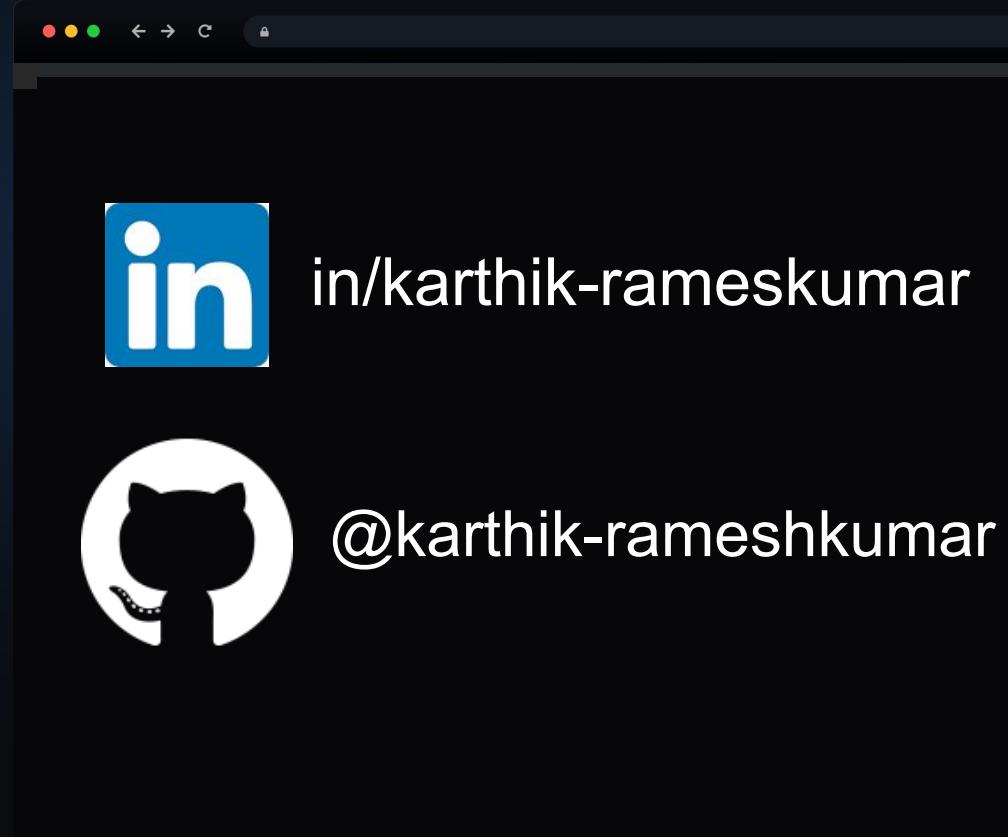


Supercharging your DevOps Toolchain in the age of Generative AI



Karthik Rameshkumar

Director, Field Services - APAC West



GitHub is the largest developer platform on Earth

100M+
Developers

200M+
Private and public
repositories

1,000s
Top open source
communities

2.6B+
Contributions
per year

4M+
Organizations

90%
Fortune 100
companies

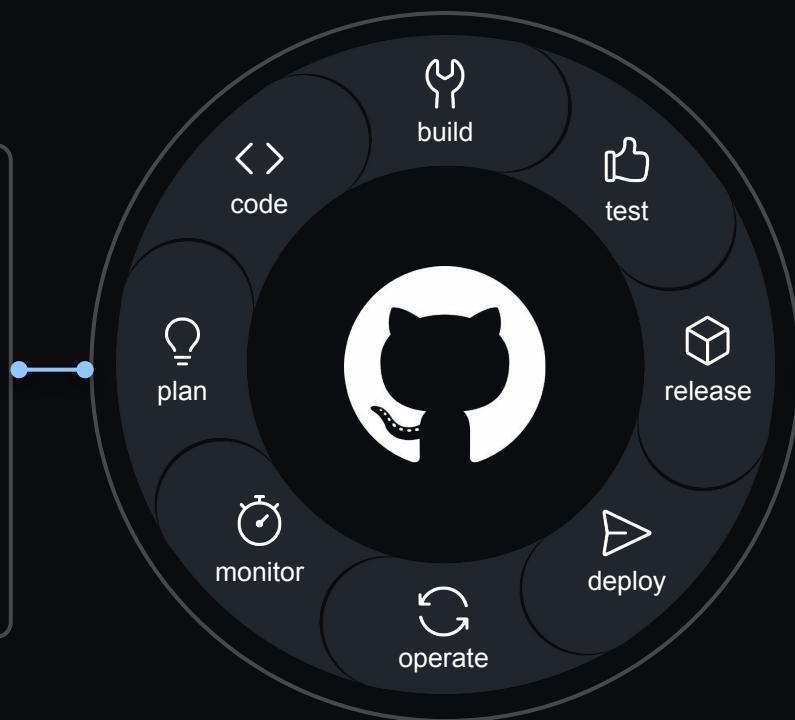


**A fully integrated platform
from idea-to-production**

**Collaborative, automated
workflows**

**Seamless access to OSS
and Innersource**

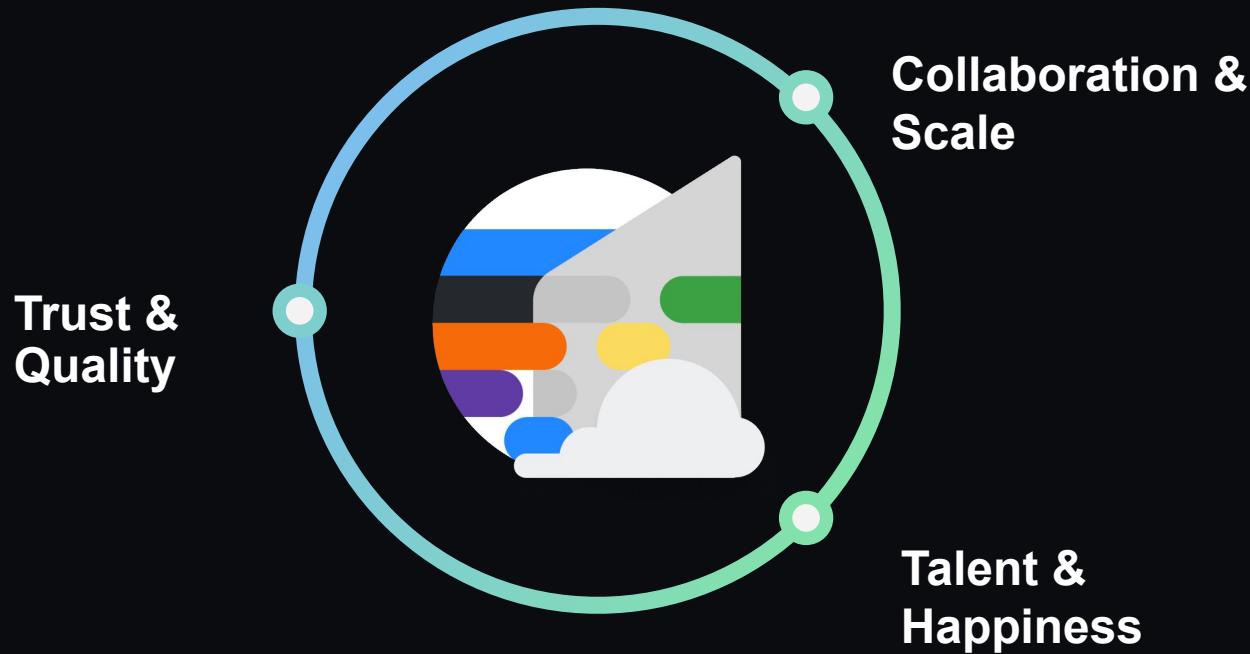
**Developer focused security
and compliance**



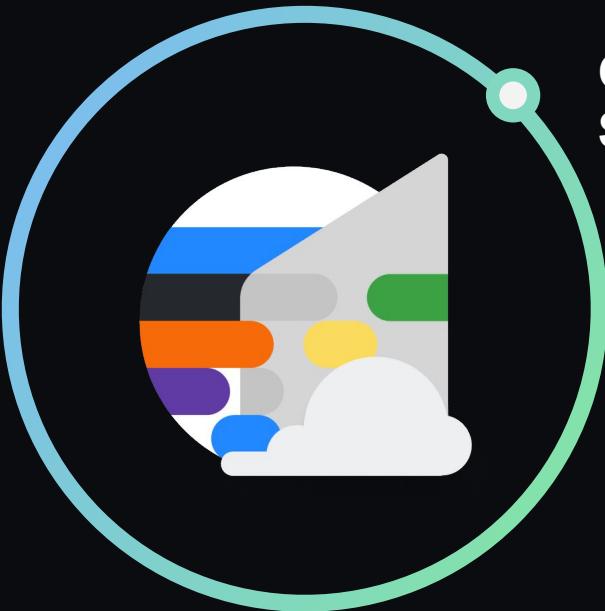
**Deploy anywhere including
your own data centers**



Developer Velocity



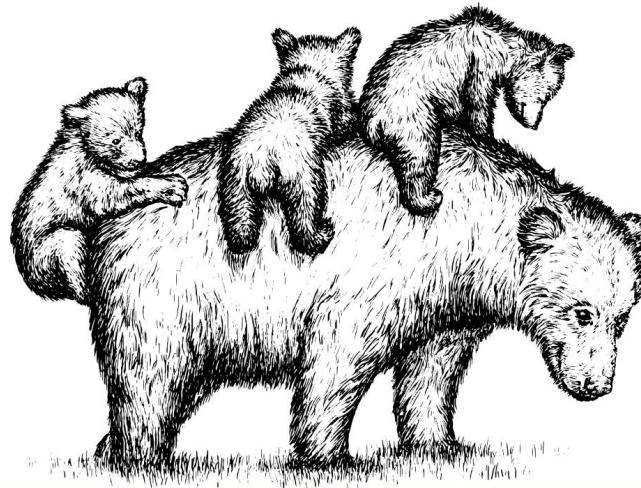
Developer Velocity



Collaboration &
Scale



Getting the wrong idea from that conference talk you attended



Solving Imaginary Scaling Issues

At Scale

O RLY?

@ThePracticalDev



Modern, Collaborative DevOps

100x

faster to set up a dev environment

970x

more frequent code deployments

7x

fewer failures on deployments

2x

more likely to have internal communities of practice

8x

more likely to have integrated security

4-5x

Higher revenue growth

Source: McKinsey



Key Drivers: Standardized DevOps Tools, OSS & InnerSource, Cloud Migration

Tools are top driver of Developer Velocity; architecture and test automation are key enablers

Open source adoption was a differentiator for mature companies

Cloud is key differentiator in Retail & Banking

Companies look to standardize toolchain in a few key areas: typically, CI/CD pipeline, planning and code management tools and leave flexibility in other areas to individual teams and developers

OSS is a differentiator between the top performers vs. the rest – particularly for attracting and retaining top talent and increasing speed of innovation

Inner-source adoption is early across the board but top of the agenda for more mature companies

Successful migration to cloud requires a top down executive focus (jointly across business and technology, typically CEO agenda) – **with focus on speed and agility vs. IT efficiencies.**

Source: McKinsey study on Developer Velocity in Retail & Banking



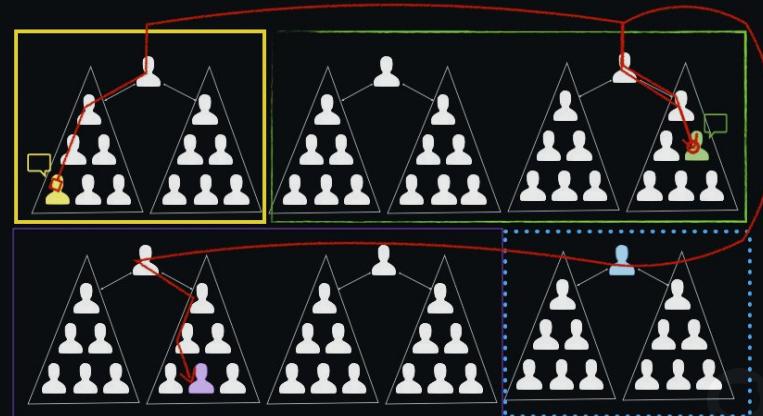
Broadening focus from Team to Org

Software journey since mid-2010s focused on few priority areas:

- Consolidation of platforms / tech stacks
- Tooling and engineering practices standardization
- Agile team practices



Areas of further improvement in the next 2 years include – improving organizational agility, tooling, agile practices at organizational level, and usage of hybrid / public cloud

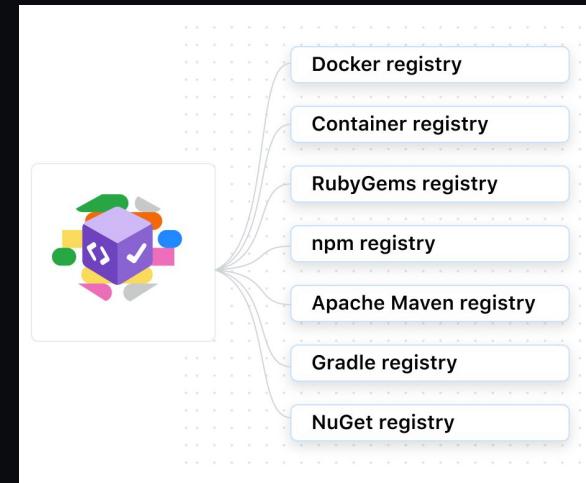


Source: large US-based banks (60M+ accounts, \$85B+ revenue)



Collaboration Best Practices

- Train devs to look for existing code before writing something new, e.g. searching open/innersource in GitHub or via a dev portal
- Innersource goes beyond raw code sharing
 - Use private packages to reuse code as versioned dependencies
 - Config-as-code = collaborate on env setup, workflows, & security
- Simplify development and improve replicability with containers
- Automate, automate, automate!



Workflows made for your JavaScript repository Suggested

Publish Node.js Package
By ml-team
Automatically publish to GitHub Packages with each release.
[Set up this workflow](#)

Node.js
By ml-team
Builds node projects with access to the QA database and larger runner machines.
[Set up this workflow](#)



**TO ALL THOSE AWESOME SCRUM
MASTERS OUT THERE**

WE SALUTE YOU





exclusive

REGIONAL
SCRUM
GATHERINGSM

4-5 AUGUST 2023 | CHENNAI



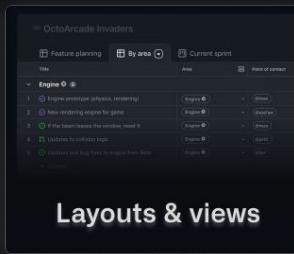
Project Management

Projects, Issues, Discussions, Wiki



Plan and Track close to your code

Create issues, break them into tasks, track relationships, add custom fields, and have conversations. Visualize large projects as spreadsheets or boards, and automate everything with code.



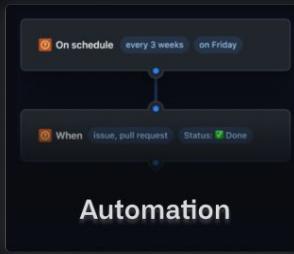
Layouts & views



Custom Fields



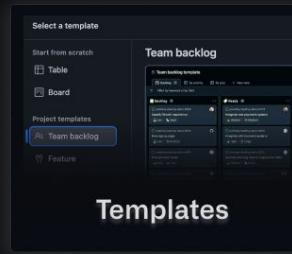
Relationships



Automation



Charts & Insights



Templates



GitHub Issues & Projects

Project management made simple



Start with an issue

Create an issue to suggest a new idea or track a bug. Then organize and assign tasks to your team.



Link issues and pull requests

Track updates to your project without complicating your workflow by crosslinking between issues and pull requests anywhere in your GitHub organization.



See your work take shape with Projects

Visualize all of your work and prioritize it right alongside your code with projects boards. See what tasks are planned or in-progress, either in a repository or across your organization.

The screenshot shows the GitHub Projects interface for the repository "SmartTractor". The main view displays two columns: "Backlog" (13 items) and "In Progress" (6 items). The "Backlog" column includes tasks like "Collect satellite data and deliver to farmers" and "Launch Plan". The "In Progress" column includes tasks like "Crawl tractor engine data (John Deere)" and "Performance updates for data script". A third column, "Ready to deploy", contains a single item: "[Data] Soil data collection scripts". A tooltip for the task "Share farms' soil moisture data" in the backlog is open, showing it was added by Eddie and has a due date of Nov 10, 2022.



Developer Velocity

Trust &
Quality



A decade ago, NASA was aiming for the red planet

- Mars Curiosity was launched after years of development
- Myriad of security tools, processes, and reviews

Somewhere between Earth and Mars, disaster threatens

- NASA JPL Continues code review while mission is in flight
- Crashable bug discovered via manual review
- No clear way to identify how to detect this bug with traditional approaches



**Stakes were
extremely high...**

\$2.5B

Total estimated project cost

"It could take the entire Mars program down with it. It is victory or death."

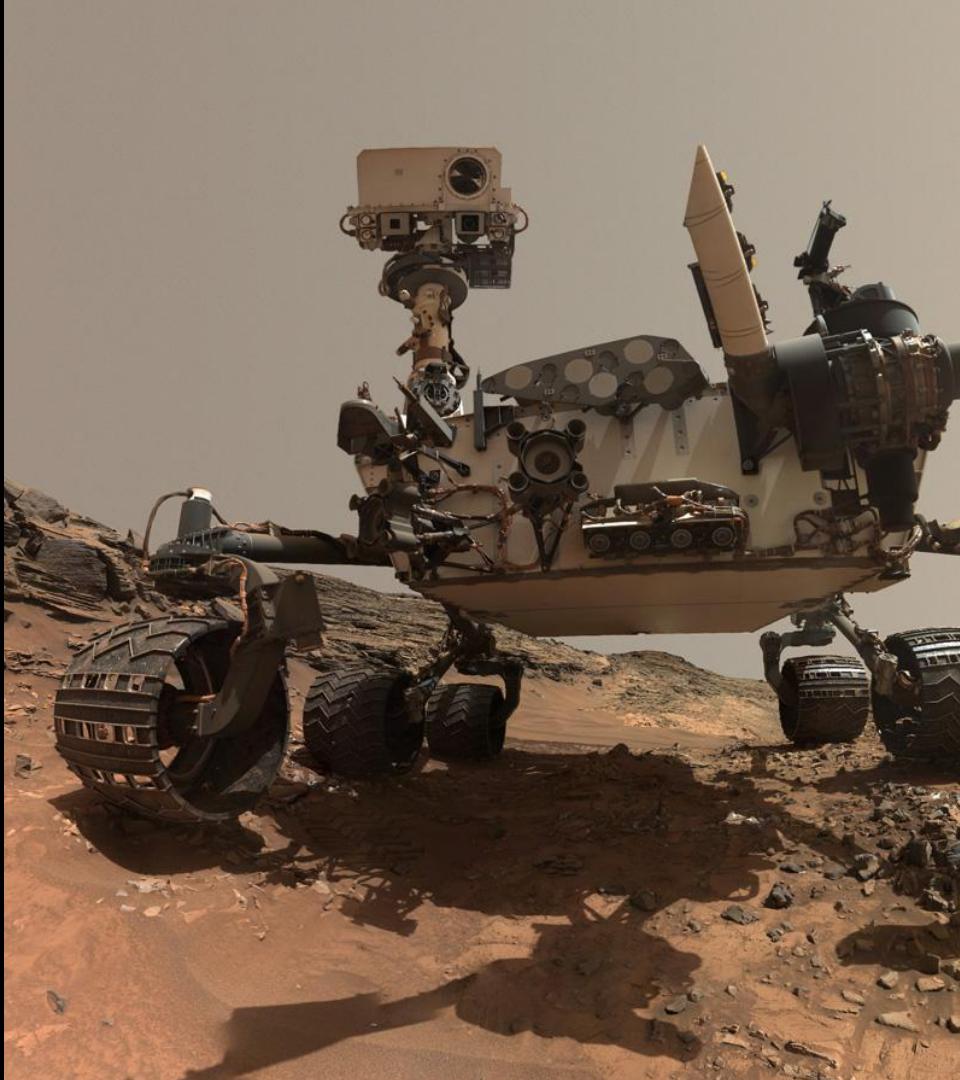
Robert Zubrin



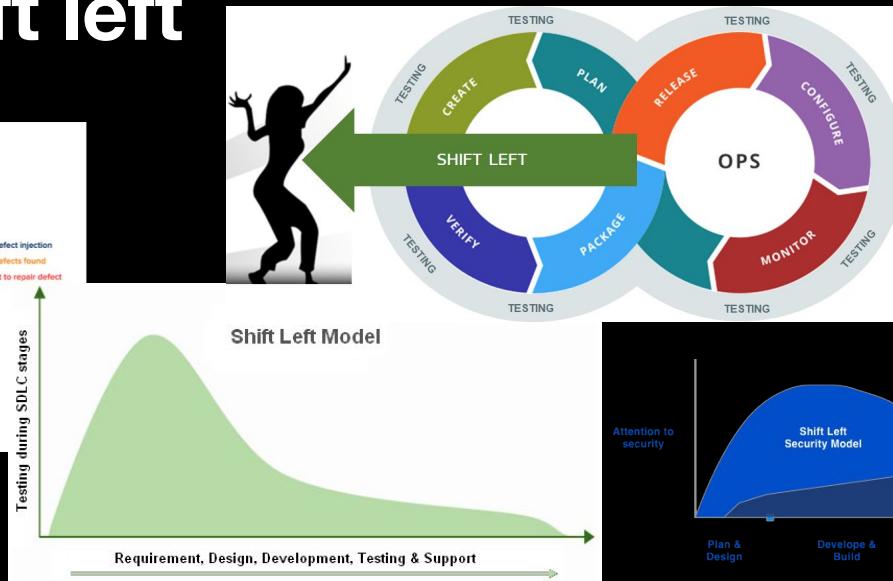
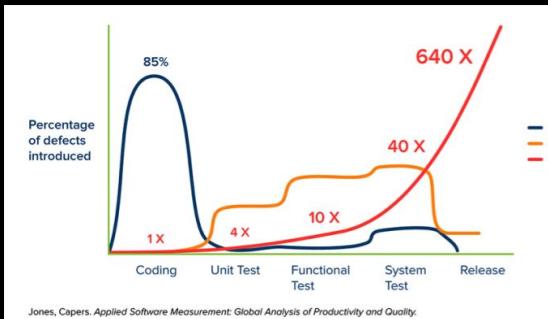
CodeQL to the rescue!



30 bug variants found & fixed



Let's just shift left



There are many interpretations of “shift-left”, and there are no right or wrong ways of implementing a “shift-left” approach. The fundamental principles always stay the same, though. It’s about finding bugs/issues/problems early in your software development life cycle.

On behalf of most developers out there ...

This is how it feels



Taking a step back now

*It's **so** easy to consider the apparent upfront value of "shifting-left", the changed downstream end-user experiences are often overlooked and overshadowed.*

Problems:

- Such high context switching
- Lack of focused productivity
- Not delivering as quickly as the business requires
- Not a feel of empowerment or developers voices being considered

It's the unapparent unknown consequences that generally have the most significant impact.



snyk

[Product](#) [Pricing](#) [Docs](#) [Learn](#) [Company](#)
[Log in](#) [BOOK A DEMO](#) [SIGN UP](#)

Developer-first Cloud Native Application Security

Loved by both developers and security teams

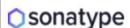


ON-DEMAND WEBINARS

Application Security In 2020: Keep Calm And Shift Left

For years, businesses have developed applications to meet customer, partner, and vendor needs, and for years, attackers have taken advantage of application flaws to compromise organizations and put their customers at risk.

[REGISTER NOW >](#)


[Products](#) [Solutions](#) [Success Stories](#) [Resources](#)


Getting Your Security Program to Shift Left: Operationalizing Security Controls via DevSecOps

May 21, 2020 By Daniel Longest

We believe in empowering developers to own Code Security. Project Security starts in the code; SonarQube helps you own it

[Request a Free Trial](#)



[Community Home](#) > [Security](#) > [Security Blog](#) > The Basics of Shift Left AppSec

The Basics of Shift Left AppSec



Harley_Adams

Micro Focus Expert



6 Likes

2020-08-20 13:00

Software developers know that the phrase "Shift Left" means testing (and preventing) defects earlier in the software development process. It's a phrase that's so ingrained, that it's even trendy to say "Shift Left isn't enough". (Basically, the point of that is that testing needs to be automated, too.)

Rob Aragao, CISSP, is a Chief Security Strategist at Micro Focus. He recently shared his view of shift left in this great video on our [Fortify Unplugged](#) YouTube channel:



Tag: Shift Left



KICS - How We Made It

Feb 25, 2021 by Nuno Oliveira

Cloud Native implies a complete change in philosophy on how modern applications are designed, developed, and deployed. Finally, monoliths are broken into small and self-contained microservices, independent and unaware of their context. Orchestration glues them together in applications that become more scalable, reliable, and utterly resilient. Orchestration, in this context, deals not only with the

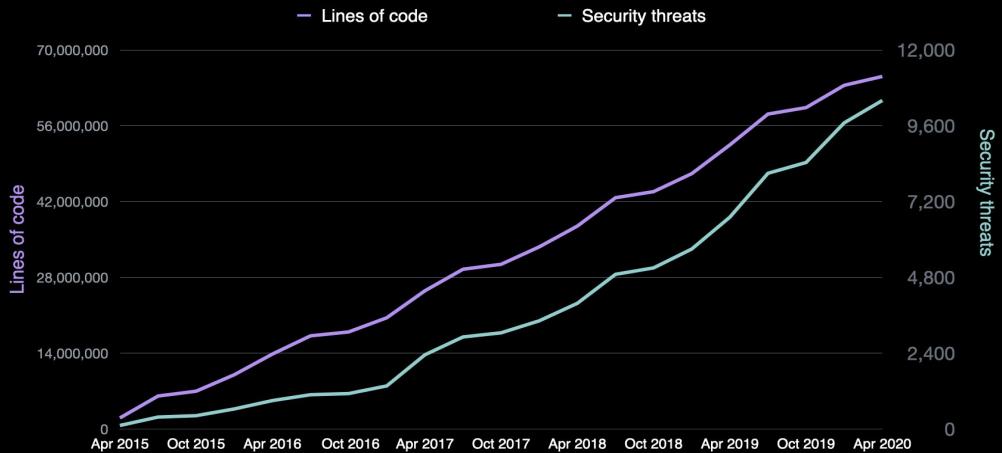
[Read More >](#)

Schedule a Demo Contact Us Blog Community Login

Products Solutions Developers Partners Resources About Us

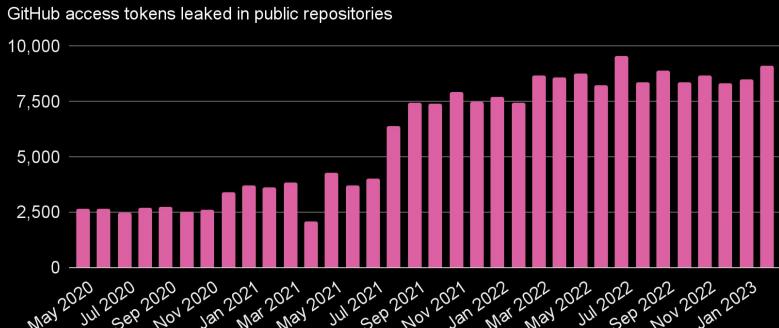
[Home](#) | [Blog](#) | [Intro to AppSec](#)

4 First Steps to Help Your Organization Shift AppSec Left



It hasn't quite worked out, has it?

Source: GitHub Data Science Team - Analysis of 70 million lines of code in major OSS projects added over a 5 year period



Source: GitHub Data of PAT's leaked in public repositories

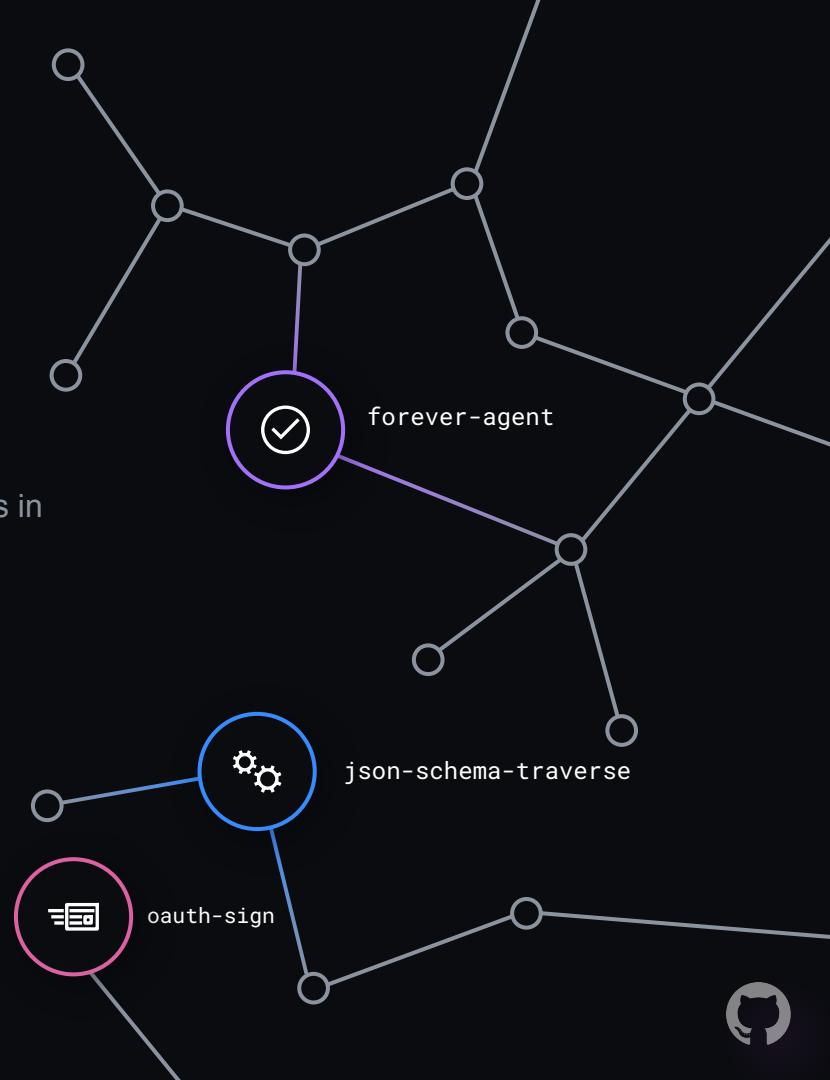
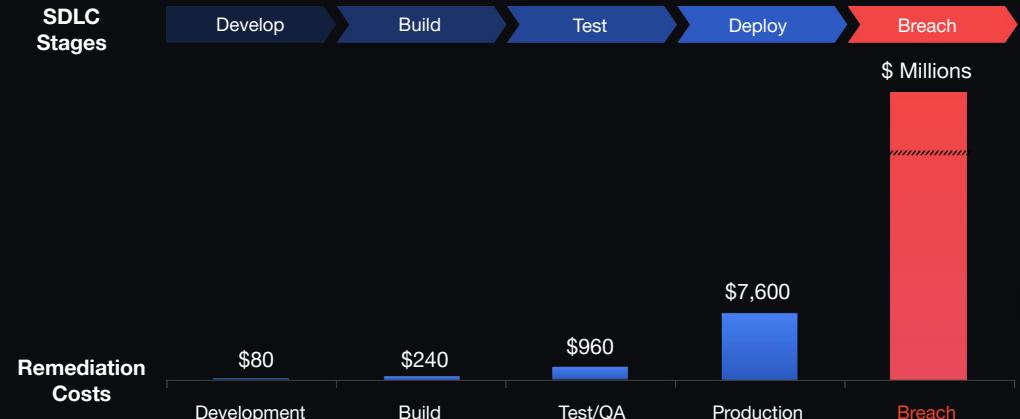
Modern software development is built on open source

80-90%

open source in new applications

100x

More expensive to remediate vulnerabilities in production than builds

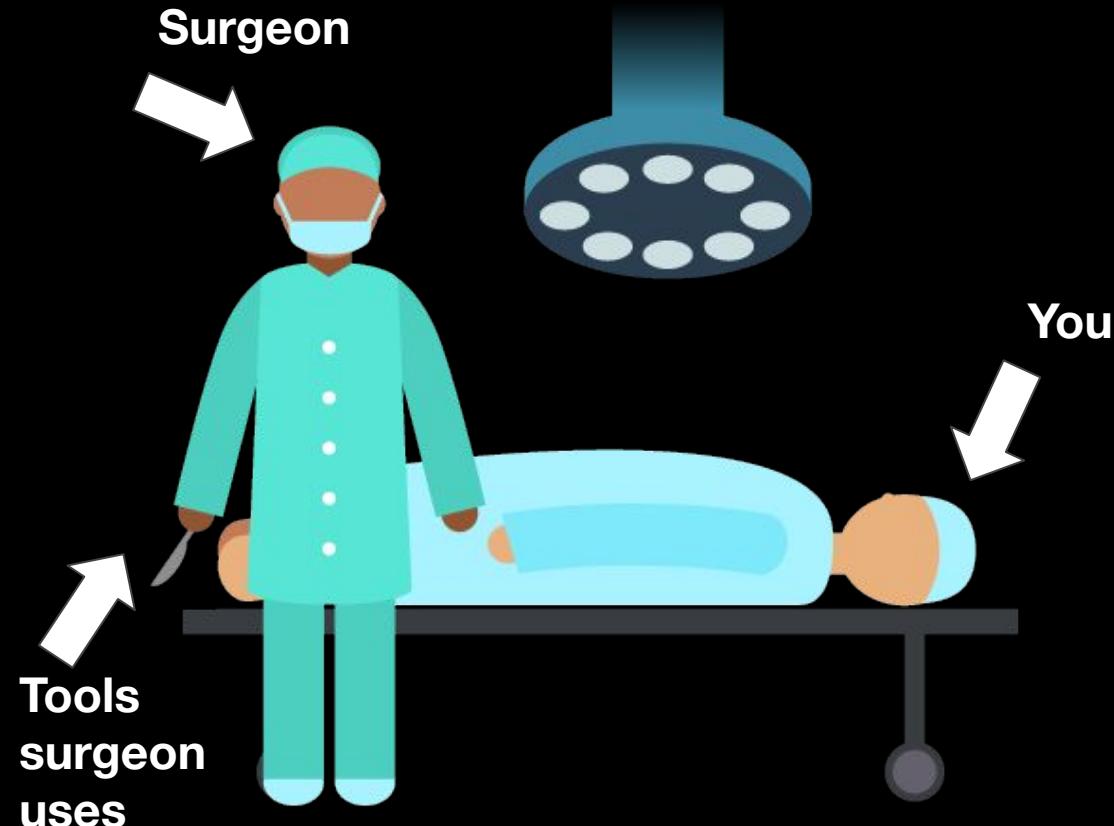


Getting Real about this for a second ...

Question:

If you went for surgery tomorrow, would you be happy if the surgeon used tools that weren't meant for a surgeon? Think about the consequences? Slow surgery, lousy success rate, etc.

You wouldn't! So why in the developer world are we happy giving tools to developers that weren't initially meant for them.



20,000+

public repos have enabled automated code review using CodeQL
(before it's native!)

2,000+

open source queries with contributions from Microsoft, Google and many others

27%

of recent JS CVEs would have been identified by a default CodeQL query

72%

fix rate for potential vulnerabilities flagged in open source projects



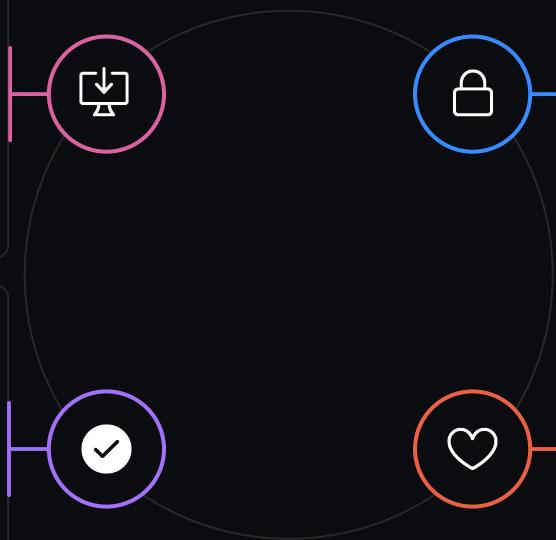
...and many more

The GitHub Difference

Developer empowered security to fix in minutes not months

Native

- Embedded from init to ship
- Automatically deployed
- Detailed changelog
- Visibility into security posture across all code



Trust-By-Design

- Static code analysis
- Supply chain analysis
- Secret scanning
- Security Lab
- Extensible platform

Fix in minutes

- In-context feedback
- Automated fixes
- Jargon free language
- Blameless culture

Community Driven

- Supported by millions of developers and security researchers
- Contributions from leading peers

@author

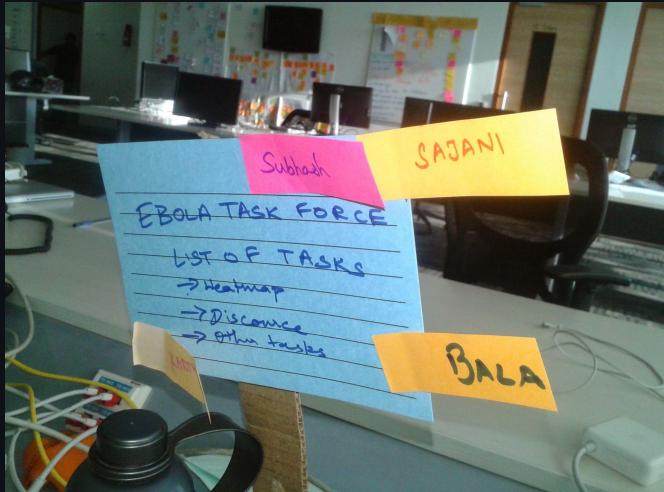
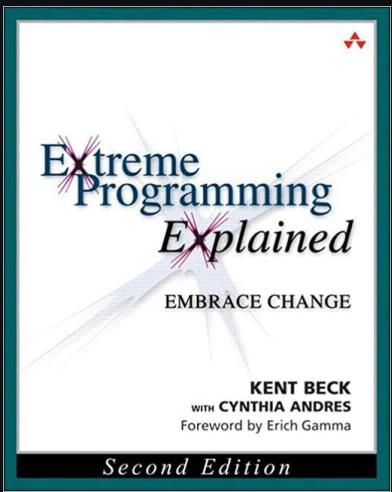
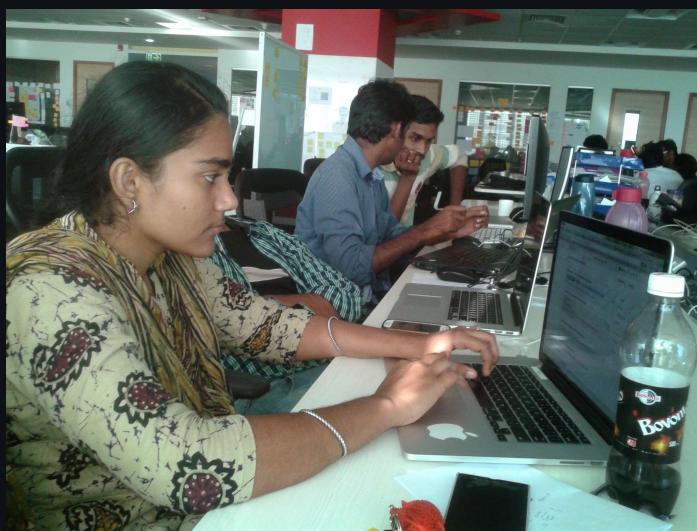


Developer Velocity



Talent &
Happiness





It would be a pure function if not for the side effects on your sanity



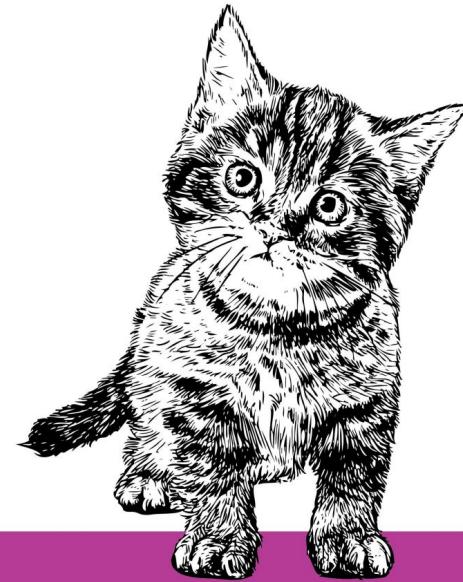
Turning Coffee Into Code

The Definitive Guide

O RLY?

@ThePracticalDev

How to actually learn any new programming concept



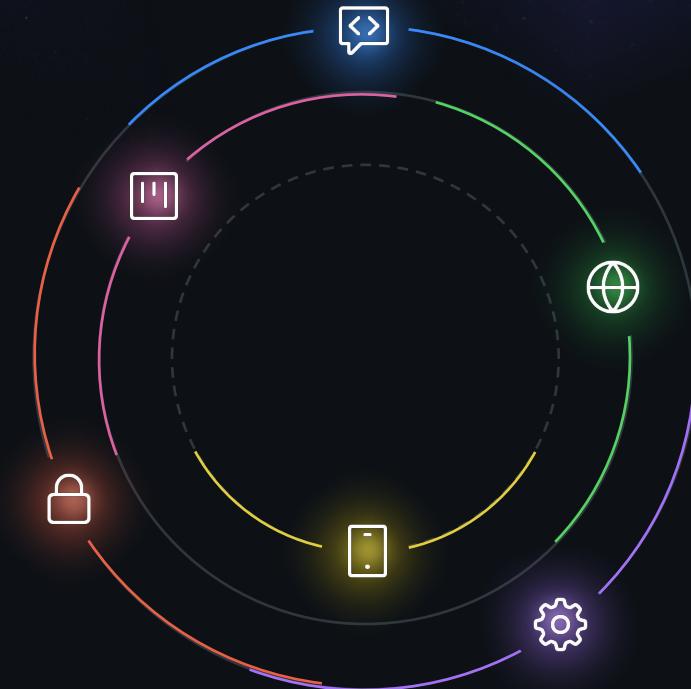
Essential

Changing Stuff and
Seeing What Happens

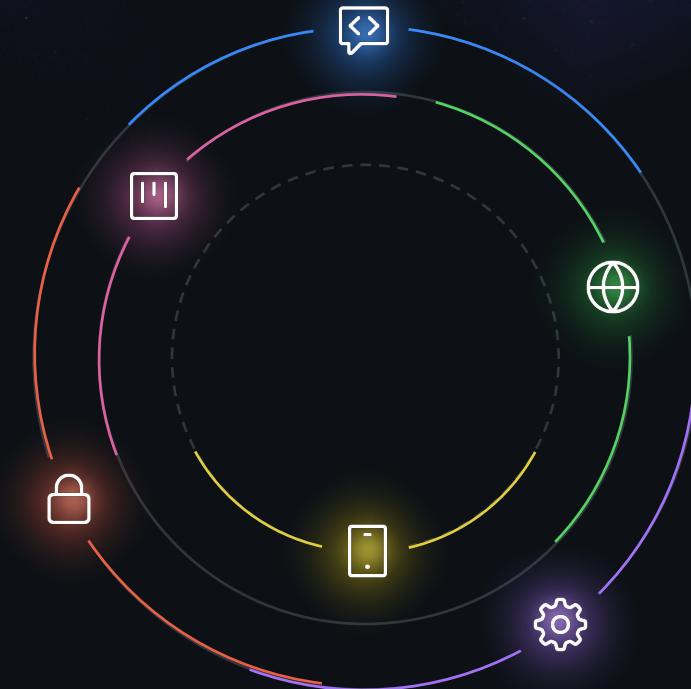
O RLY?

@ThePracticalDev

But it is no longer 2015!



The age of generative AI is here



Generative AI |

through the lens of software development



GITHUB COPILOT



55%

⌚ Faster Coding



75%

❤️ More fulfilled



46%

<> Code written



GitHub Copilot

 We recruited **95** developers , and split them randomly into two groups.

We gave them a task of writing a web server in JavaScript



45
Used Copilot



78%
Finished



1hr 11min
Average to complete the task



50
Did not use Copilot



70%
Finished



2hrs 41min
Average to complete the task

Prevention

GitHub Copilot enables developers to reach that **magical flow state**—including delivering fast, accurate vulnerability prevention right in the editor. This, combined with GitHub Advanced Security provides an end-to-end **seamless experience** for developers to secure their code.



AI-based vulnerability prevention
Blocks insecure coding patterns in real-time to make GitHub Copilot suggestions more secure.



Secure suggestions
Insecure coding patterns are quickly blocked and replaced by alternative suggestions.



Incredibly fast
GitHub Copilot runs advanced AI models on powerful compute resources



Copilot Chat

Context-aware ChatGPT-styled conversations with Copilot

The image shows a split-screen interface. On the left is the GitHub Copilot Chat window, and on the right is a code editor window.

GITHUB COPILOT: CHAT

monalisa

Write a set of unit test functions for the selected code

GitHub Copilot

```
import unittest
import datetime

class TestParseExpenses(unittest.TestCase):
    def test_parse_expenses_with_valid_input(self):
        expenses_string = '2023-01-02 -34.01 USD\n2023-01-03 2.59 DKK'
        expected_output = [(datetime.datetime(2023, 1, 2),
                            float(-34.01), "USD"),
                           (datetime.datetime(2023, 1, 3),
                            float(2.59), "DKK")]
        self.assertEqual(parse_expenses(expenses_string), expected_output)

    def test_parse_expenses_with_empty_input(self):
        expenses_string = ''
        expected_output = []
        self.assertEqual(parse_expenses(expenses_string), expected_output)

    def test_parse_expenses_with_comments(self):
        expenses_string = '# This is a comment\n2023-01-02 -34.01 USD\n2023-01-03 2.59 DKK'
        expected_output = [(datetime.datetime(2023, 1, 2),
                            float(-34.01), "USD"),
                           (datetime.datetime(2023, 1, 3),
                            float(2.59), "DKK")]
        self.assertEqual(parse_expenses(expenses_string), expected_output)

if __name__ == '__main__':
    unittest.main()
```

The code assumes that the `datetime` module is imported. The test cases cover different scenarios such as valid input, empty input, input with comments, invalid date format, and invalid value format.

Analyze and debug

Ask a question or type '/' for topics

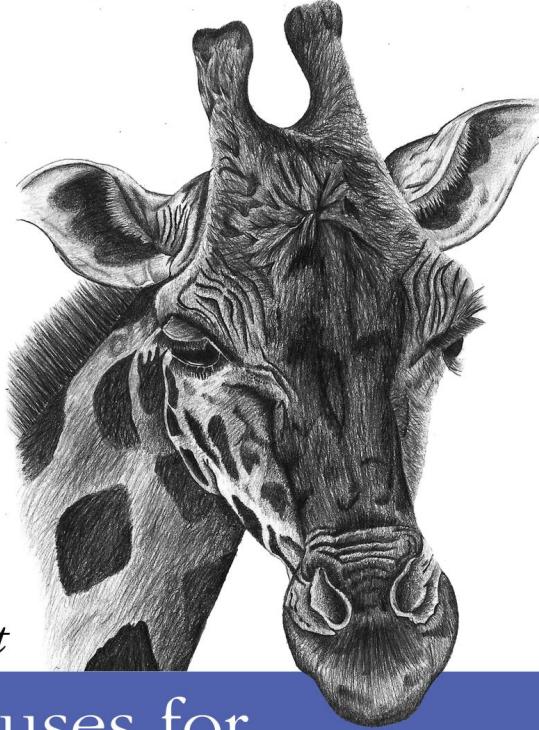
module.py

```
1 import datetime
2
3 def parse_expenses(expenses_string):
4     """Parse the list of expenses and return the list of triples (date, amount, currency).
5     Ignore lines starting with #.
6     Parse the date using datetime.
7     Example expenses_string:
8         2023-01-02 -34.01 USD
9         2023-01-03 2.59 DKK
10        2023-01-03 -2.72 EUR
11
12     expenses = []
13
14     for line in expenses_string.splitlines():
15         if line.startswith("#"):
16             continue
17         date, value, currency = line.split(" ")
18         expenses.append((datetime.datetime.strptime(date, "%Y-%m-%d"),
19                          float(value),
20                          currency))
21
22     return expenses
23
24 expenses_data = '''2023-01-02 -34.01 USD
25 2023-01-03 2.59 DKK
26 2023-01-03 -2.72 EUR'''
```

Ln 17, Col 3 Spaces: 2 UTF-8 LF ⓘ TypeScript



Your application is a special snowflake



Expert

Excuses for Not Writing Unit Tests

O RLY?

@ThePracticalDev



value, value, currency) but it is currently (value, currency, date).
3. The format string used in strftime() is incorrect. It should be "%Y-%m-%d" instead of "%d-%m-%Y".

Here's the corrected code:

```
def parse_expenses(expenses_string):
    """Parse the list of expenses and return
    list of triples (date, value, currency).
    Ignore lines starting with #.
    Parse the date using datetime.
    Example expenses_string:
        2023-01-02 -34.01 USD
        2023-01-03 2.59 DKK
        2023-01-03 -2.72 EUR
    .....
    expenses = []

    for line in expenses_string.splitlines():
        if line.startswith("#"):
            continue
        date, value, currency = line.split(",")
        expenses.append(datetime.datetime(
            float(value),
            currency))
```

↳ How can I use the parse_expenses function in my code?

Ask Copilot a question or type '/' for topics

github-copilot-examples > python > parse_expenses.py > parse_expenses

```
1 import datetime
2
3 def parse_expenses(expenses_string):
4     """Parse the list of expenses and return the
5     list of triples (date, value, currency).
6     Ignore lines starting with #.
7     Parse the date using datetime.
8     Example expenses_string:
9         2023-01-02 -34.01 USD
10        2023-01-03 2.59 DKK
```

OUTPUT TERMINAL DEBUG CONSOLE

```
● Apple ✘ ~/Dev/github.c/affrae/github-copilot-examples 🐍 main !3 > cd python
      ⚡ 08:21:07
● Apple ✘ ~/Dev/github.c/affrae/github-copilot-examples/python 🐍 main !3 > python3
    parse_expenses.py
    Traceback (most recent call last):
      File "/Users/dfigucio/Developer/github.com/affrae/github-copilot-examples/py
      thon/parse_expenses.py", line 28, in <module>
          expenses = parse_expenses (expenses_data)
      File "/Users/dfigucio/Developer/github.com/affrae/github-copilot-examples/py
      thon/parse_expenses.py", line 18, in parse_expenses
          date, value, currency = line.split(",")
    ValueError: not enough values to unpack (expected 3, got 1)
● Apple ✘ ~/Dev/github.c/affrae/github-copilot-examples/python 🐍 main !3 > python3
    parse_expenses.py
    2023-01-02 00:00:00 -34.01 USD
    2023-01-03 00:00:00 2.59 DKK
    2023-01-03 00:00:00 -2.72 EUR
● Apple ✘ ~/Dev/github.c/affrae/github-copilot-examples/python 🐍 main !4 >
```

+ ⌂ ⌄ ⌁ ⌂

zsh python

zsh

Copilot for Pull Requests

Revolutionizes your Pull Requests with Copilot

The screenshot shows a GitHub pull request interface. The title of the PR is "Add edges for type assignability #920". The status is "Open" and it says "monalisa wants to merge 2 commits into main from awesome_feature". A green button with a copy icon is visible.

A comment from "monalisa" is shown, edited by copilot one second ago. The comment text reads:

This pull request adds semantic analysis to the feature extraction plugin, which enhances the feature graph with information about types and assignability of expressions in Java code. It modifies the pom.ml file, the FeaturePlugin.

Code walkthrough

- Import new semantic analysis classes in FeaturePlugin.java ([link](#))
- Create and invoke TypeAnalysis, TypeScanner, and AssignabilityAnalysis on compilation unit in FeaturePlugin.java ([link](#))
- Write type subgraph and add attributes for new edge types in DotOutput.java ([link](#), [link](#))
- Add fields, methods, and constructor logic for type node associations in FeatureGraph.java ([link](#), [link](#), [link](#), [link](#), [link](#))

generated by copilot

8



git commit -m "changes"

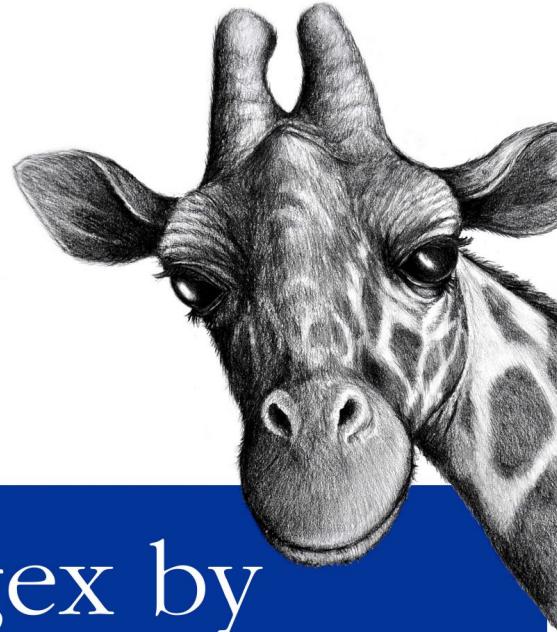


Writing

Useless Git Commit Messages



Combining slashes and dots until a thing happens



Expert

Regex by Trial and Error

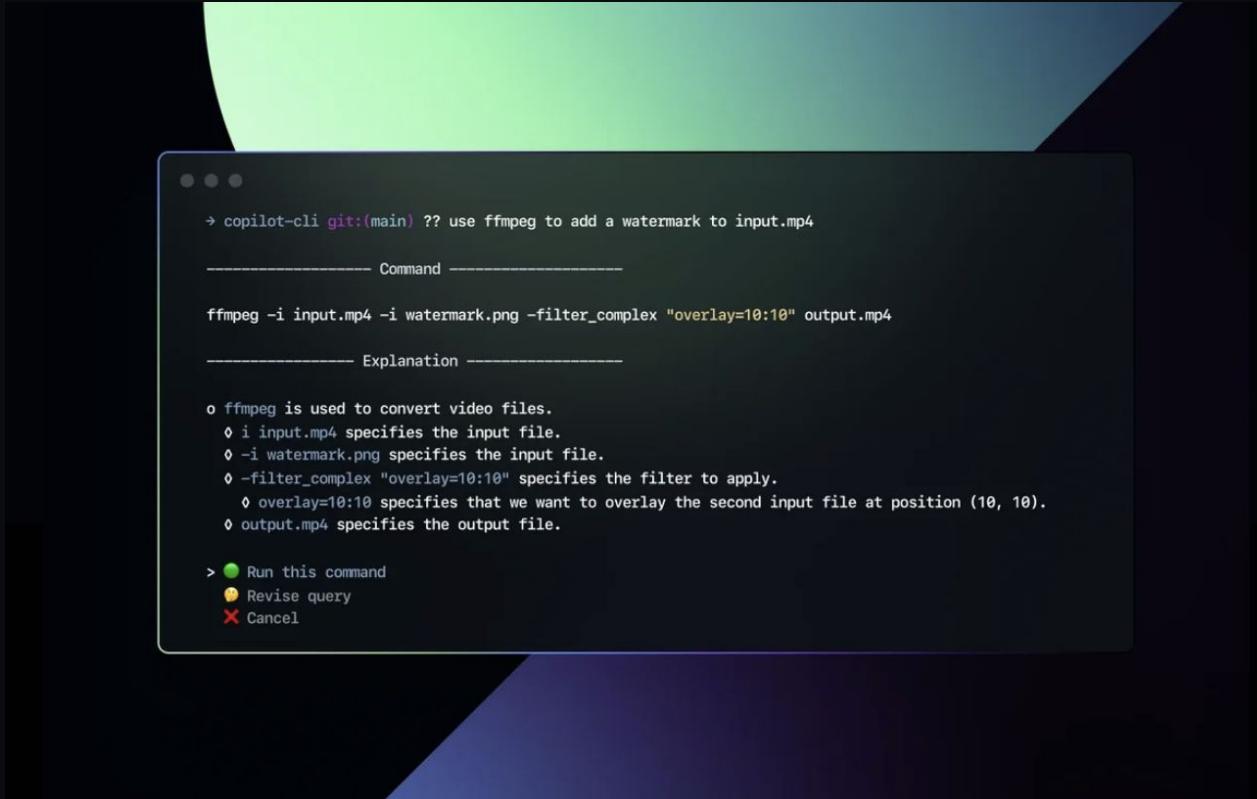
O RLY?

@ThePracticalDev

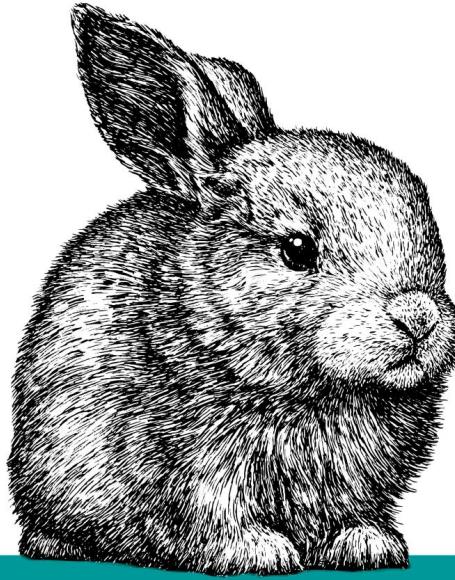


Copilot for your CLI

Ask GitHub Copilot for assistance right in your terminal



Feigning knowledge of a word you've heard a few times



Expert

Pretending to Know About Stuff

O RLY?

@ThePracticalDev



Copilot for Docs

Spend less time searching and more time learning

The screenshot shows a dark-themed messaging interface. At the top, there's a header with a star icon, a user profile picture, the text "MDN", and a message timestamp "1 message – started 1 minute ago". Below this, a message card has a small icon of a cat's head and the question "How do I vertically center a div?". The response text explains that there are different ways to vertically center a div, mentioning flexbox as a common method. It provides a code snippet demonstrating how to use flexbox for this purpose:

```
/* Parent element */
.container {
  display: flex;
  align-items: center;
  height: 300px; /* optional, but you need some height for the centering to work */
}

/* Child element */
.div {
  width: 200px;
  height: 100px;
  background-color: lightblue;
}
```

At the bottom of the message, there's a note: "You can learn more about flexbox and its properties on MDN . Another method is to use absolute positioning and the transform property, which allows you to move and rotate elements relative to their position. To use this method, you need to set the position property of the parent element to relative, and" followed by a redacted portion of the text.



SPRINT GOAL MET



**NOW WHERE'S MY
TREAT?**





The World is Waiting...

Let's Build from here!

