Carnegie Mellon University

Collaborative Development: Documentation & Testing

17-313, Foundations of Software Engineering, Fall 2022

S3D
Software and Societal Systems Department

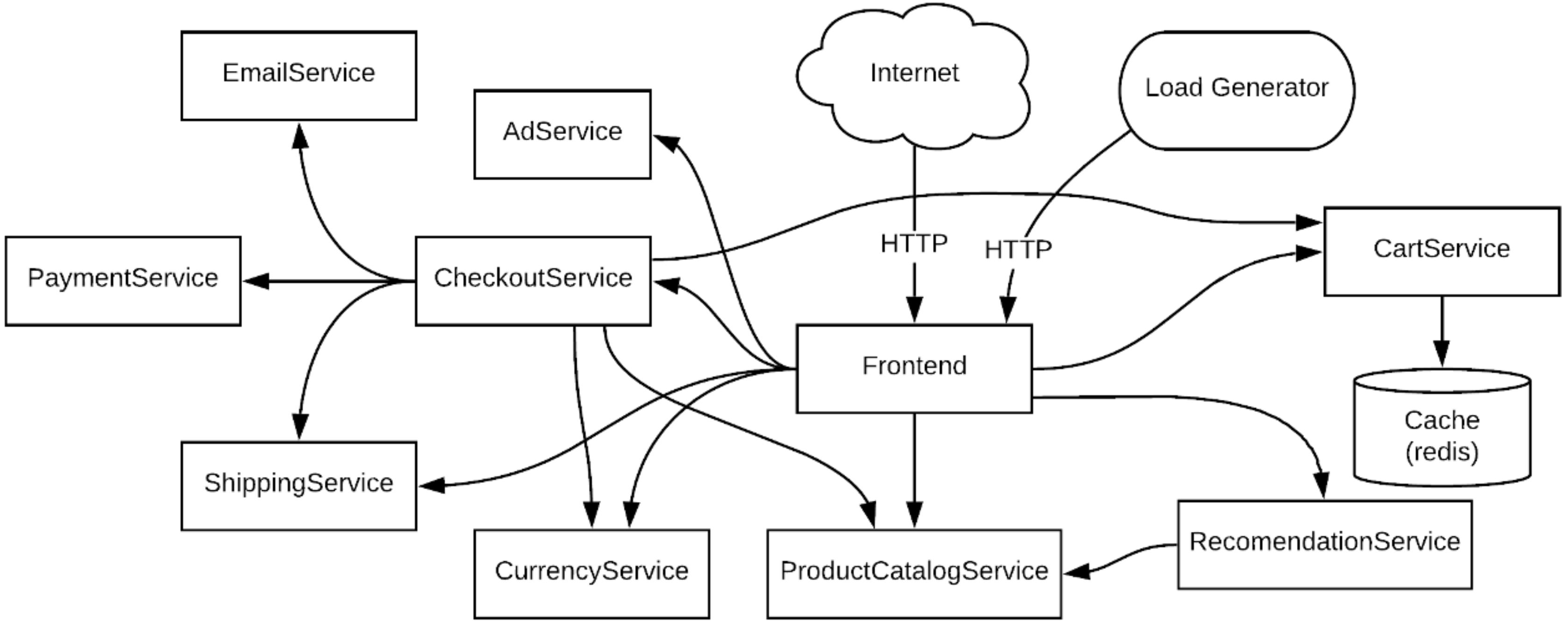
Administrivia

- Homework 3B due tonight (October 6th)
  - Homework 3C (Reflection) due October 13th
- Midterm next Tuesday, October 11th (in class, regular timing)
  - review session during recitation this week (come prepared)
  - any questions on the previous midterm questions – bring them to recitation to discuss as a class
  - cheatsheet: you can bring a single page of notes to the exam
- Teamwork Survey

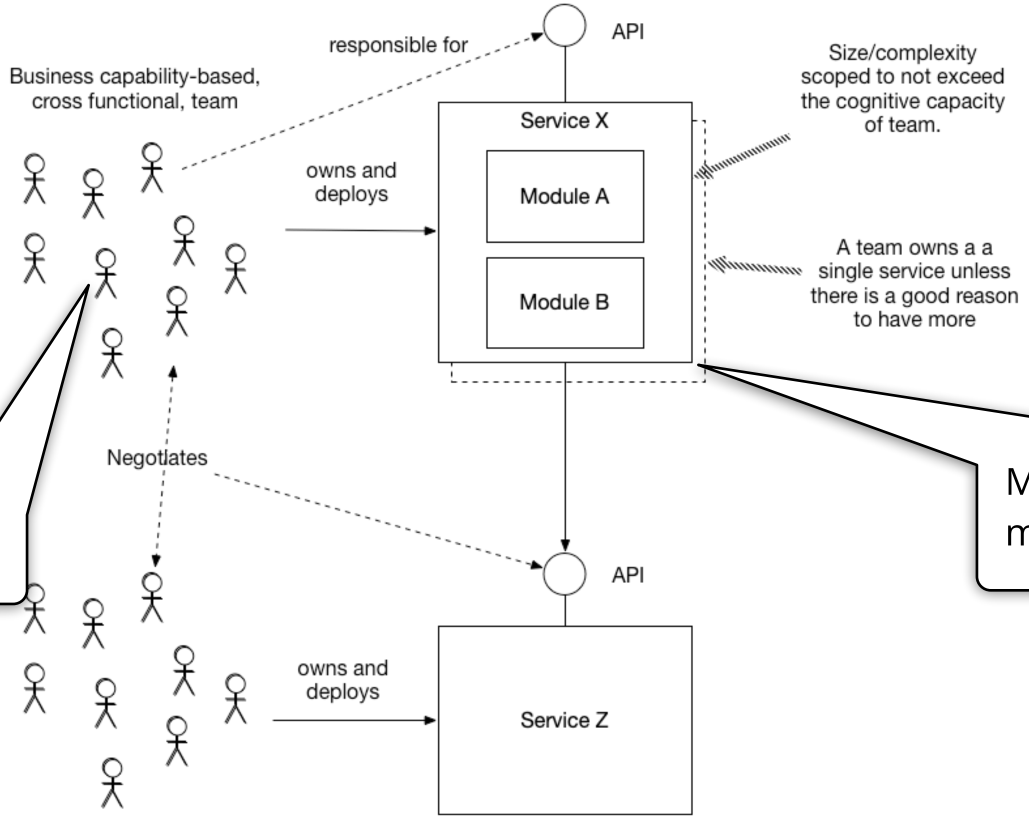
Learning Goals

- Examine how documentation and testing can be used to aid collaborative development across teams
- Reason about different testing approaches and their associated tradeoffs
- Learn how testability affects development and how it can be improved

Previously: Microservices



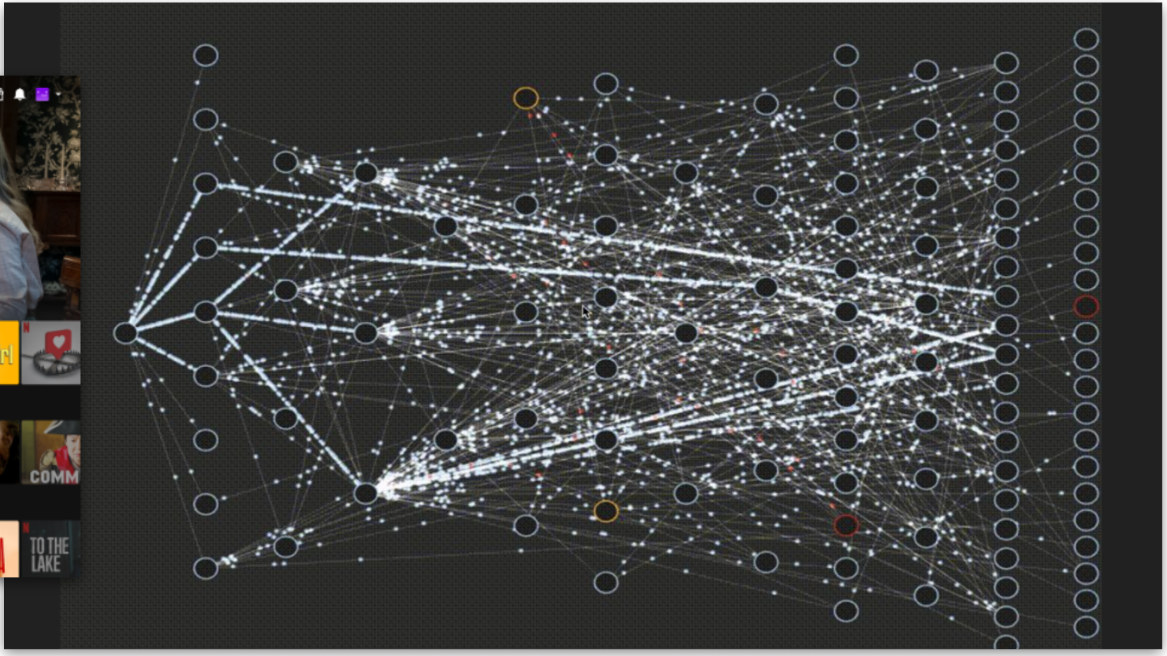
Challenge: Communication and Coordination


responsible for
API
Business capability-based, cross functional, team
Size/complexity scoped to not exceed the cognitive capacity of team.
owns and deploys
Service X
Module A
Module B
A team owns a a single service unless there is a good reason to have more
Negotiates
API
owns and deploys
Service Z

Located in a different physical location

May interact with many other services

You might have a lot of microservices!



(as of 2016)

https://www.youtube.com/watch?v=CZ3wIuvmHeM

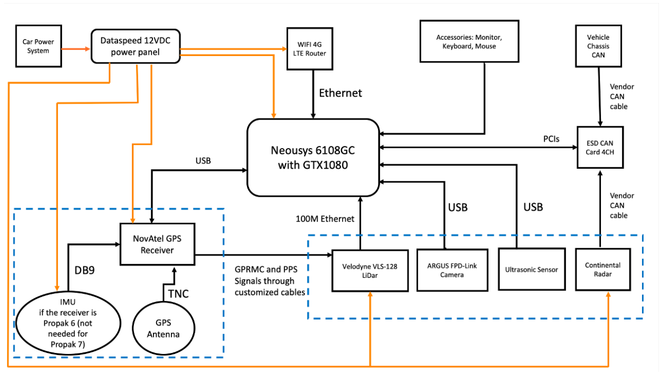
S3D

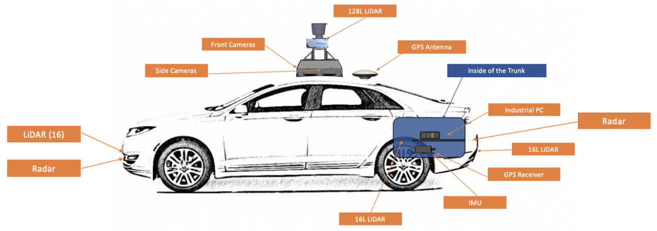
Integration Woes in Practice: Teedy

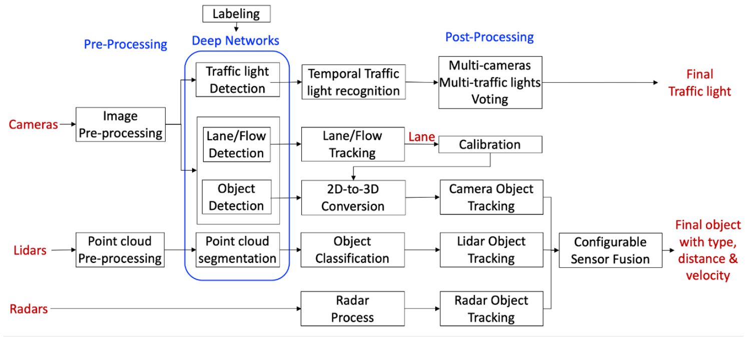
- Problems when integrating the frontend and backend?

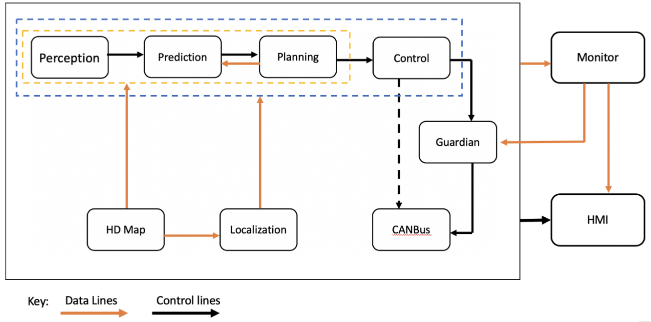
How can we avoid these problems?

Architecture diagrams give a big picture view

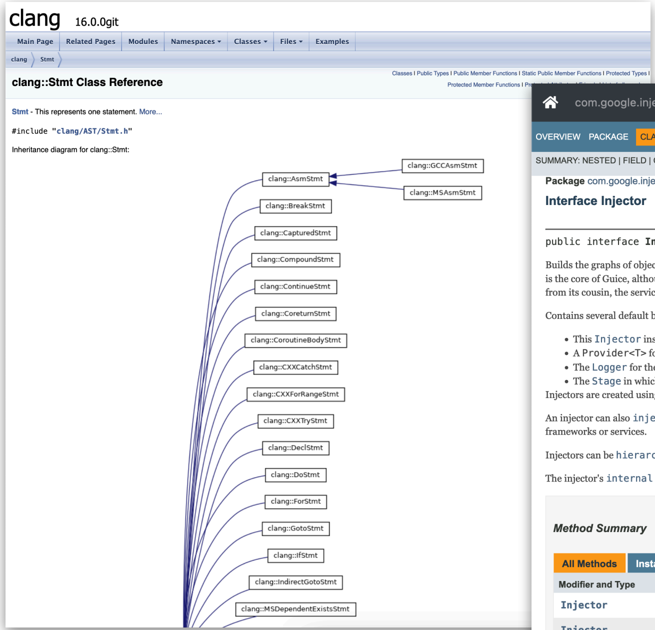








Code-Level API Documentation



clang 16.0.0git

Main Page | Related Pages | Modules | Namespaces | Classes | Files | Examples

clang › Stmt

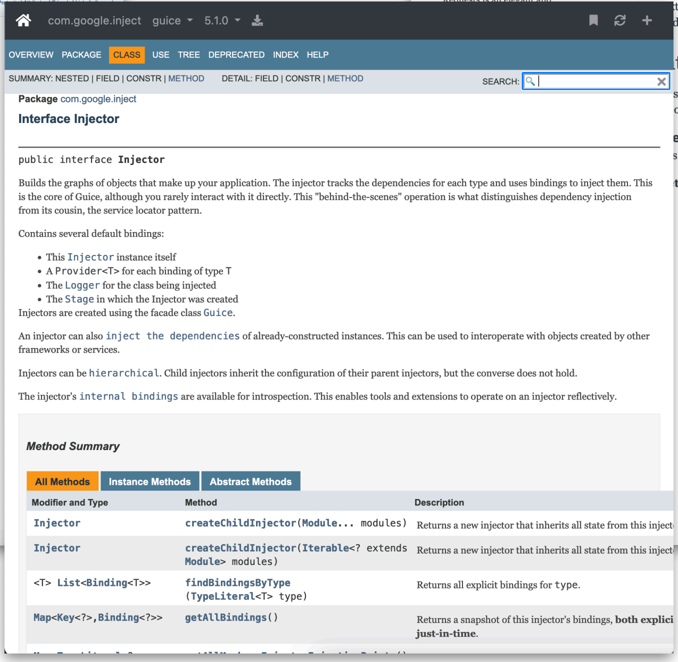
Classes | Public Types | Public Member Functions | Static Public Member Functions | Protected Types |
Protected Member Functions | Pr

clang::Stmt Class Reference

Stmt - This represents one statement. More...

```
#include "clang/AST/Stmt.h"
```

Inheritance diagram for clang::Stmt:



com.google.inject guice 5.1.0
OVERVIEW PACKAGE CLASS USE TREE DEPRECATED INDEX HELP
SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD
SEARCH:

Package com.google.inject

Interface Injector

```
public interface Injector
```

Builds the graphs of objects that make up your application. The injector tracks the dependencies for each type and uses bindings to inject them. This is the core of Guice, although you rarely interact with it directly. This "behind-the-scenes" operation is what distinguishes dependency injection from its cousin, the service locator pattern.

Contains several default bindings:

- This `Injector` instance itself
- A `Provider<T>` for each binding of type T
- The `Logger` for the class being injected
- The `Stage` in which the Injector was created

Injectors are created using the facade class `Guice`.

An injector can also `inject the dependencies` of already-constructed instances. This can be used to interoperate with objects created by other frameworks or services.

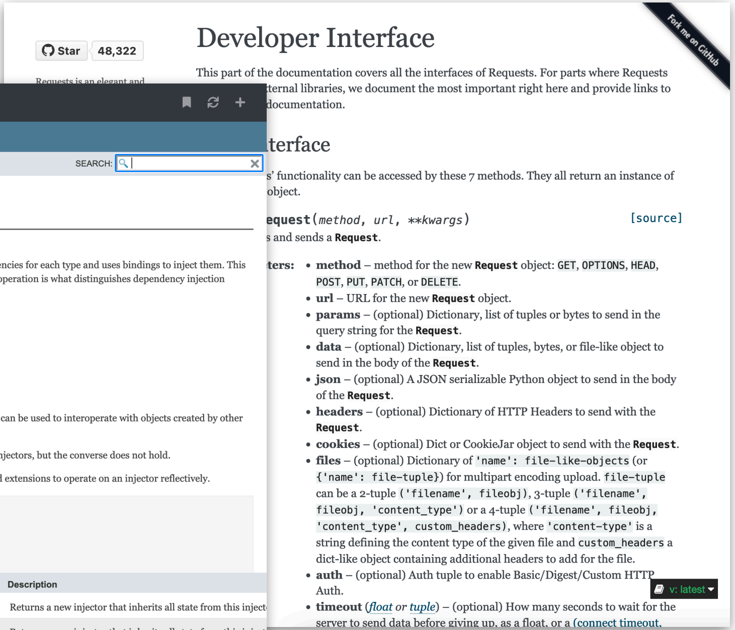
Injectors can be `hierarchical`. Child injectors inherit the configuration of their parent injectors, but the converse does not hold.

The injector's `internal bindings` are available for introspection. This enables tools and extensions to operate on an injector reflectively.

Method Summary

All Methods | Instance Methods | Abstract Methods

Modifier and Type	Method	Description
`Injector`	`createChildInjector(Module... modules)`	Returns a new injector that inherits all state from this inject
`Injector`	`createChildInjector(Iterable<? extends Module> modules)`	Returns a new injector that inherits all state from this inject
`<T> List<Binding<T>>`	`findBindingsByType (TypeLiteral<T> type)`	Returns all explicit bindings for `type`.
`Map<Key<?>,Binding<?>>`	`getAllBindings()`	Returns a snapshot of this injector's bindings, **both explici just-in-time**.



Requests is an elegant and

Developer Interface

This part of the documentation covers all the interfaces of Requests. For parts where Requests
ternal libraries, we document the most important right here and provide links to
documentation.

terface

s' functionality can be accessed by these 7 methods. They all return an instance of
object.

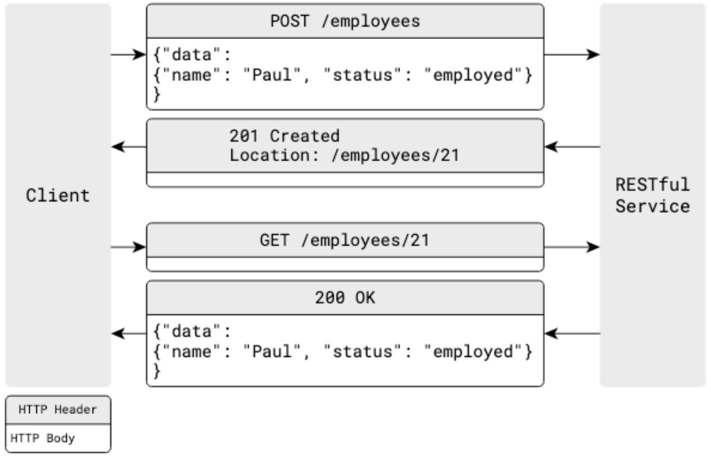
`equest(method, url, **kwargs)` [source]

s and sends a **Request**.

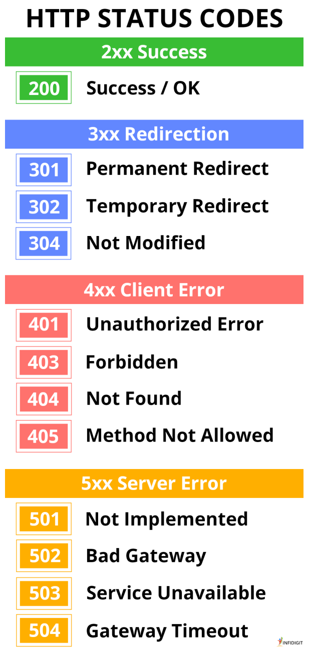
ters:

- **method** – method for the new **Request** object: `GET`, `OPTIONS`, `HEAD`, `POST`, `PUT`, `PATCH`, or `DELETE`.
- **url** – URL for the new **Request** object.
- **params** – (optional) Dictionary, list of tuples or bytes to send in the query string for the **Request**.
- **data** – (optional) Dictionary, list of tuples, bytes, or file-like object to send in the body of the **Request**.
- **json** – (optional) A JSON serializable Python object to send in the body of the **Request**.
- **headers** – (optional) Dictionary of HTTP Headers to send with the **Request**.
- **cookies** – (optional) Dict or CookieJar object to send with the **Request**.
- **files** – (optional) Dictionary of `'name': file-like-objects` (or `{'name': file-tuple}`) for multipart encoding upload. `file-tuple` can be a 2-tuple `('filename', fileobj)`, 3-tuple `('filename', fileobj, 'content_type')` or a 4-tuple `('filename', fileobj, 'content_type', custom_headers)`, where `'content-type'` is a string defining the content type of the given file and `custom_headers` a dict-like object containing additional headers to add for the file.
- **auth** – (optional) Auth tuple to enable Basic/Digest/Custom HTTP Auth.
- **timeout** (*float* or *tuple*) – (optional) How many seconds to wait for the server to send data before giving up, as a float, or a (connect timeout,

RESTful APIs: Nouns and Verbs

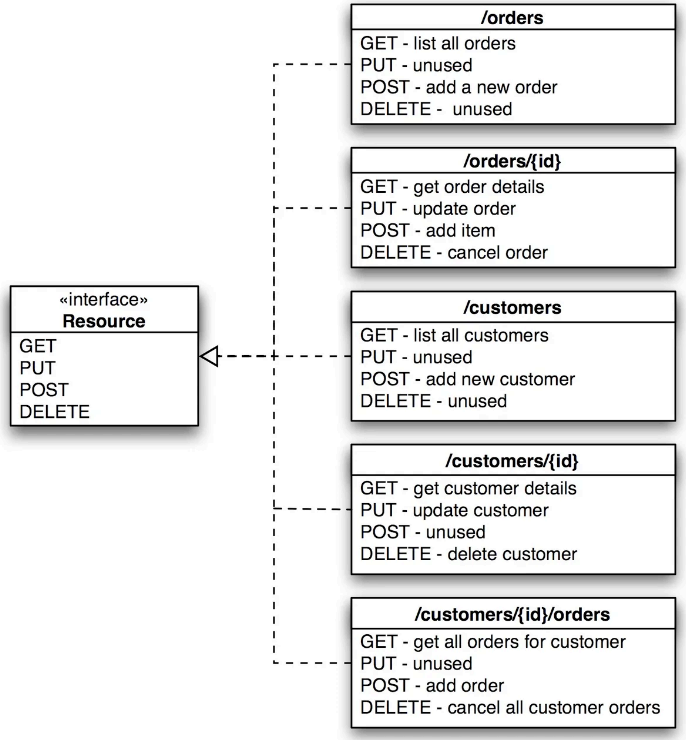


```
POST /employees
{"data":
{"name": "Paul", "status": "employed"}
}

201 Created
Location: /employees/21

GET /employees/21

200 OK
{"data":
{"name": "Paul", "status": "employed"}
}
```



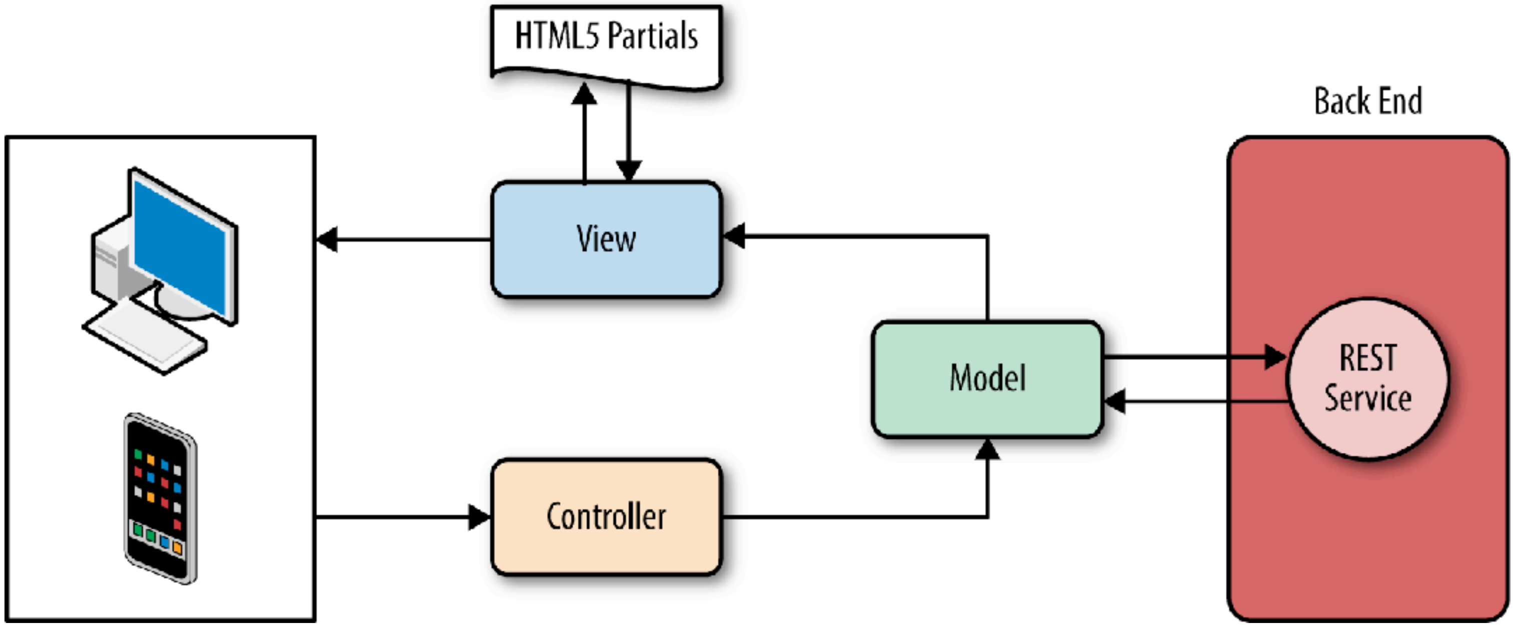
HTTP STATUS CODES

2xx Success	
200	Success / OK
3xx Redirection	
301	Permanent Redirect
302	Temporary Redirect
304	Not Modified
4xx Client Error	
401	Unauthorized Error
403	Forbidden
404	Not Found
405	Method Not Allowed
5xx Server Error	
501	Not Implemented
502	Bad Gateway
503	Service Unavailable
504	Gateway Timeout

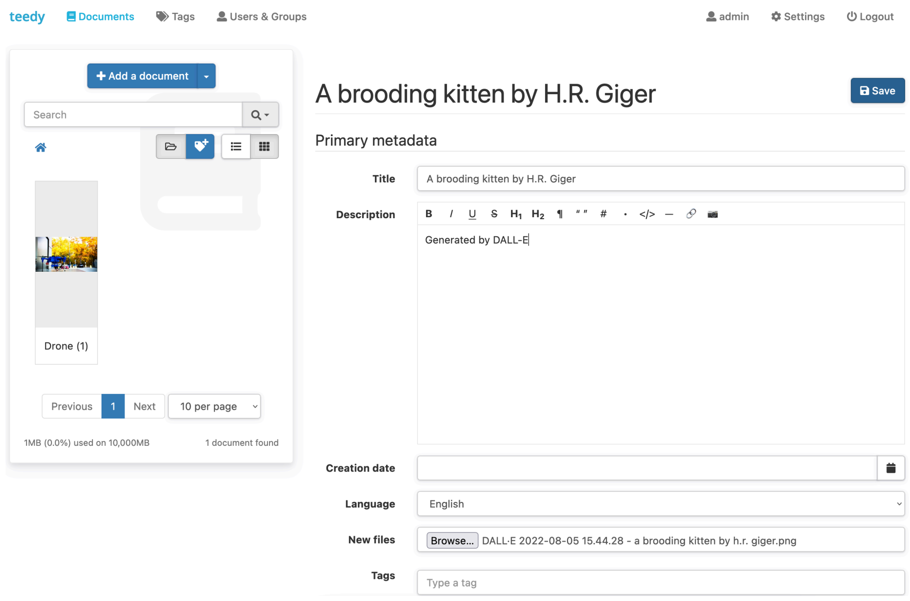


https://www.infoq.com/articles/rest-introduction

REST is used in Client–Server Architectures, too



REST in Action: Teedy



teedy · Documents · Tags · Users & Groups · admin · Settings · Logout

Add a document

Drone (1)

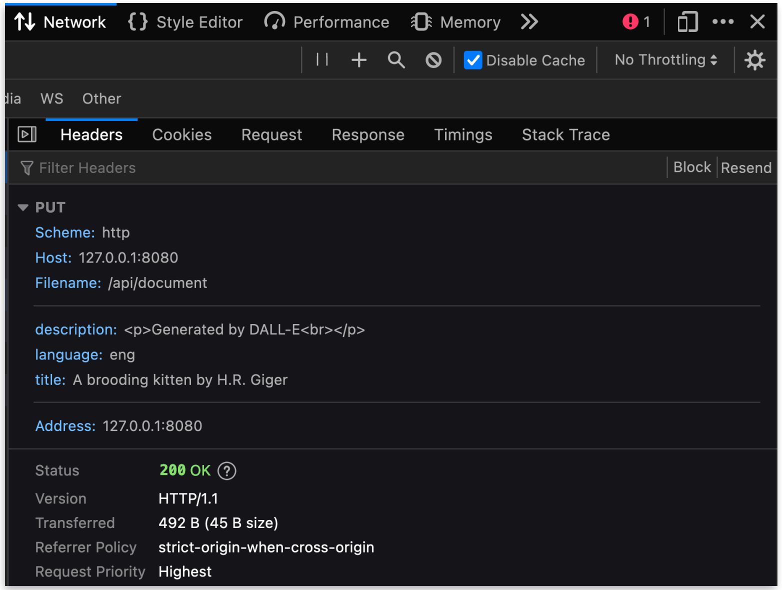
Previous 1 Next 10 per page

1MB (0.0%) used on 10,000MB · 1 document found

A brooding kitten by H.R. Giger

Primary metadata

Title	A brooding kitten by H.R. Giger
Description	Generated by DALL-E
Creation date	
Language	English
New files	DALL·E 2022-08-05 15.44.28 - a brooding kitten by h.r. giger.png
Tags	Type a tag



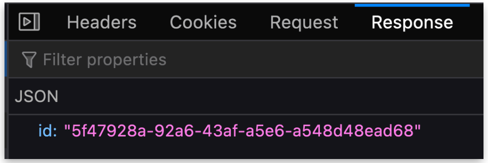
PUT

Scheme: http
Host: 127.0.0.1:8080
Filename: /api/document

description: <p>Generated by DALL-E
</p>
language: eng
title: A brooding kitten by H.R. Giger

Address: 127.0.0.1:8080

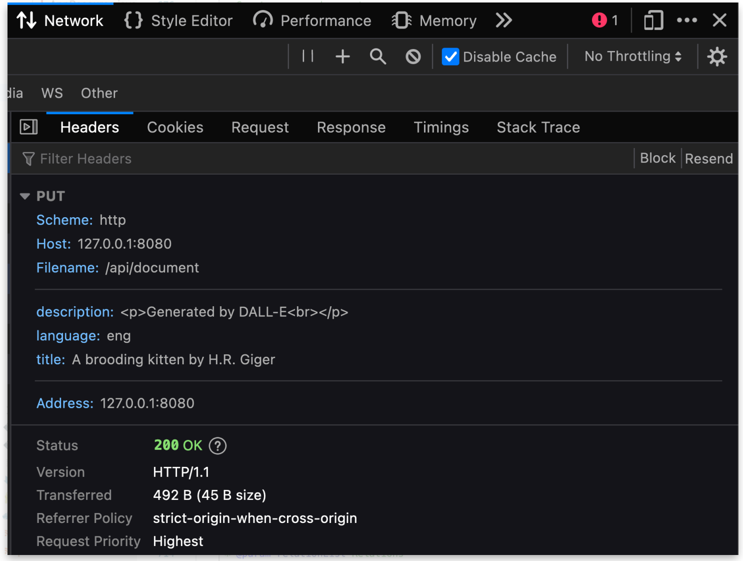
Status	200 OK
Version	HTTP/1.1
Transferred	492 B (45 B size)
Referrer Policy	strict-origin-when-cross-origin
Request Priority	Highest



JSON

id: "5f47928a-92a6-43af-a5e6-a548d48ead68"

REST in Action: Teedy

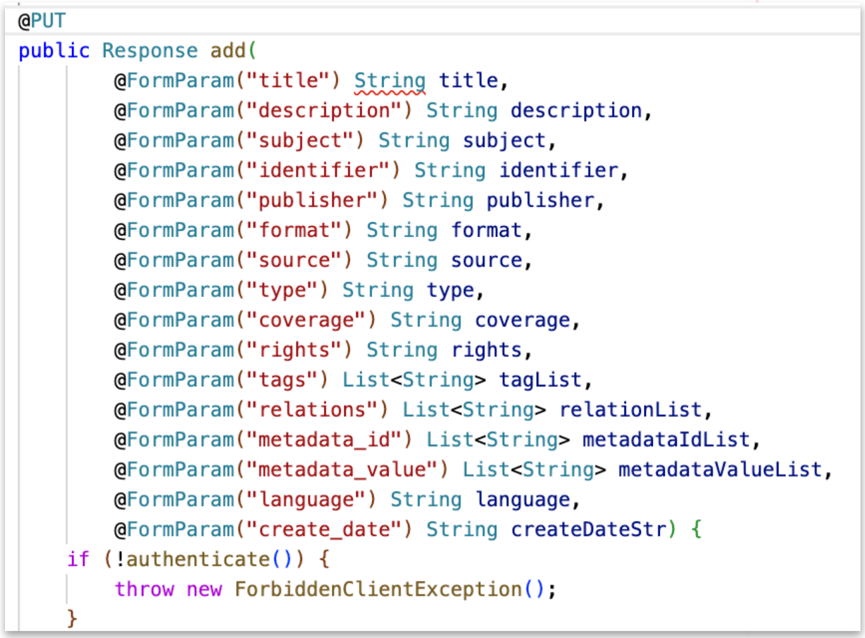


Network · Style Editor · Performance · Memory

Disable Cache · No Throttling

Headers · Cookies · Request · Response · Timings · Stack Trace

```
PUT
  Scheme: http
  Host: 127.0.0.1:8080
  Filename: /api/document

  description: <p>Generated by DALL-E<br></p>
  language: eng
  title: A brooding kitten by H.R. Giger

  Address: 127.0.0.1:8080

  Status            200 OK
  Version           HTTP/1.1
  Transferred       492 B (45 B size)
  Referrer Policy   strict-origin-when-cross-origin
  Request Priority  Highest
```

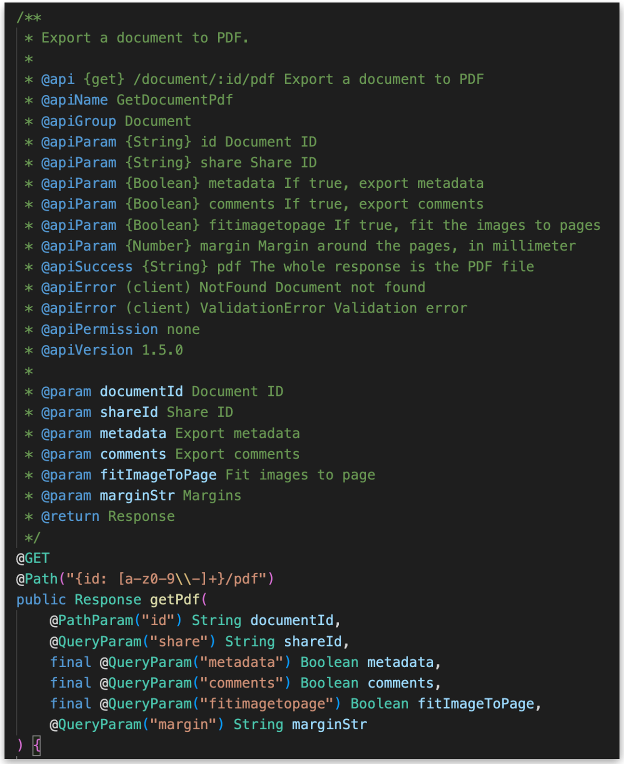


```java
@PUT
public Response add(
        @FormParam("title") String title,
        @FormParam("description") String description,
        @FormParam("subject") String subject,
        @FormParam("identifier") String identifier,
        @FormParam("publisher") String publisher,
        @FormParam("format") String format,
        @FormParam("source") String source,
        @FormParam("type") String type,
        @FormParam("coverage") String coverage,
        @FormParam("rights") String rights,
        @FormParam("tags") List<String> tagList,
        @FormParam("relations") List<String> relationList,
        @FormParam("metadata_id") List<String> metadataIdList,
        @FormParam("metadata_value") List<String> metadataValueList,
        @FormParam("language") String language,
        @FormParam("create_date") String createDateStr) {
    if (!authenticate()) {
        throw new ForbiddenClientException();
    }
```

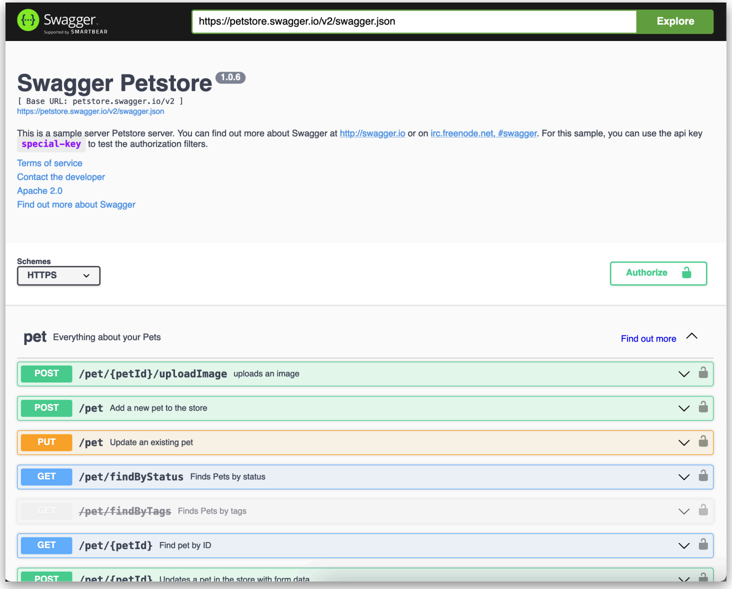
How can we enable collaborative design?

- Can we allow all teams to work in parallel without blocking on one another?
- How do service providers and consumers know what to implement and interact with?

API Documentation: OpenAPI (Swagger)



```java
/**
 * Export a document to PDF.
 *
 * @api {get} /document/:id/pdf Export a document to PDF
 * @apiName GetDocumentPdf
 * @apiGroup Document
 * @apiParam {String} id Document ID
 * @apiParam {String} share Share ID
 * @apiParam {Boolean} metadata If true, export metadata
 * @apiParam {Boolean} comments If true, export comments
 * @apiParam {Boolean} fitimagetopage If true, fit the images to pages
 * @apiParam {Number} margin Margin around the pages, in millimeter
 * @apiSuccess {String} pdf The whole response is the PDF file
 * @apiError (client) NotFound Document not found
 * @apiError (client) ValidationError Validation error
 * @apiPermission none
 * @apiVersion 1.5.0
 *
 * @param documentId Document ID
 * @param shareId Share ID
 * @param metadata Export metadata
 * @param comments Export comments
 * @param fitImageToPage Fit images to page
 * @param marginStr Margins
 * @return Response
 */
@GET
@Path("{id: [a-z0-9\\-]+}/pdf")
public Response getPdf(
    @PathParam("id") String documentId,
    @QueryParam("share") String shareId,
    final @QueryParam("metadata") Boolean metadata,
    final @QueryParam("comments") Boolean comments,
    final @QueryParam("fitimagetopage") Boolean fitImageToPage,
    @QueryParam("margin") String marginStr
) {
```



https://petstore.swagger.io/v2/swagger.json

Swagger Petstore 1.0.6

[Base URL: petstore.swagger.io/v2]
https://petstore.swagger.io/v2/swagger.json

This is a sample server Petstore server. You can find out more about Swagger at http://swagger.io or on irc.freenode.net, #swagger. For this sample, you can use the api key `special-key` to test the authorization filters.

Terms of service
Contact the developer
Apache 2.0
Find out more about Swagger

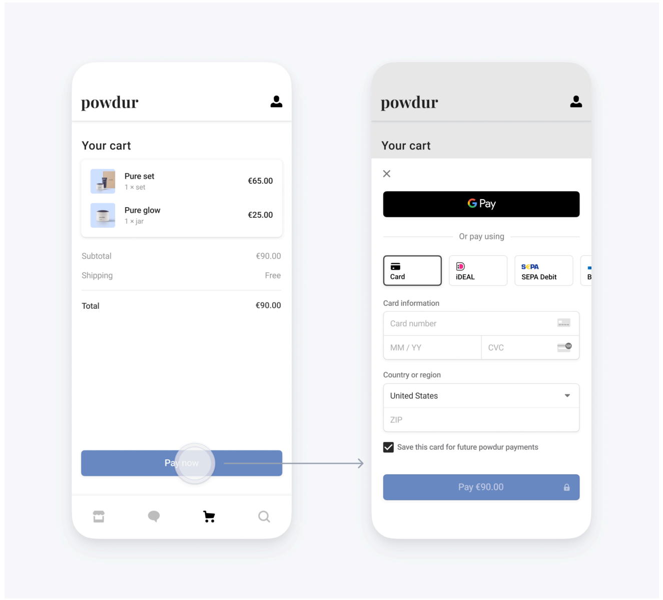
Schemes: HTTPS

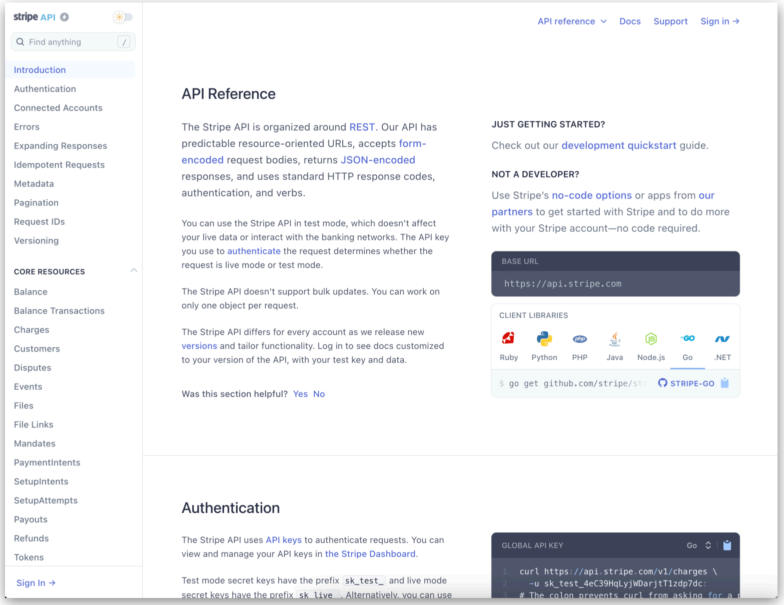
pet Everything about your Pets — Find out more

POST	/pet/{petId}/uploadImage	uploads an image
POST	/pet	Add a new pet to the store
PUT	/pet	Update an existing pet
GET	/pet/findByStatus	Finds Pets by status
GET	~~/pet/findByTags~~	Finds Pets by tags
GET	/pet/{petId}	Find pet by ID
POST	/pet/{petId}	Updates a pet in the store with form data

https://swagger.io

Swagger for Real: Stripe





Introduction
Authentication
Connected Accounts
Errors
Expanding Responses
Idempotent Requests
Metadata
Pagination
Request IDs
Versioning

CORE RESOURCES
Balance
Balance Transactions
Charges
Customers
Disputes
Events
Files
File Links
Mandates
PaymentIntents
SetupIntents
SetupAttempts
Payouts
Refunds
Tokens

Sign In →

API Reference

The Stripe API is organized around REST. Our API has predictable resource-oriented URLs, accepts form-encoded request bodies, returns JSON-encoded responses, and uses standard HTTP response codes, authentication, and verbs.

You can use the Stripe API in test mode, which doesn't affect your live data or interact with the banking networks. The API key you use to authenticate the request determines whether the request is live mode or test mode.

The Stripe API doesn't support bulk updates. You can work on only one object per request.

The Stripe API differs for every account as we release new versions and tailor functionality. Log in to see docs customized to your version of the API, with your test key and data.

Was this section helpful? Yes No

JUST GETTING STARTED?

Check out our development quickstart guide.

NOT A DEVELOPER?

Use Stripe's no-code options or apps from our partners to get started with Stripe and to do more with your Stripe account—no code required.

BASE URL

```
https://api.stripe.com
```

CLIENT LIBRARIES

Ruby Python PHP Java Node.js Go .NET

```
$ go get github.com/stripe/st
```

STRIPE-GO

Authentication

The Stripe API uses API keys to authenticate requests. You can view and manage your API keys in the Stripe Dashboard.

Test mode secret keys have the prefix `sk_test_` and live mode secret keys have the prefix `sk_live_`. Alternatively, you can use

GLOBAL API KEY Go

```
curl https://api.stripe.com/v1/charges \
  -u sk_test_4eC39HqLyjWDarjtT1zdp7dc:
# The colon prevents curl from asking for a
```

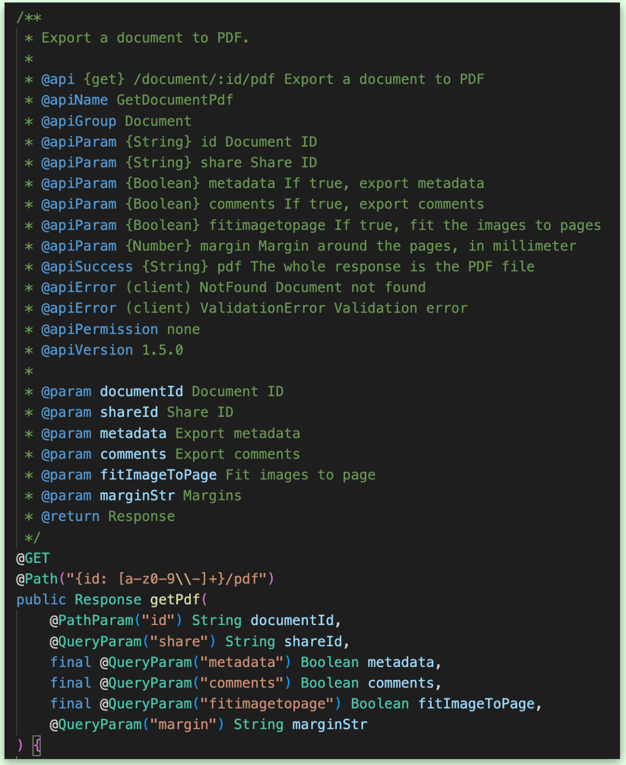
stripe

https://stripe.com/docs/api

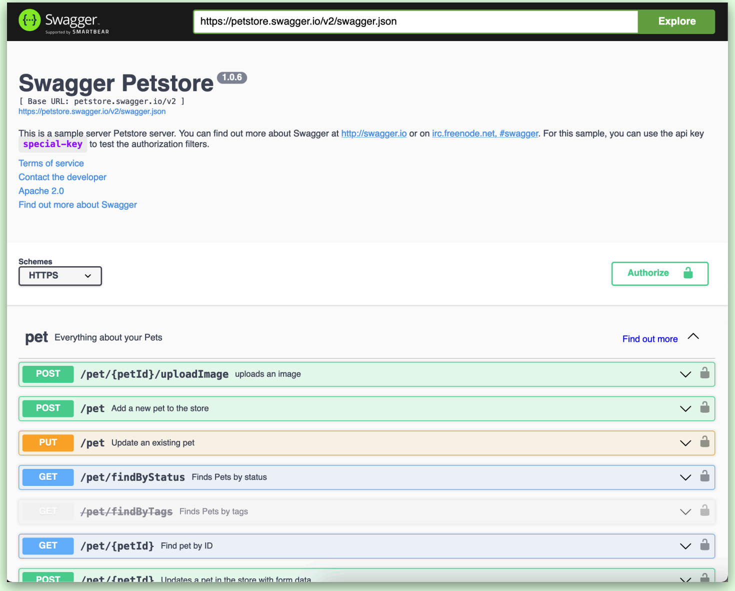
https://github.com/stripe/openapi

S3D

Exercise: Let’s Document Teedy



```
/**
 * Export a document to PDF.
 *
 * @api {get} /document/:id/pdf Export a document to PDF
 * @apiName GetDocumentPdf
 * @apiGroup Document
 * @apiParam {String} id Document ID
 * @apiParam {String} share Share ID
 * @apiParam {Boolean} metadata If true, export metadata
 * @apiParam {Boolean} comments If true, export comments
 * @apiParam {Boolean} fitimagetopage If true, fit the images to pages
 * @apiParam {Number} margin Margin around the pages, in millimeter
 * @apiSuccess {String} pdf The whole response is the PDF file
 * @apiError (client) NotFound Document not found
 * @apiError (client) ValidationError Validation error
 * @apiPermission none
 * @apiVersion 1.5.0
 *
 * @param documentId Document ID
 * @param shareId Share ID
 * @param metadata Export metadata
 * @param comments Export comments
 * @param fitImageToPage Fit images to page
 * @param marginStr Margins
 * @return Response
 */
@GET
@Path("{id: [a-z0-9\\-]+}/pdf")
public Response getPdf(
    @PathParam("id") String documentId,
    @QueryParam("share") String shareId,
    final @QueryParam("metadata") Boolean metadata,
    final @QueryParam("comments") Boolean comments,
    final @QueryParam("fitimagetopage") Boolean fitImageToPage,
    @QueryParam("margin") String marginStr
) {
```



Swagger Petstore 1.0.6

[Base URL: petstore.swagger.io/v2]
https://petstore.swagger.io/v2/swagger.json

This is a sample server Petstore server. You can find out more about Swagger at http://swagger.io or on irc.freenode.net, #swagger. For this sample, you can use the api key `special-key` to test the authorization filters.

Terms of service
Contact the developer
Apache 2.0
Find out more about Swagger

Schemes: HTTPS

pet Everything about your Pets — Find out more

POST `/pet/{petId}/uploadImage` uploads an image
POST `/pet` Add a new pet to the store
PUT `/pet` Update an existing pet
GET `/pet/findByStatus` Finds Pets by status
GET ~~`/pet/findByTags`~~ Finds Pets by tags
GET `/pet/{petId}` Find pet by ID
POST `/pet/{petId}` Updates a pet in the store with form data

Collaborative Design via Documentation

- **Design:** OpenAPI docs, ...
- **Discuss:** Issue Tracker, Meetings, ...
- **Refine:** Pull Requests
- **Repeat**

S3D

Collaborative Development via Testing

- Catch bugs before they occur in production
- Gain confidence in the implementation
- **Drive the development process**
    - enable parallel development (chicken and egg problem!)
    - identify ambiguities in the design; find bugs in our ideas
    - encode assumptions and expectations
    - living, executable documentation
- ...

How should we test our systems?

Recap: Avoid manual testing

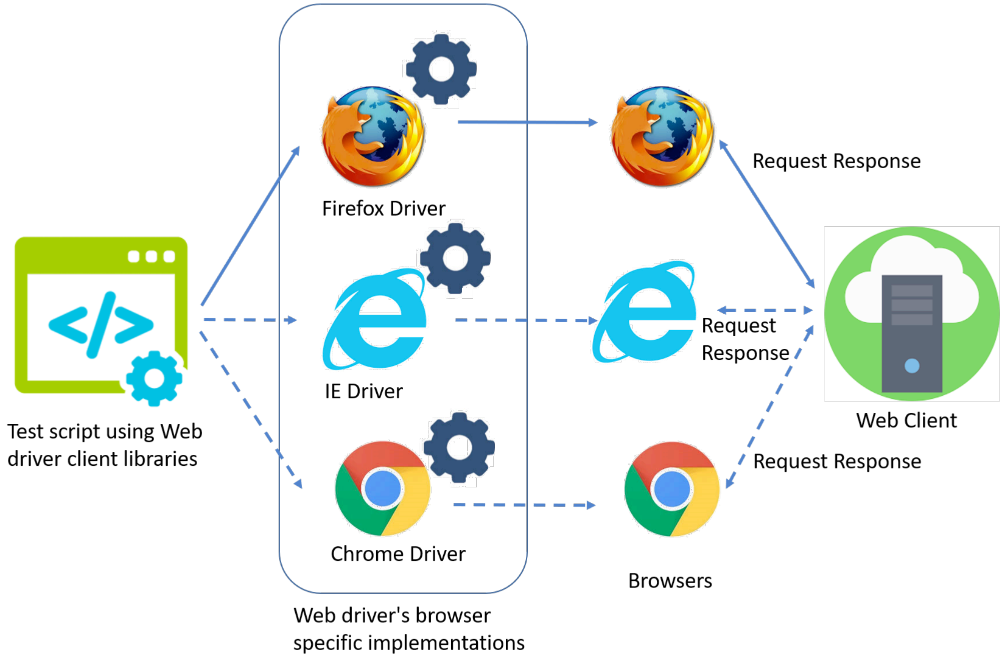
Automated Testing

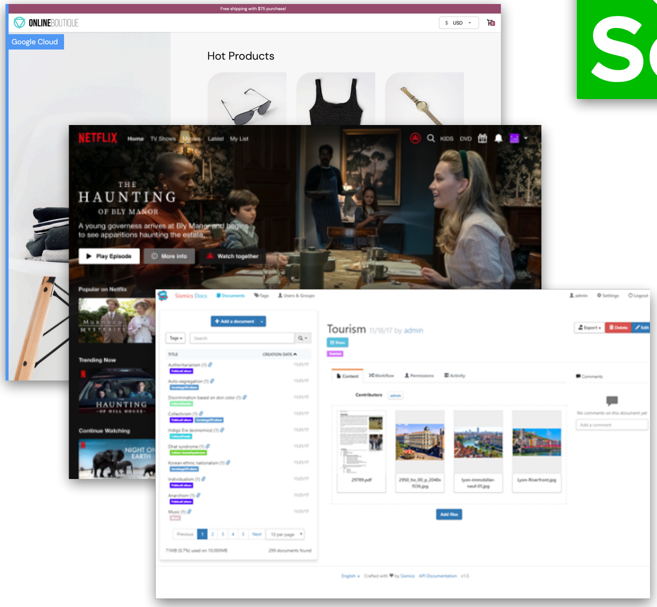
+ Reproducible
+ Some upfront effort
+ Zero marginal effort
+ Runs on every commit
+ Finds regressions!

Manual Testing

- Unreproducible
- Low upfront effort
- High marginal effort
- Runs when you remember
- Unsustainable

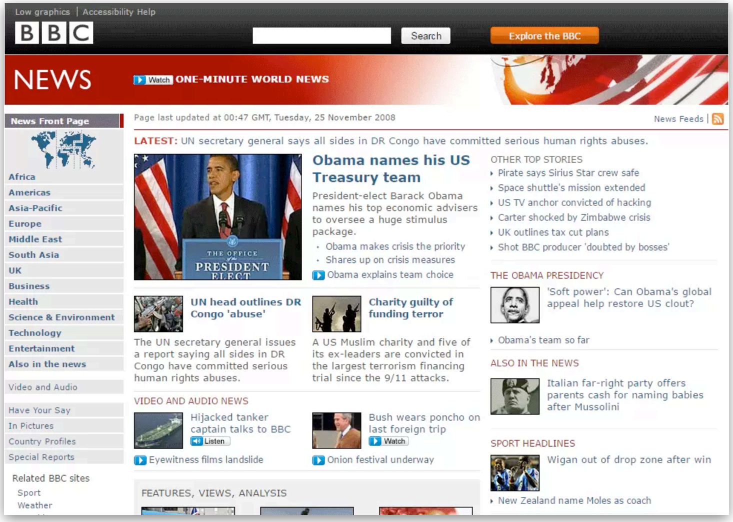
End-to-End Testing (E2E)





https://www.selenium.dev

End-to-end tests are fragile



Page last updated at 00:47 GMT, Tuesday, 25 November 2008

LATEST: UN secretary general says all sides in DR Congo have committed serious human rights abuses.

Obama names his US Treasury team

President-elect Barack Obama names his top economic advisers to oversee a huge stimulus package.

- Obama makes crisis the priority
- Shares up on crisis measures
- Obama explains team choice

UN head outlines DR Congo 'abuse'

The UN secretary general issues a report saying all sides in DR Congo have committed serious human rights abuses.

Charity guilty of funding terror

A US Muslim charity and five of its ex-leaders are convicted in the largest terrorism financing trial since the 9/11 attacks.

VIDEO AND AUDIO NEWS

Hijacked tanker captain talks to BBC

Bush wears poncho on last foreign trip

Eyewitness films landslide

Onion festival underway

FEATURES, VIEWS, ANALYSIS

OTHER TOP STORIES

- Pirate says Sirius Star crew safe
- Space shuttle's mission extended
- US TV anchor convicted of hacking
- Carter shocked by Zimbabwe crisis
- UK outlines tax cut plans
- Shot BBC producer 'doubted by bosses'

THE OBAMA PRESIDENCY

'Soft power': Can Obama's global appeal help restore US clout?

- Obama's team so far

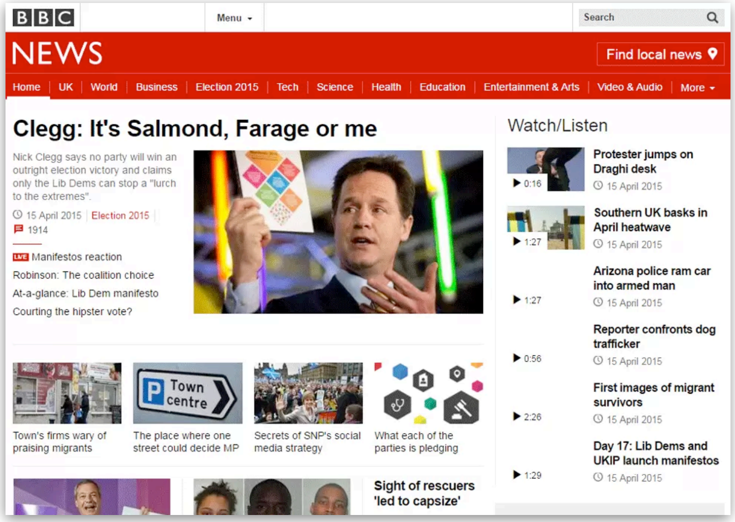
ALSO IN THE NEWS

Italian far-right party offers parents cash for naming babies after Mussolini

SPORT HEADLINES

Wigan out of drop zone after win

- New Zealand name Moles as coach



Clegg: It's Salmond, Farage or me

Nick Clegg says no party will win an outright election victory and claims only the Lib Dems can stop a "lurch to the extremes".

15 April 2015 | Election 2015

1914

LIVE Manifestos reaction

Robinson: The coalition choice

At-a-glance: Lib Dem manifesto

Courting the hipster vote?

Town's firms wary of praising migrants

The place where one street could decide MP

Secrets of SNP's social media strategy

What each of the parties is pledging

Sight of rescuers 'led to capsize'

Watch/Listen

Protester jumps on Draghi desk 0:16 15 April 2015

Southern UK basks in April heatwave 1:27 15 April 2015

Arizona police ram car into armed man 1:27 15 April 2015

Reporter confronts dog trafficker 0:56 15 April 2015

First images of migrant survivors 2:26 15 April 2015

Day 17: Lib Dems and UKIP launch manifestos 1:29 15 April 2015

Guess I'll rewrite the test suite.

https://www.bbc.com/news/uk-41890165

End-to-end tests can be difficult to automate

- We need to maintain a test environment
  - We don't run end-to-end tests in production
- Harder to run tests in parallel
  - Tests might affect one another
  - Race conditions
  - Sequential test execution for *idempotency*
- Software might only run on certain machines
  - Licensed third-party dependencies

https://www.alamy.com/stock-photo-vector-isometric-low-poly-movie-set-131706020.html

End-to-end tests are slow and expensive

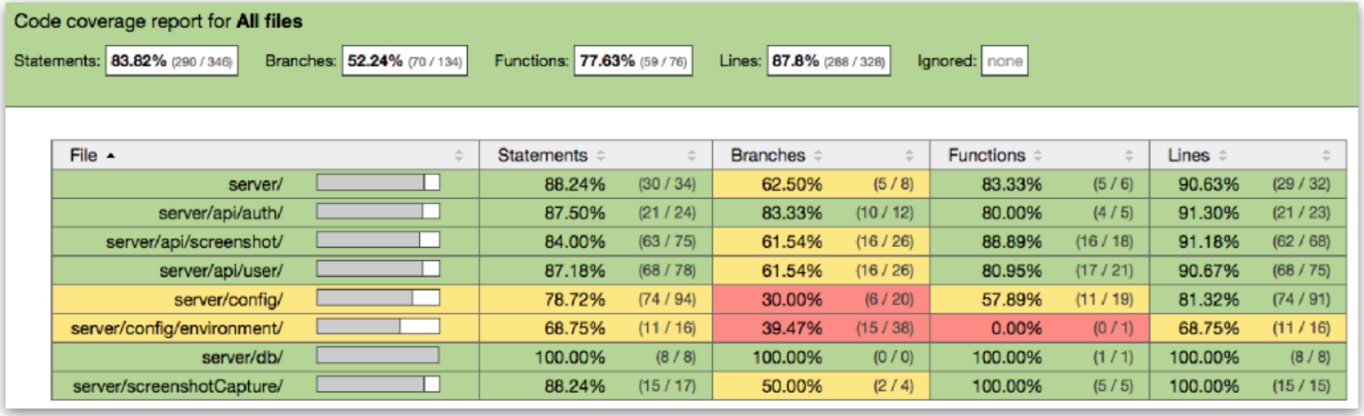
- License fees
- Longer start-up, tear-down, and execution times
- Consumes a lot of resources
- Slower release velocity

https://globalnews.ca/news/4208975/scientists-transfer-memory-snail/

S3D

End-to-end tests have high coverage but *poor test isolation*

- Does not isolate individual components
- Makes it harder to debug
- Redundancy between tests (e.g., initialization, route forwarding, ...)

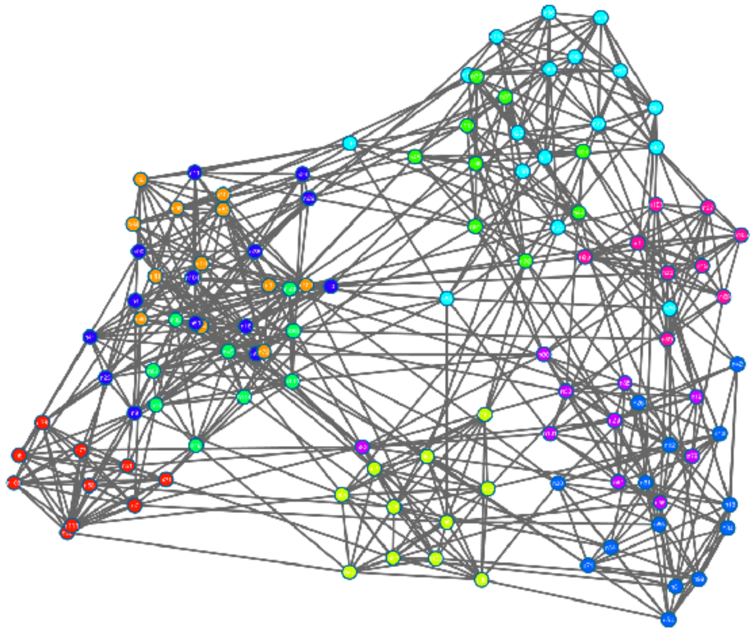


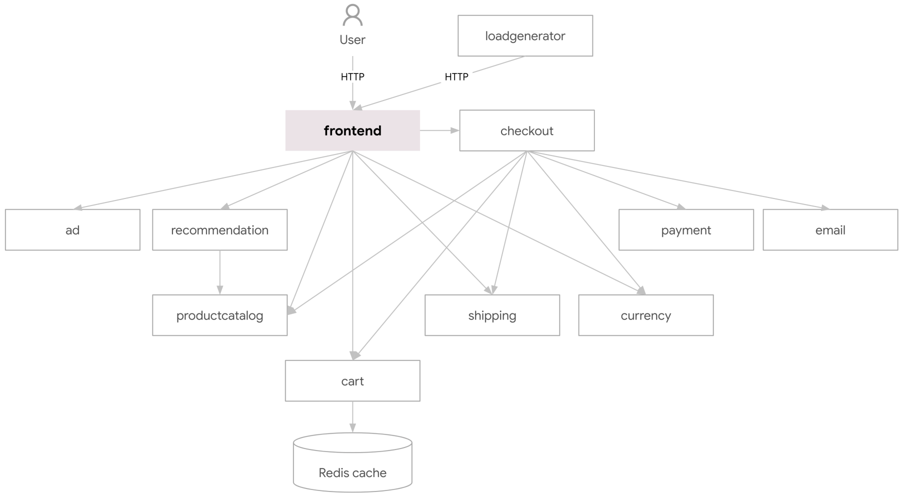
Code coverage report for **All files**

Statements: **83.82%** (290 / 346) Branches: **52.24%** (70 / 134) Functions: **77.63%** (59 / 76) Lines: **87.8%** (288 / 328) Ignored: none

File ▲	Statements		Branches		Functions		Lines	
server/	88.24%	(30 / 34)	62.50%	(5 / 8)	83.33%	(5 / 6)	90.63%	(29 / 32)
server/api/auth/	87.50%	(21 / 24)	83.33%	(10 / 12)	80.00%	(4 / 5)	91.30%	(21 / 23)
server/api/screenshot/	84.00%	(63 / 75)	61.54%	(16 / 26)	88.89%	(16 / 18)	91.18%	(62 / 68)
server/api/user/	87.18%	(68 / 78)	61.54%	(16 / 26)	80.95%	(17 / 21)	90.67%	(68 / 75)
server/config/	78.72%	(74 / 94)	30.00%	(6 / 20)	57.89%	(11 / 19)	81.32%	(74 / 91)
server/config/environment/	68.75%	(11 / 16)	39.47%	(15 / 38)	0.00%	(0 / 1)	68.75%	(11 / 16)
server/db/	100.00%	(8 / 8)	100.00%	(0 / 0)	100.00%	(1 / 1)	100.00%	(8 / 8)
server/screenshotCapture/	88.24%	(15 / 17)	50.00%	(2 / 4)	100.00%	(5 / 5)	100.00%	(15 / 15)

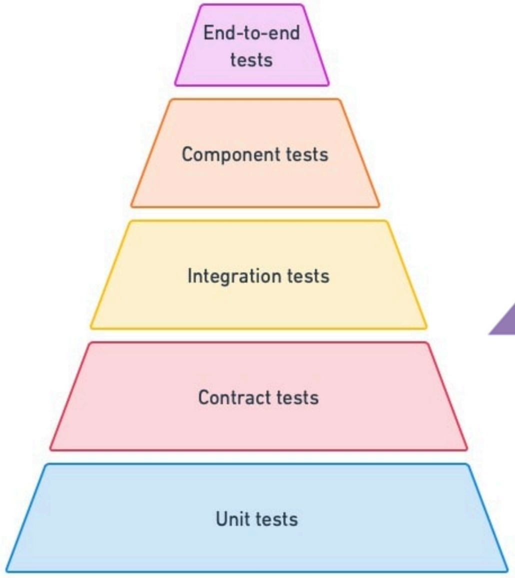
In E2E tests, the entire system is the system under test (SUT)

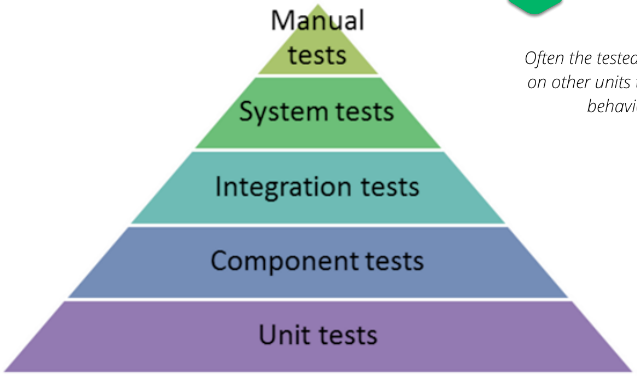


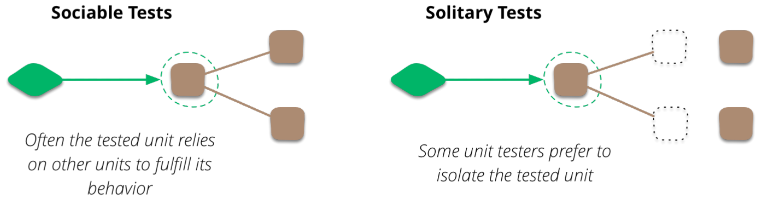


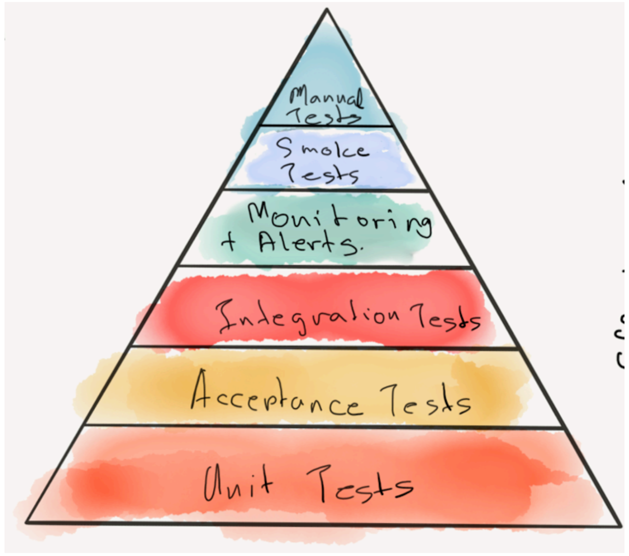
What is a unit test?

Beware of Testing Definitions!

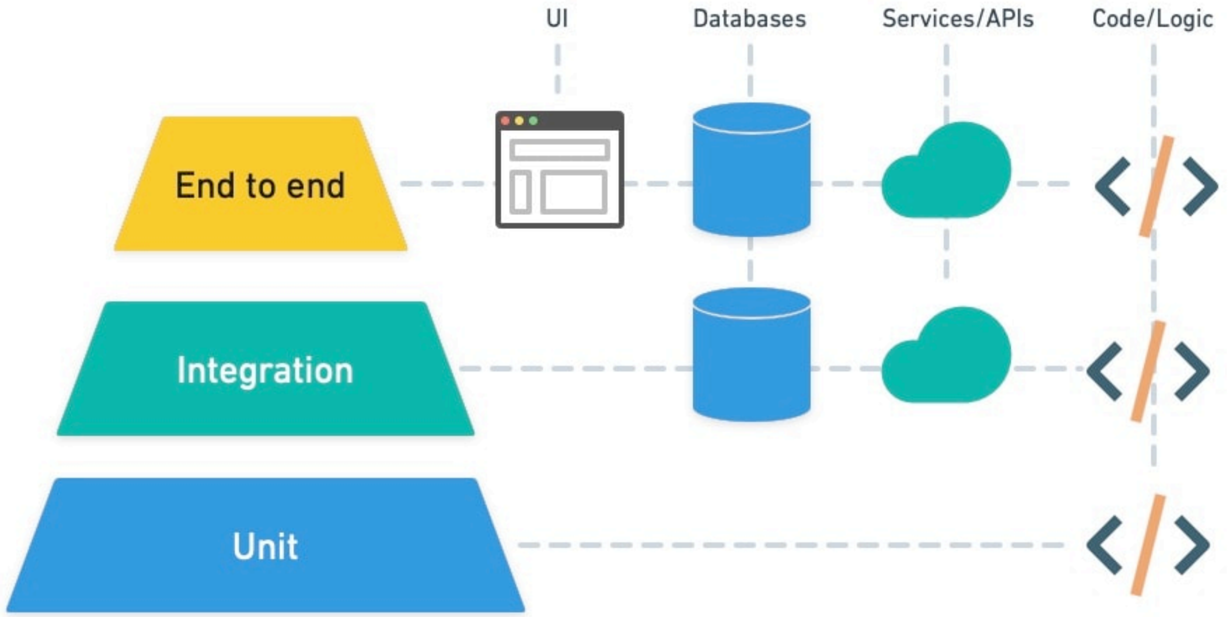





Sociable Tests
Often the tested unit relies on other units to fulfill its behavior
Solitary Tests
Some unit testers prefer to isolate the tested unit



A simple version of the Test Pyramid

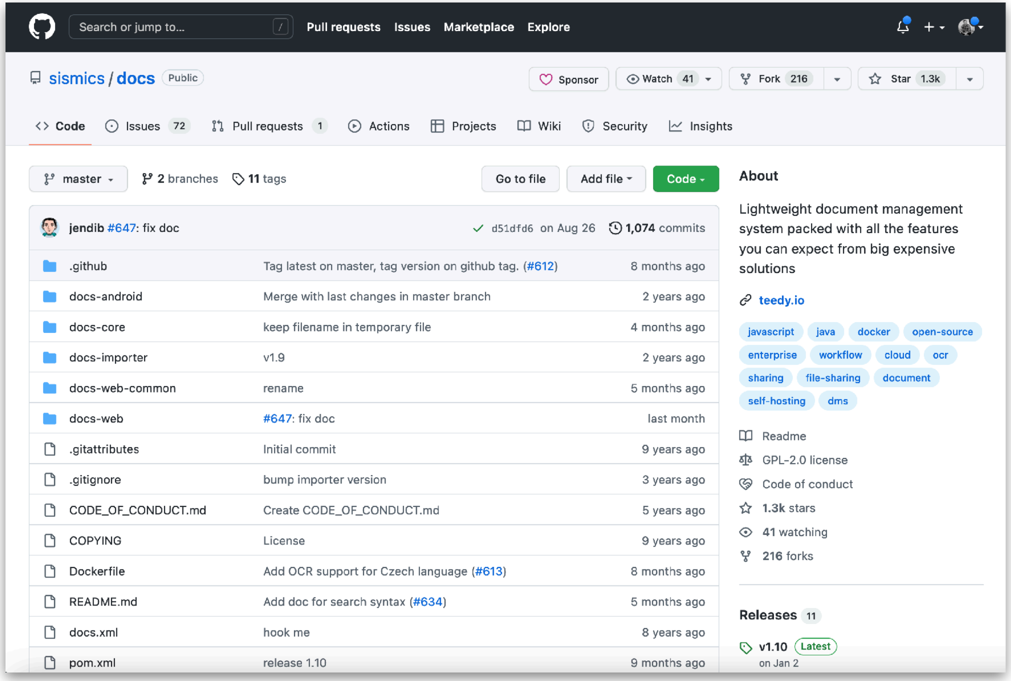


Brittle, Slow, Expensive, Imprecise

Fast, Cheap, Precise

https://semaphoreci.com/blog/testing-pyramid

Testing in the Wild: Teedy



jendib #647: fix doc — d51dfd6 on Aug 26 — 1,074 commits

.github	Tag latest on master, tag version on github tag. (#612)	8 months ago
docs-android	Merge with last changes in master branch	2 years ago
docs-core	keep filename in temporary file	4 months ago
docs-importer	v1.9	2 years ago
docs-web-common	rename	5 months ago
docs-web	#647: fix doc	last month
.gitattributes	Initial commit	9 years ago
.gitignore	bump importer version	3 years ago
CODE_OF_CONDUCT.md	Create CODE_OF_CONDUCT.md	5 years ago
COPYING	License	9 years ago
Dockerfile	Add OCR support for Czech language (#613)	8 months ago
README.md	Add doc for search syntax (#634)	5 months ago
docs.xml	hook me	8 years ago
pom.xml	release 1.10	9 months ago

About

Lightweight document management system packed with all the features you can expect from big expensive solutions

teedy.io

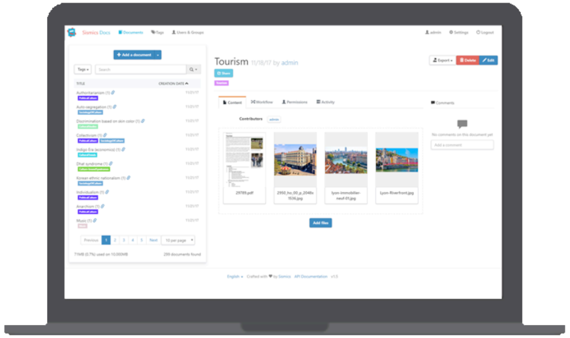
javascript, java, docker, open-source, enterprise, workflow, cloud, ocr, sharing, file-sharing, document, self-hosting, dms

Readme
GPL-2.0 license
Code of conduct
1.3k stars
41 watching
216 forks

Releases 11

v1.10 Latest
on Jan 2

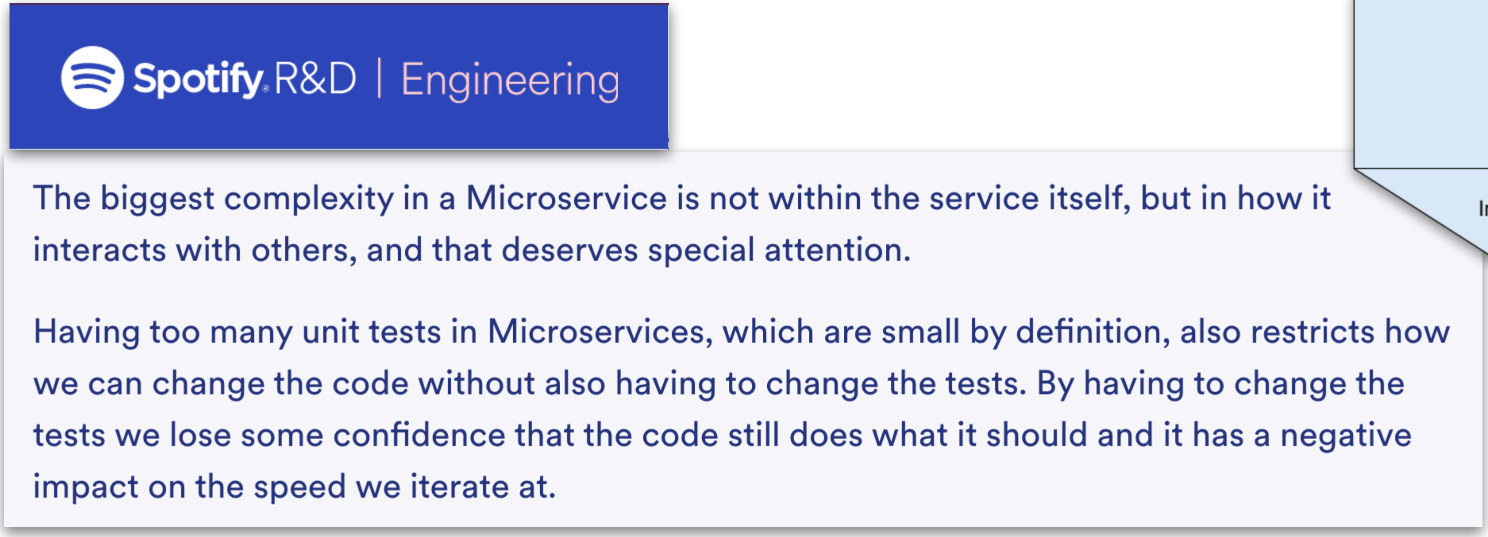
teedy



https://github.com/CMU-313/Teedy

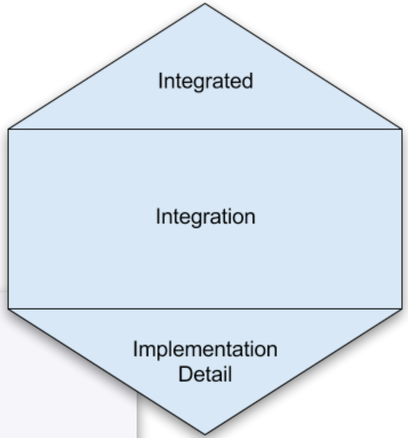
S3D

Testing in the Wild: Spotify


Spotify R&D | Engineering

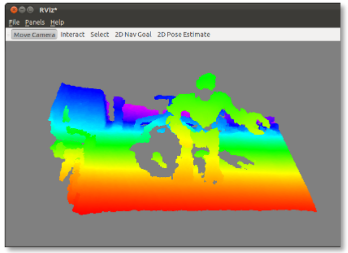
The biggest complexity in a Microservice is not within the service itself, but in how it interacts with others, and that deserves special attention.

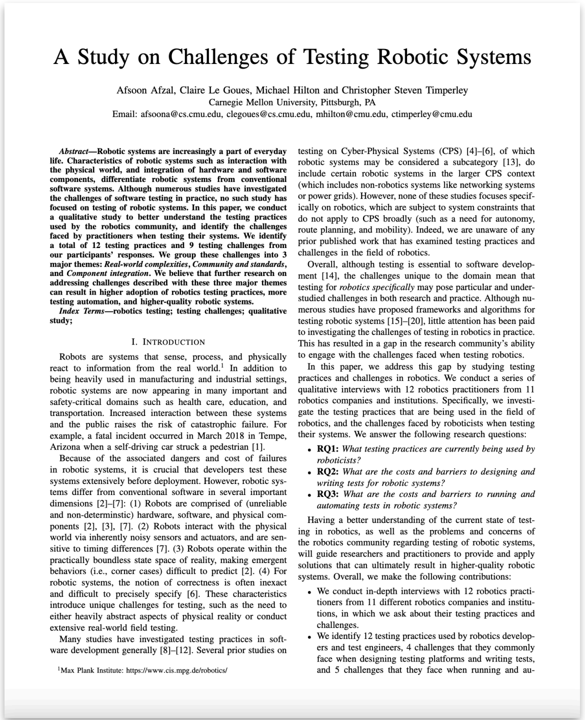
Having too many unit tests in Microservices, which are small by definition, also restricts how we can change the code without also having to change the tests. By having to change the tests we lose some confidence that the code still does what it should and it has a negative impact on the speed we iterate at.



https://engineering.atspotify.com/2018/01/testing-of-microservices

Testing in the Wild: Robots





A Study on Challenges of Testing Robotic Systems

Afsoon Afzal, Claire Le Goues, Michael Hilton and Christopher Steven Timperley
Carnegie Mellon University, Pittsburgh, PA
Email: afsoona@cs.cmu.edu, clegoues@cs.cmu.edu, mhilton@cmu.edu, ctimperley@cmu.edu

***Abstract*—Robotic systems are increasingly a part of everyday life. Characteristics of robotic systems such as interaction with the physical world, and integration of hardware and software components, differentiate robotic systems from conventional software systems. Although numerous studies have investigated the challenges of software testing in practice, no such study has focused on testing of robotic systems. In this paper, we conduct a qualitative study to better understand the testing practices used by the robotics community, and identify the challenges faced by practitioners when testing their systems. We identify a total of 12 testing practices and 9 testing challenges from our participants' responses. We group these challenges into 3 major themes: *Real-world complexities*, *Community and standards*, and *Component integration*. We believe that further research on addressing challenges described with these three major themes can result in higher adoption of robotics testing practices, more testing automation, and higher-quality robotic systems.**

***Index Terms*—robotics testing; testing challenges; qualitative study;**

I. Introduction

Robots are systems that sense, process, and physically react to information from the real world.[1] In addition to being heavily used in manufacturing and industrial settings, robotic systems are now appearing in many important and safety-critical domains such as health care, education, and transportation. Increased interaction between these systems and the public raises the risk of catastrophic failure. For example, a fatal incident occurred in March 2018 in Tempe, Arizona when a self-driving car struck a pedestrian [1].

Because of the associated dangers and cost of failures in robotic systems, it is crucial that developers test these systems extensively before deployment. However, robotic systems differ from conventional software in several important dimensions [2]–[7]: (1) Robots are comprised of (unreliable and non-determinstic) hardware, software, and physical components [2], [3], [7]. (2) Robots interact with the physical world via inherently noisy sensors and actuators, and are sensitive to timing differences [7]. (3) Robots operate within the practically boundless state space of reality, making emergent behaviors (i.e., corner cases) difficult to predict [2]. (4) For robotic systems, the notion of correctness is often inexact and difficult to precisely specify [6]. These characteristics introduce unique challenges for testing, such as the need to either heavily abstract aspects of physical reality or conduct extensive real-world field testing.

Many studies have investigated testing practices in software development generally [8]–[12]. Several prior studies on testing on Cyber-Physical Systems (CPS) [4]–[6], of which robotic systems may be considered a subcategory [13], do include certain robotic systems in the larger CPS context (which includes non-robotics systems like networking systems or power grids). However, none of these studies focuses specifically on robotics, which are subject to system constraints that do not apply to CPS broadly (such as a need for autonomy, route planning, and mobility). Indeed, we are unaware of any prior published work that has examined testing practices and challenges in the field of robotics.

Overall, although testing is essential to software development [14], the challenges unique to the domain mean that testing for *robotics specifically* may pose particular and understudied challenges in both research and practice. Although numerous studies have proposed frameworks and algorithms for testing robotic systems [15]–[20], little attention has been paid to investigating the challenges of testing in robotics in practice. This has resulted in a gap in the research community's ability to engage with the challenges faced when testing robotics.

In this paper, we address this gap by studying testing practices and challenges in robotics. We conduct a series of qualitative interviews with 12 robotics practitioners from 11 robotics companies and institutions. Specifically, we investigate the testing practices that are being used in the field of robotics, and the challenges faced by roboticists when testing their systems. We answer the following research questions:

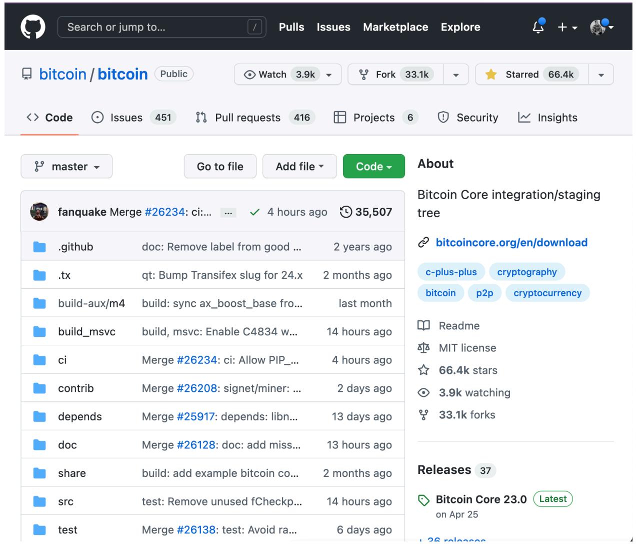
- **RQ1:** *What testing practices are currently being used by roboticists?*
- **RQ2:** *What are the costs and barriers to designing and writing tests for robotic systems?*
- **RQ3:** *What are the costs and barriers to running and automating tests in robotic systems?*

Having a better understanding of the current state of testing in robotics, as well as the problems and concerns of the robotics community regarding testing of robotic systems, will guide researchers and practitioners to provide and apply solutions that can ultimately result in higher-quality robotic systems. Overall, we make the following contributions:

- We conduct in-depth interviews with 12 robotics practitioners from 11 different robotics companies and institutions, in which we ask about their testing practices and challenges.
- We identify 12 testing practices used by robotics developers and test engineers, 4 challenges that they commonly face when designing testing platforms and writing tests, and 5 challenges that they face when running and au-

[1]Max Plank Institute: https://www.cis.mpg.de/robotics/

Testing in the Wild: Bitcoin



fanquake Merge #26234: ci:... … ✓ 4 hours ago 35,507

.github	doc: Remove label from good ...	2 years ago
.tx	qt: Bump Transifex slug for 24.x	2 months ago
build-aux/m4	build: sync ax_boost_base fro...	last month
build_msvc	build, msvc: Enable C4834 w...	14 hours ago
ci	Merge #26234: ci: Allow PIP_...	4 hours ago
contrib	Merge #26208: signet/miner: ...	2 days ago
depends	Merge #25917: depends: libn...	13 days ago
doc	Merge #26128: doc: add miss...	13 hours ago
share	build: add example bitcoin co...	2 months ago
src	test: Remove unused fCheckp...	14 hours ago
test	Merge #26138: test: Avoid ra...	6 days ago

About

Bitcoin Core integration/staging tree

bitcoincore.org/en/download

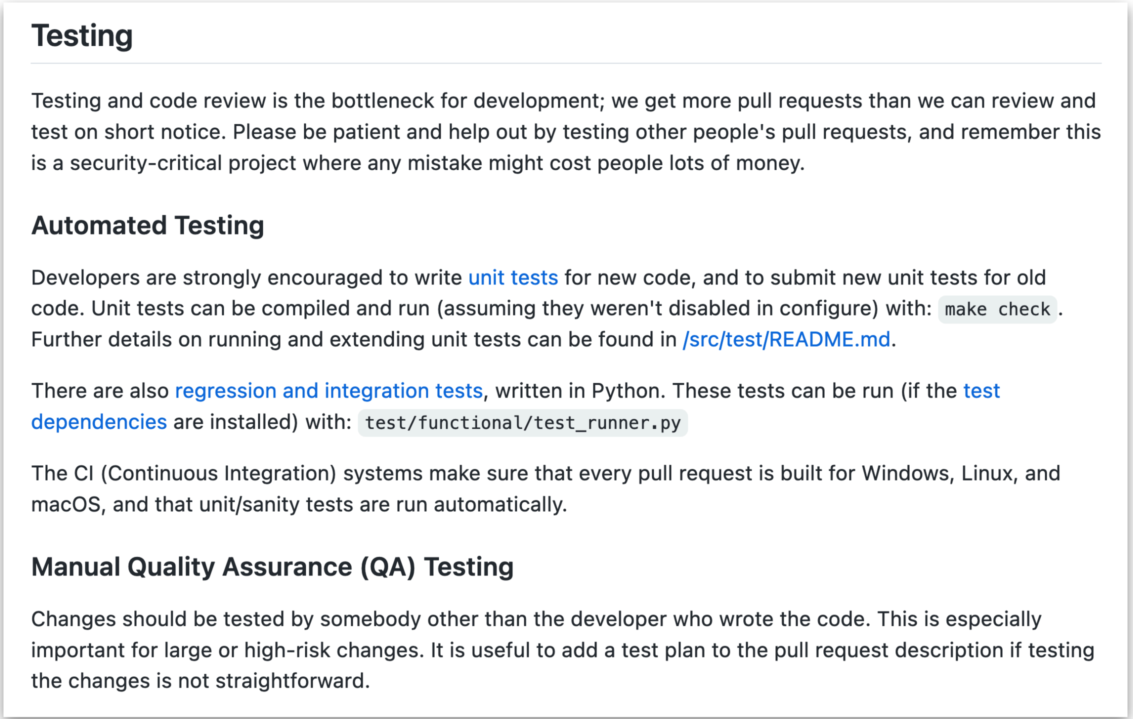
c-plus-plus cryptography bitcoin p2p cryptocurrency

Readme
MIT license
66.4k stars
3.9k watching
33.1k forks

Releases 37

Bitcoin Core 23.0 Latest
on Apr 25

https://github.com/bitcoin/bitcoin



Testing

Testing and code review is the bottleneck for development; we get more pull requests than we can review and test on short notice. Please be patient and help out by testing other people's pull requests, and remember this is a security-critical project where any mistake might cost people lots of money.

Automated Testing

Developers are strongly encouraged to write unit tests for new code, and to submit new unit tests for old code. Unit tests can be compiled and run (assuming they weren't disabled in configure) with: `make check`. Further details on running and extending unit tests can be found in /src/test/README.md.

There are also regression and integration tests, written in Python. These tests can be run (if the test dependencies are installed) with: `test/functional/test_runner.py`

The CI (Continuous Integration) systems make sure that every pull request is built for Windows, Linux, and macOS, and that unit/sanity tests are run automatically.

Manual Quality Assurance (QA) Testing

Changes should be tested by somebody other than the developer who wrote the code. This is especially important for large or high-risk changes. It is useful to add a test plan to the pull request description if testing the changes is not straightforward.

Testability: How difficult is it to test the system?

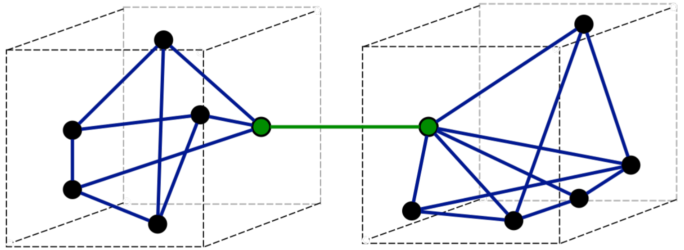
Effort required to provide input to, extract output from, and check the behavior of the system under test.

- **Test efficiency:** effort required to provide input and execute SUT
  - How hard is to setup the SUT? How isolated is it?
  - What inputs are required by the SUT? How hard is to produce them?
- **Test effectiveness:** effort required to collect outputs and check correctness
  - What information do we need to determine pass/fail? (Related to Oracle problem)
  - How hard is it to collect that information?
  - Non-determinism
- **Accidental vs Inherent:** is the code bad or is the problem hard?

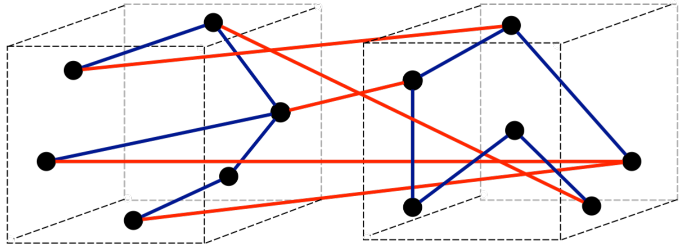
https://robertvbinder.com/software-testability-part-1-what-is-it/
https://arxiv.org/pdf/1801.02201.pdf

Design for Testability: General Principles

Simple, modular, quiet

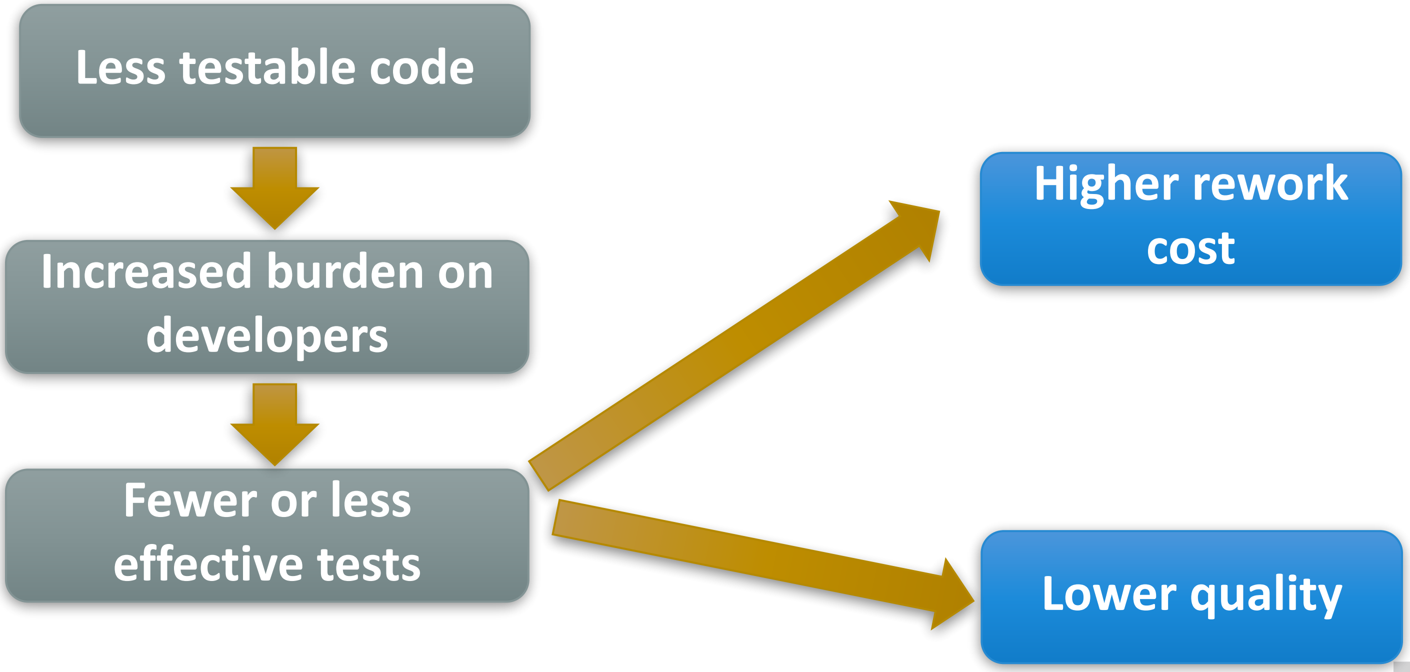


a) Good (loose coupling, high cohesion)



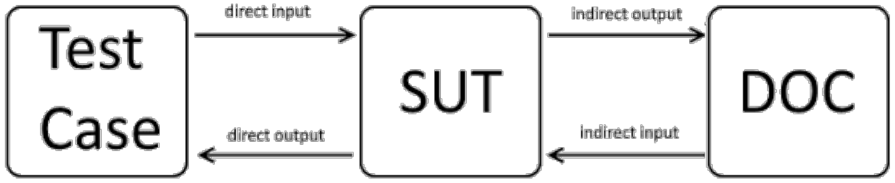
b) Bad (high coupling, low cohesion)

Why should we care about testability?

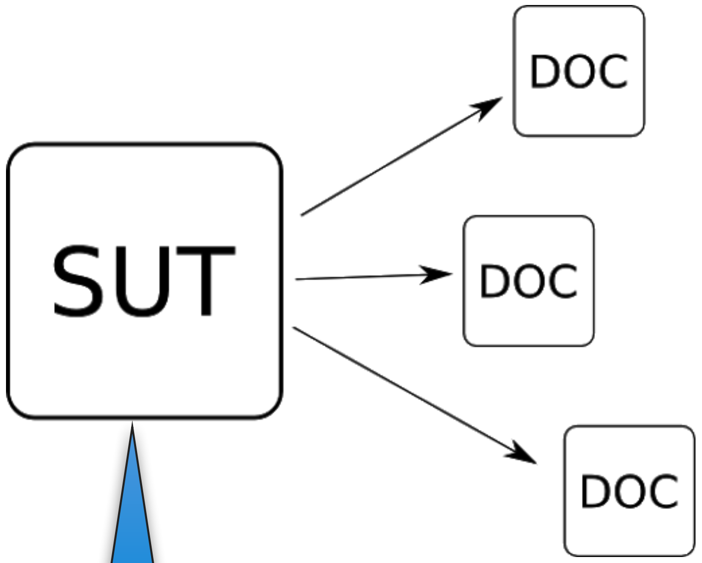


How can we improve testability?

Core Concept: Isolation

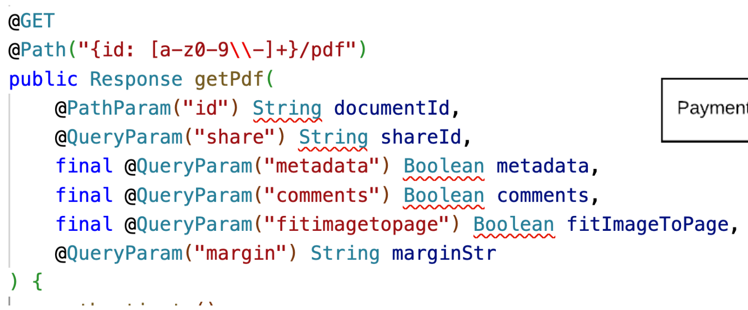


SUT: System under Test
DOC: Depended On Collaborator

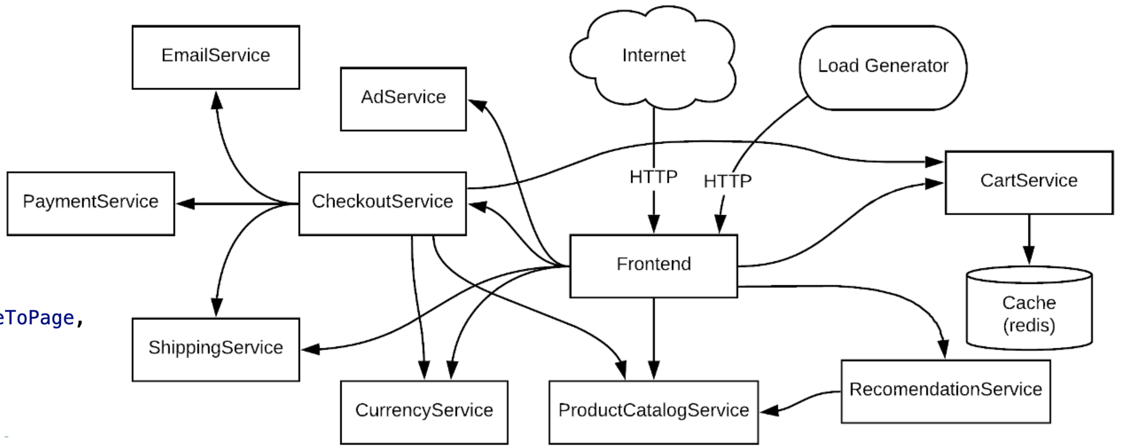


Could be a method, class, or entire service

Collaborators can be classes, services, functions, …



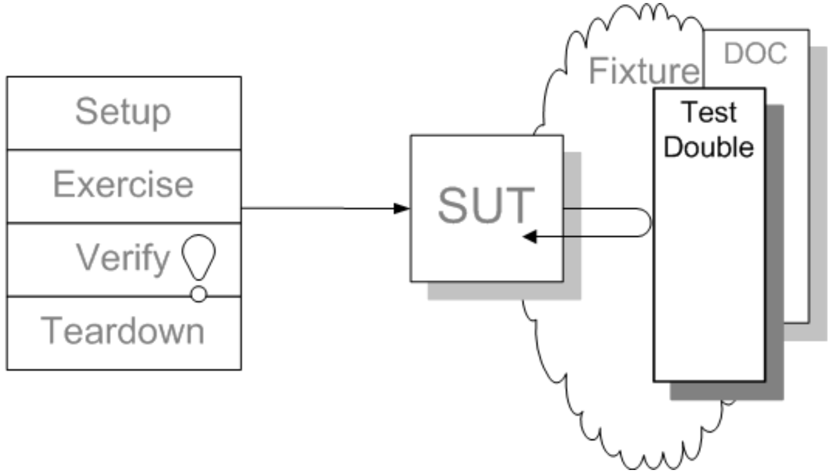
```
@GET
@Path("{id: [a-z0-9\\-]+}/pdf")
public Response getPdf(
    @PathParam("id") String documentId,
    @QueryParam("share") String shareId,
    final @QueryParam("metadata") Boolean metadata,
    final @QueryParam("comments") Boolean comments,
    final @QueryParam("fitimagetopage") Boolean fitImageToPage,
    @QueryParam("margin") String marginStr
) {
```



```
DocumentDao documentDao = new DocumentDao();
```

```
ValidationUtil.validateInteger(marginStr, "margin");
```

Test doubles replace collaborators during testing



Test doubles provide numerous benefits

- **Test services that haven’t been implemented!**
- Isolate the code under test -- easier to find bugs!
- Faster test execution
- Deterministic test outcomes
- Simulate special conditions
- Provide access to hidden information
- ...

https://www.stillbreathing.co.uk/2015/11/15/crash-test-dummies

S3D

A motivating example: An Autonomous Car *

`Car`

https://sf.curbed.com/2020/3/10/21173265/uber-self-driving-cars-return-san-francisco-sf

Test doubles can speed up test execution

- `Route` uses a slow and complex algorithm to find shortest path between two GPS locations.
  - When we aren't testing `Route` itself, we care whether the route is optimal.
- We can use a `Route` double to provide canned directions

https://sf.curbed.com/2020/3/10/21173265/uber-self-driving-cars-return-san-francisco-sf
https://www.autofreaks.com/2014/37923/volvo-reveals-new-3-cylinder-engines/

Test doubles can remove non-determinism

- `Route` relies on real-time information to produce directions
  - E.g., weather, traffic, time of day, etc.
  - This makes `Route` non-deterministic and difficult to test
- Use a `Route` double to return same directions under same conditions

https://sf.curbed.com/2020/3/10/21173265/uber-self-driving-cars-return-san-francisco-sf
https://www.autofreaks.com/2014/37923/volvo-reveals-new-3-cylinder-engines
https://cdn.icon-icons.com/icons2/2106/PNG/512/night_storm_showers_weather_icon_129621.png
https://www.flaticon.com/free-icon/clock_223404

Test doubles can simulate special conditions and inject faults

- `Route` gets its directions from an external service (e.g., Google Maps)
- We want to test how the `Car` behaves when it loses its internet connection

https://sf.curbed.com/2020/3/10/21173265/uber-self-driving-cars-return-san-francisco-sf
https://www.autofreaks.com/2014/37923/volvo-reveals-new-3-cylinder-engines/

Test doubles can expose hidden information

- `Engine` should be started when `Car` is started
  - Engine’s internal state is not accessible to tests
- Use a Engine double to reveal the engine’s simulated state (idle/active)

https://sf.curbed.com/2020/3/10/21173265/uber-self-driving-cars-return-san-francisco-sf
https://www.autofreaks.com/2014/37923/volvo-reveals-new-3-cylinder-engines/

Code-Level vs. Service-Level Doubles

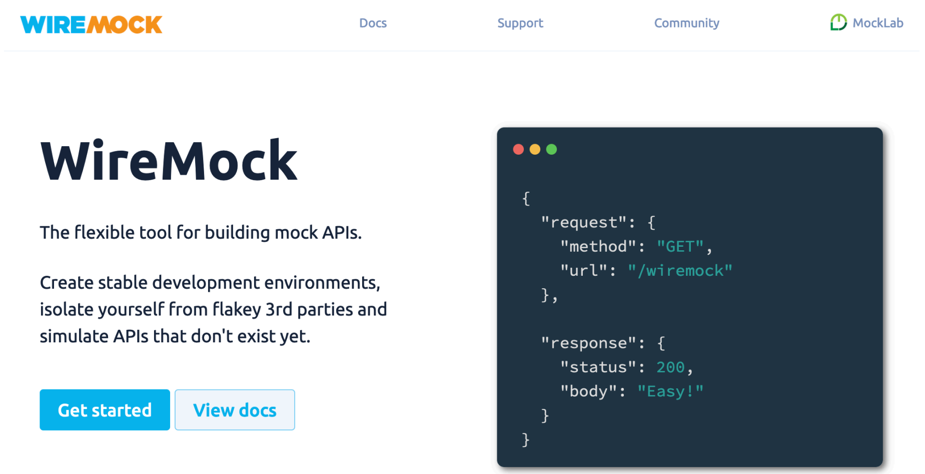
EASYMOCK

mockito

PowerMock

jMock

PACT



WireMock

The flexible tool for building mock APIs.

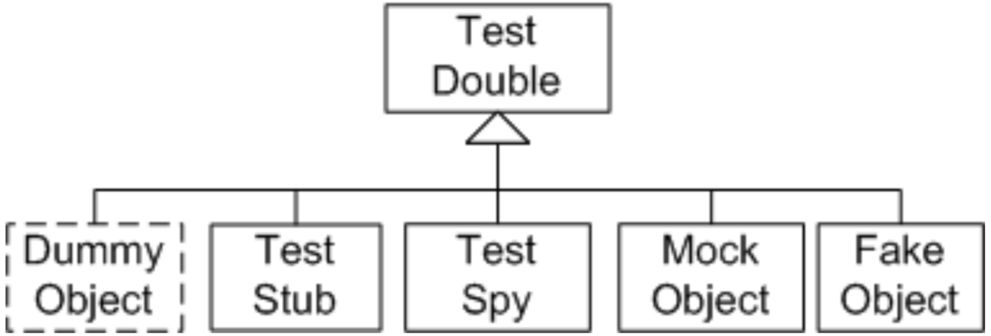
Create stable development environments, isolate yourself from flakey 3rd parties and simulate APIs that don't exist yet.

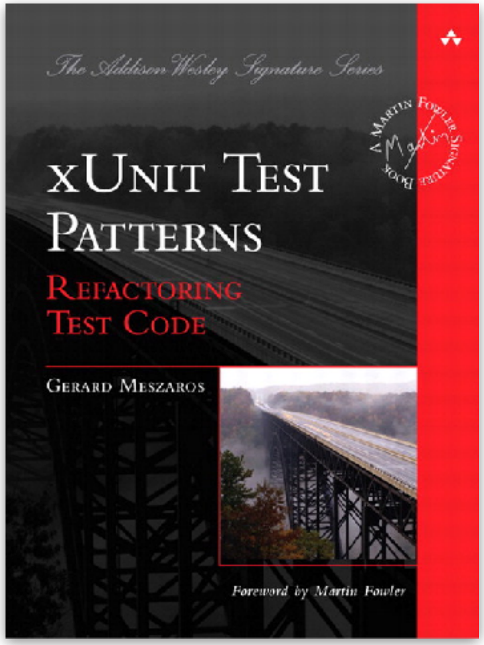
```
{
  "request": {
    "method": "GET",
    "url": "/wiremock"
  },

  "response": {
    "status": 200,
    "body": "Easy!"
  }
}
```

https://wiremock.org

https://pact.io

There are several kinds of test double




The Addison Wesley Signature Series
xUnit Test Patterns
Refactoring Test Code
Gerard Meszaros
A Martin Fowler Signature Book
Foreword by Martin Fowler

See: https://docs.microsoft.com/en-us/archive/msdn-magazine/2007/september/unit-testing-exploring-the-continuum-of-test-doubles
http://xunitpatterns.com/
https://martinfowler.com/articles/mocksArentStubs.html
http://xunitpatterns.com/Test%20Double%20Patterns.html
https://blog.pragmatists.com/test-doubles-fakes-mocks-and-stubs-1a7491dfa3da?gi=b7a3c3a0c968

Test Double: Dummy

Objects that are needed by the program (e.g., parameters) but are never actually used.

```
public interface Logger {
    public void append(String message);
}

public class LoggerDummy implements Logger {
    public void append(String message) {
        // we do nothing!
    }
}
```

Used to improve performance and test isolation, or remove the need for complicated test scaffolding.

Test Double: Stub

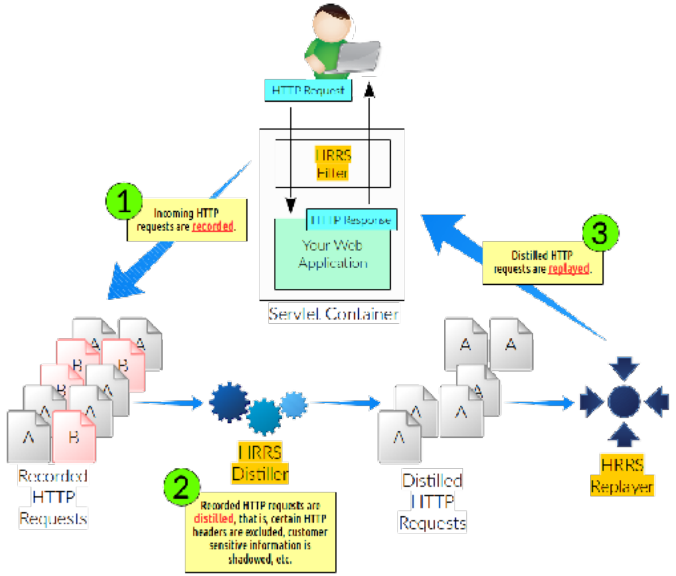
Double for a real collaborator that gives ***predefined*** answers to calls during testing.

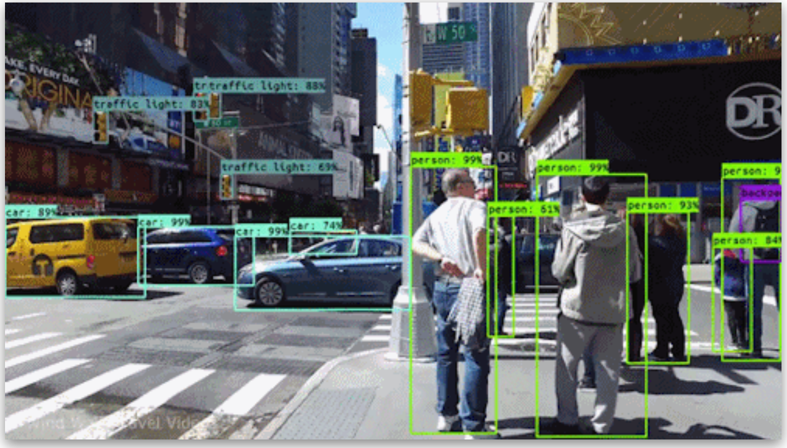
```
// Pass in a stub that was created by a mocking framework.
AccessManager accessManager = new AccessManager(stubAuthenticationService);

// The user shouldn't have access when the authentication service returns false.
when(stubAuthenticationService.isAuthenticated(USER_ID)).thenReturn(false);
assertFalse(accessManager.userHasAccess(USER_ID));

// The user should have access when the authentication service returns true.
when(stubAuthenticationService.isAuthenticated(USER_ID)).thenReturn(true);
assertTrue(accessManager.userHasAccess(USER_ID));
```

Used to improve performance and test isolation, or to test the system under certain conditions (e.g., unauthenticated user, exceptional cases).

https://testing.googleblog.com/2013/07/testing-on-toilet-know-your-test-doubles.html

Special Case: Record and Replay!


HTTP Request
HRRS Filter
HTTP Response
Your Web Application
Servlet Container
1
Incoming HTTP requests are recorded.
3
Distilled HTTP requests are replayed.
Recorded HTTP Requests
HRRS Distiller
2
Recorded HTTP requests are distilled, that is, certain HTTP headers are excluded, customer sensitive information is shadowed, etc.
Distilled HTTP Requests
HRRS Replayer



ROS

github.com/vy/hrrs

https://towardsdatascience.com/everything-you-ever-wanted-to-know-about-computer-vision-heres-a-look-why-it-s-so-awesome-e8a58dfb641e
https://www.ros.org/press-kit/

Test Double: Fake

Provides an optimized, thinned-down version of a collaborator that replicates the same behavior of the original object without certain side effects or consequences.

```
public class FakeProductDatabase implements ProductDatabase {
  private Collection<Product> products = new ArrayList<Product>();

  public void save(Product product) {
    if (findById(product) == null)
      products.add(product);
  }

  public Product findById(long id) {
    for (Product product : products) {
      if (product.getId() == id) return product;
    }
    return null;
  }
}
```

Behaves like a real ProductDatabase that accesses a database, but is simpler, faster, and side-effect free.

Test Double: Spy

Used to track and test the secret internal state of a collaborator. Monitors calls to the collaborator to track the internal state of that collaborator.

```java
public interface RubiksCube {
  public void rotate(...);
}

public class RubiksCubeSolver {
  ...
  public void solve(RubiksCube cube);
}
```

Test Double: Mock

Used to test for **expected interactions** with a collaborator (i.e., method calls). Can behave like a *spy*, a *stub*, or both.

```
// Pass in a mock that was created by a mocking framework.
AccessManager accessManager = new AccessManager(mockAuthenticationService);
accessManager.userHasAccess(USER_ID);

// The test should fail if accessManager.userHasAccess(USER_ID) didn't call
// authenticationService.isAuthenticated(USER_ID) or if it called it more than once.
verify(mockAuthenticationService).isAuthenticated(USER_ID);
```

https://testing.googleblog.com/2013/07/testing-on-toilet-know-your-test-doubles.html

Which test doubles could we use for these collaborators?

Summary

- API documentation is a tool for effective communication and collaboration across different teams
- Testing and documentation, combined, allow teams to develop systems separately without blocking on one another
- There’s a lot of choices when it comes to testing: What’s right for one project might not be a good choice in another. Consider the trade-offs and be wary of dogma and ambiguous language (e.g., testing pyramid).
- Testability drives most of our testing choices. Good systems and code-level design leads to better testability and long-term health.