo -> code-review-demo
```

The pull request status will be automatically be updated and we have the guarantee that the updated code has been verified and is correct. Looking at the history of commits, we can see that the first commit did not pass the automated code review while the updated code passes all verification.




First Pull Request #2

 **Open** xcodinglabs wants to merge 2 commits into `master` from `code-review-demo` 



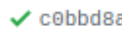
 Conversation **0**  Commits **2**  Checks **0**  Files changed **1**


 **xcodinglabs** commented 15 minutes ago  


Adding a first Python example


  First commit 


+ New changes since you last viewed [View changes](#)







  fix issues 




 **All checks have passed**
3 successful checks [Show all checks](#)

 **This branch has no conflicts with the base branch**
Only those with [write access](#) to this repository can merge pull requests.



Write **Preview** **AA** **B** *i* “ ” < >     @  

Leave a comment

Attach files by dragging & dropping, selecting or pasting them. 

Wrapping Up

In this tutorial, we explained how to automate code reviews on GitHub with Code Inspector. While the example we took in this tutorial is basic, code Inspector supports more than ten languages and can be used on multiple platforms, including GitHub, Gitlab or Bitbucket. The Code Inspector engine includes rules for code duplicates, complexity or even readability. You can also integrate our analysis engine in your Continuous Integration pipeline in order to block merge or code that does not meet a given quality standard.

Links

- Code Inspector: <https://www.code-inspector.com>
- Example of code automated review: <https://github.com/codeinspectordemo/demo/pull/2>

7 SHARES:

 SHARE 7

 TWEET







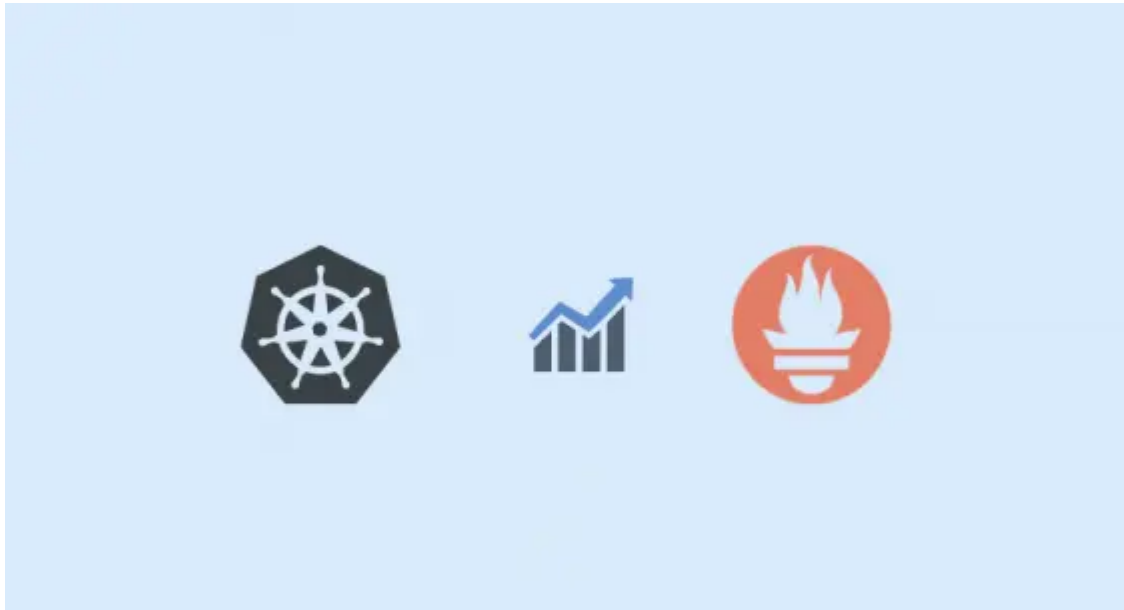
Julien Delange

Julien Delange is a software engineer living in San Francisco. He has experience of building large-scale software in different domains (cloud, social media, avionics or aerospace). Julien is the founder of Code Inspector, a platform that helps developers and managers to produce better software. Personal website: julien.gunnm.org



VIEW COMMENTS (0) ▾

YOU MAY ALSO LIKE



D — DEVOPS

How to Setup Prometheus Node Exporter on Kubernetes

by **devopscube** · April 6, 2021

If you want to know how the Kubernetes nodes perform or monitor system-level insights of kubernetes nodes, you...

How To Setup and Configure a Proxy Server – Squid Proxy

by **devopscube** · August 11, 2018

A proxy server has many use cases. it could range from personal internet access to restrict organization systems/servers...

Setup Jenkins master and Build Slaves as Docker Container

by **Mi Jia** · October 20, 2017

Do you want dockerized Jenkins which includes the configuration for build slaves also as Docker containers? So that you can...

Vagrant Tutorial For Beginners: Getting Started Guide

by **Bibin Wilson** · April 18, 2021

In this Vagrant tutorial, I will teach you to set up Vagrant on your workstation to create and...

How To Backup Jenkins Data and Configurations

by **Bibin Wilson** · June 14, 2021

It is very important to have Jenkins backup with its data and configurations. It includes job configs, builds...

How To Setup an Elasticsearch Cluster – Beginners Guide

by **devopscube** · February 4, 2016

In part I, we learned the basic concepts of elasticsearch. In this tutorial, we will learn how to...

DevopsCube

©devopscube 2021. All rights reserved.

[Privacy Policy](#)

[About](#)

[Site Map](#)

[Disclaimer](#)

[Contribute](#)

[Advertise](#)

[Archives](#)

