



Advance Robot Framework





Somkiat Puisungnoen

Somkiat Puisungnoen

Update Info 1 View Activity Log 10+ ...

Timeline About Friends 3,138 Photos More

When did you work at Opendream? X

... 22 Pending Items

Intro

Software Craftsmanship

Software Practitioner at สยามชัมนาณกิจ พ.ศ. 2556

Agile Practitioner and Technical at SPRINT3r

Post Photo/Video Live Video Life Event

What's on your mind?

Public Post

Somkiat Puisungnoen 15 mins · Bangkok · ⚙️

Java and Bigdata



Facebook somkiat.cc

Somkiat | Home | Profile | Chat | News | Help ▾

Page Messages Notifications 3 Insights Publishing Tools Settings Help ▾

somkiat.cc
@somkiat.cc

Home Posts Videos Photos

Like Liked Following Share ...

Help people take action on this Page. ×

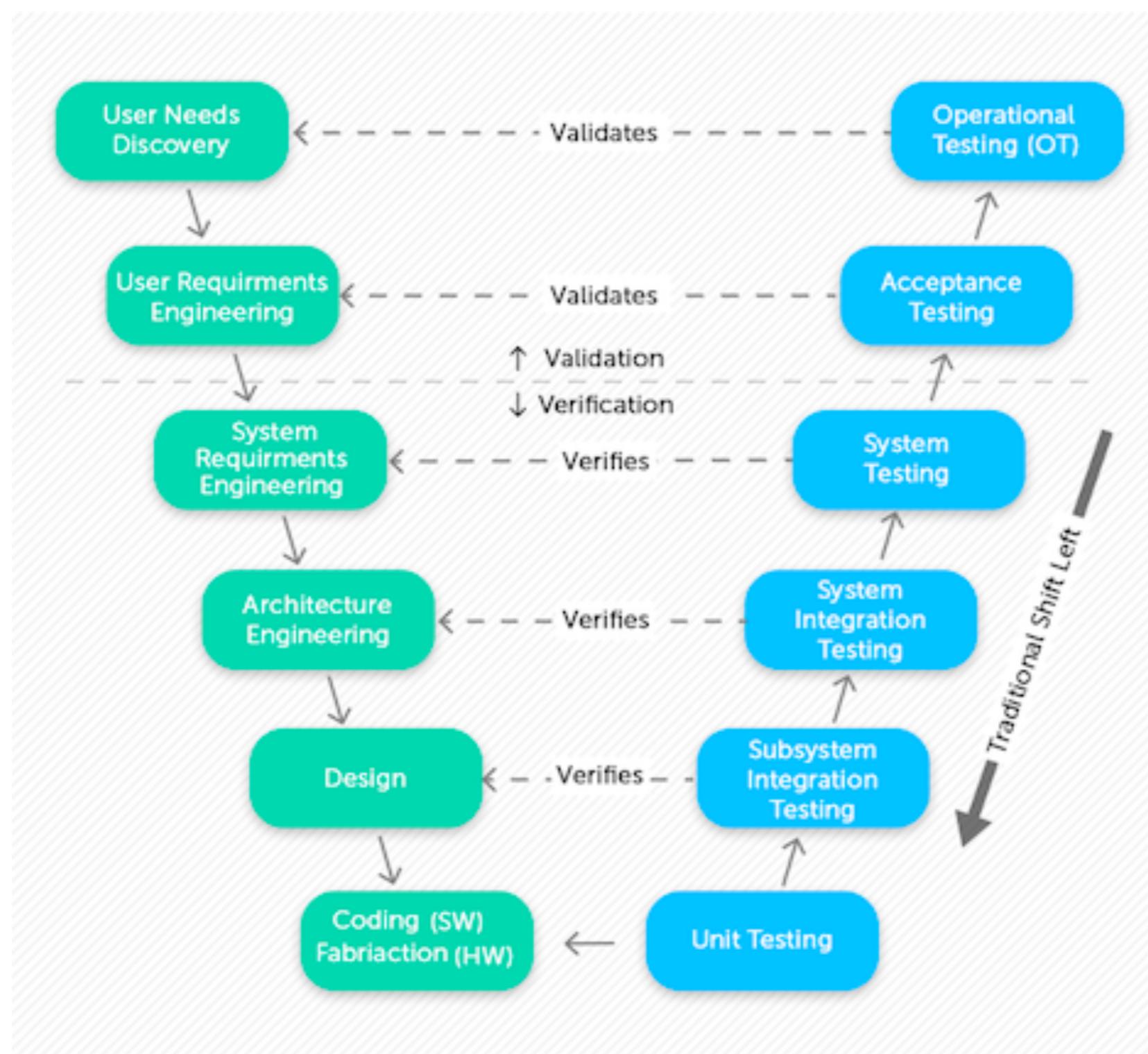
+ Add a Button

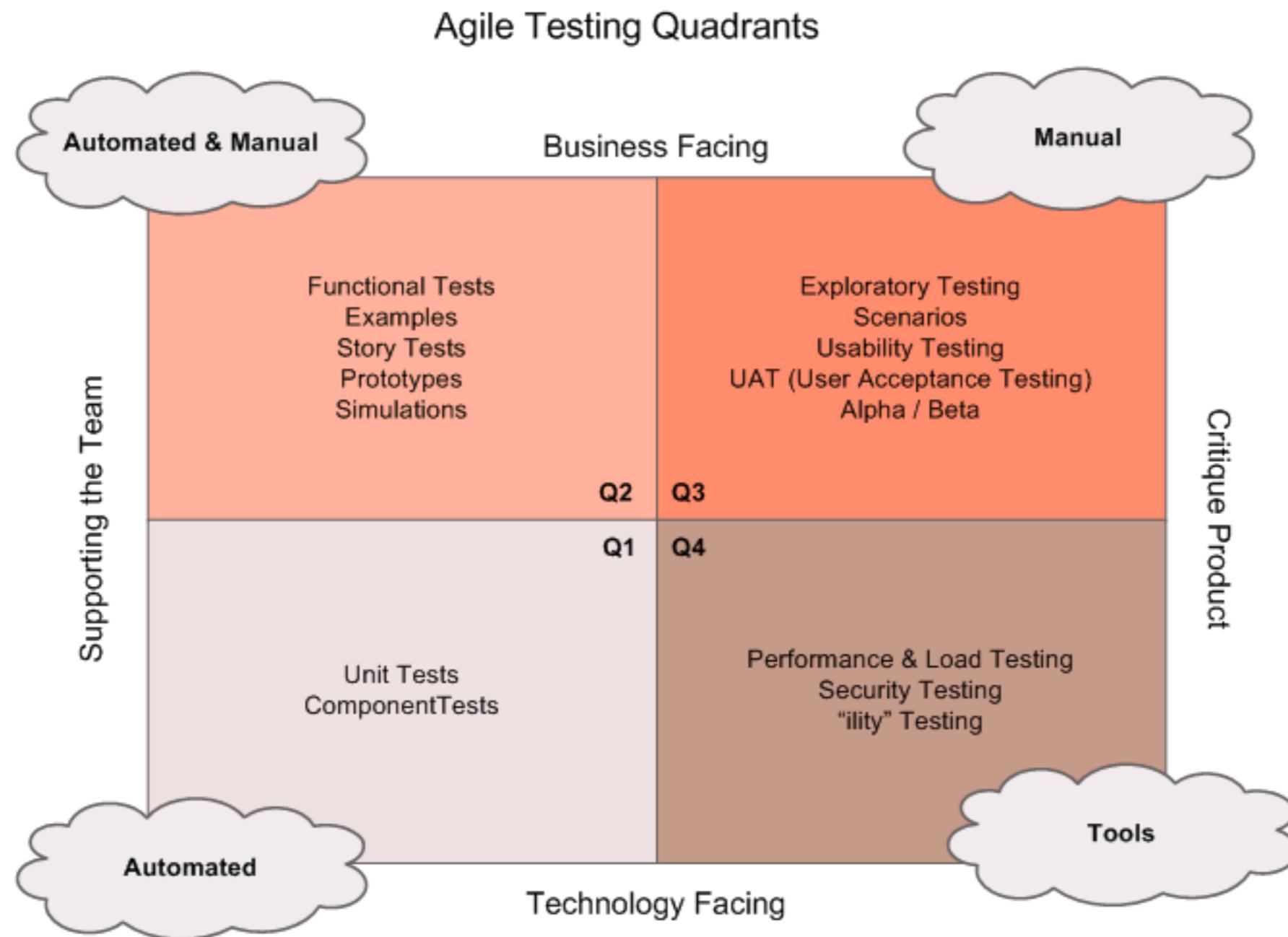


**[https://github.com/up1/course-
advance-robotframework](https://github.com/up1/course-advance-robotframework)**

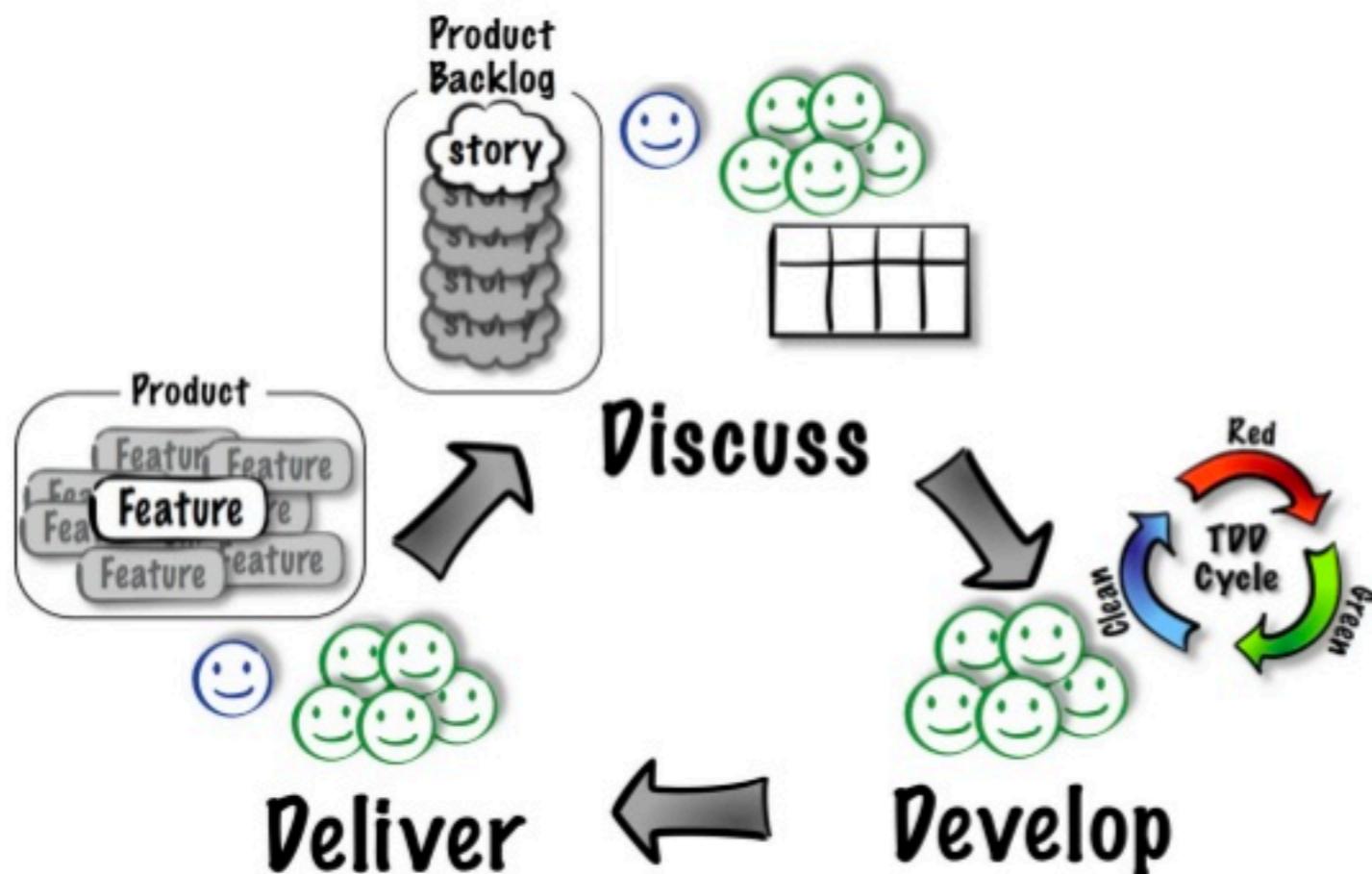








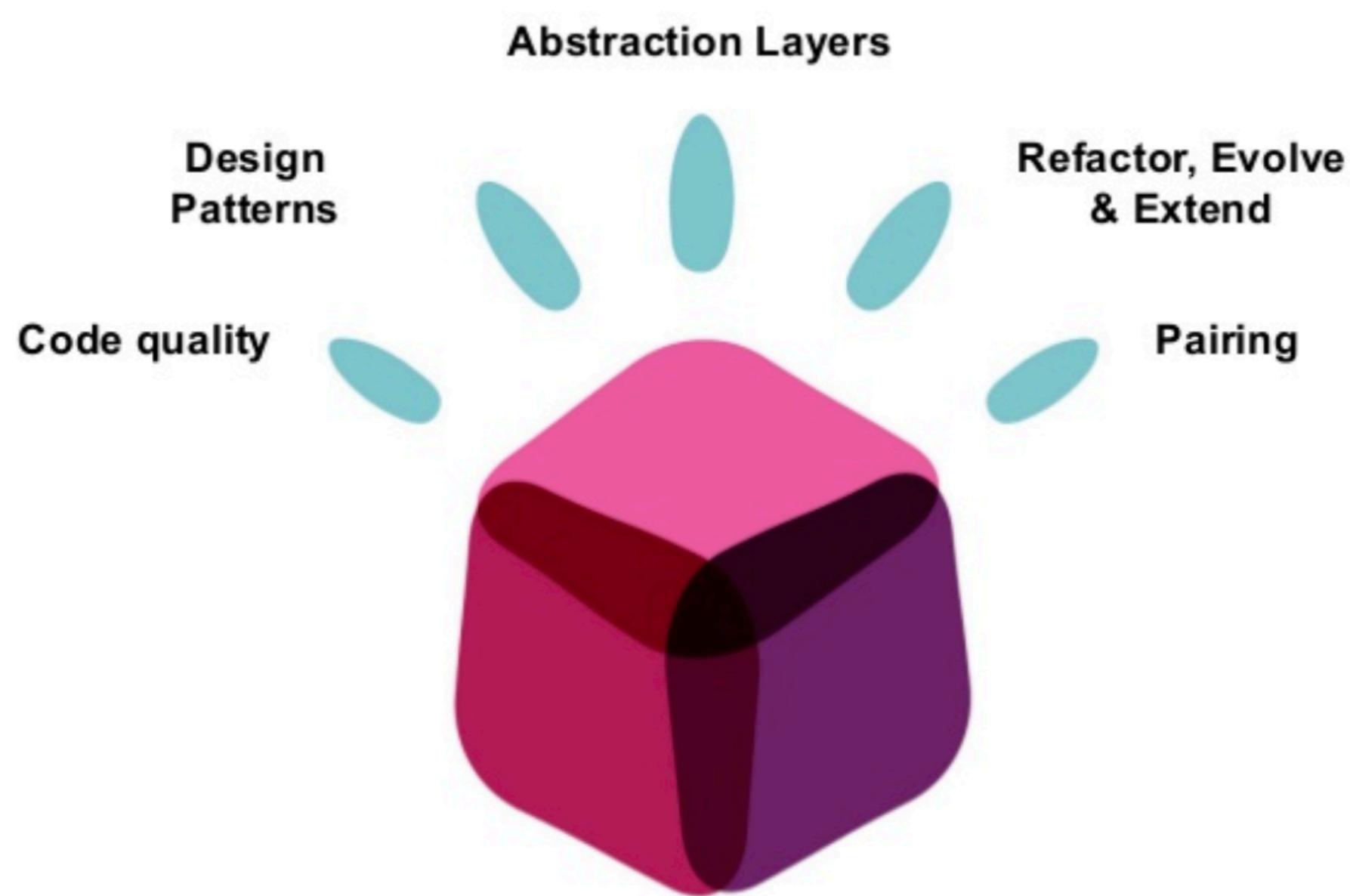
The ATDD cycle



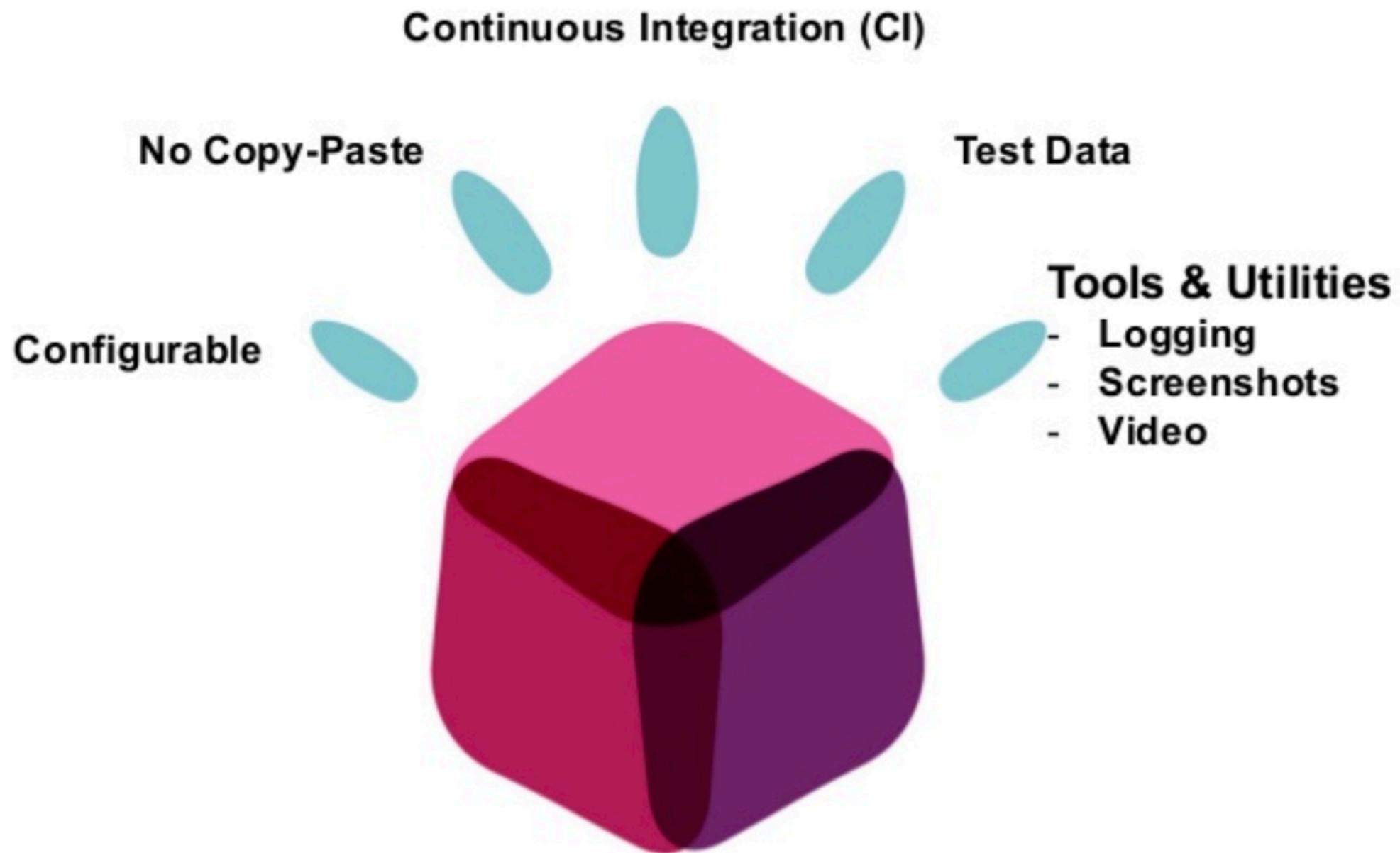
© Image copyright Elisabeth Hendrickson



Test automation framework



Test automation framework



Recap RobotFramework



<https://robotframework.org/>



Recap Robotframework

Workshop with automationpractice.com
Test structure, page object pattern
Robot command-line option



Automation Practice

Your Logo
a new experience

Search 

Cart 1 Product 

WOMEN DRESSES T-SHIRTS

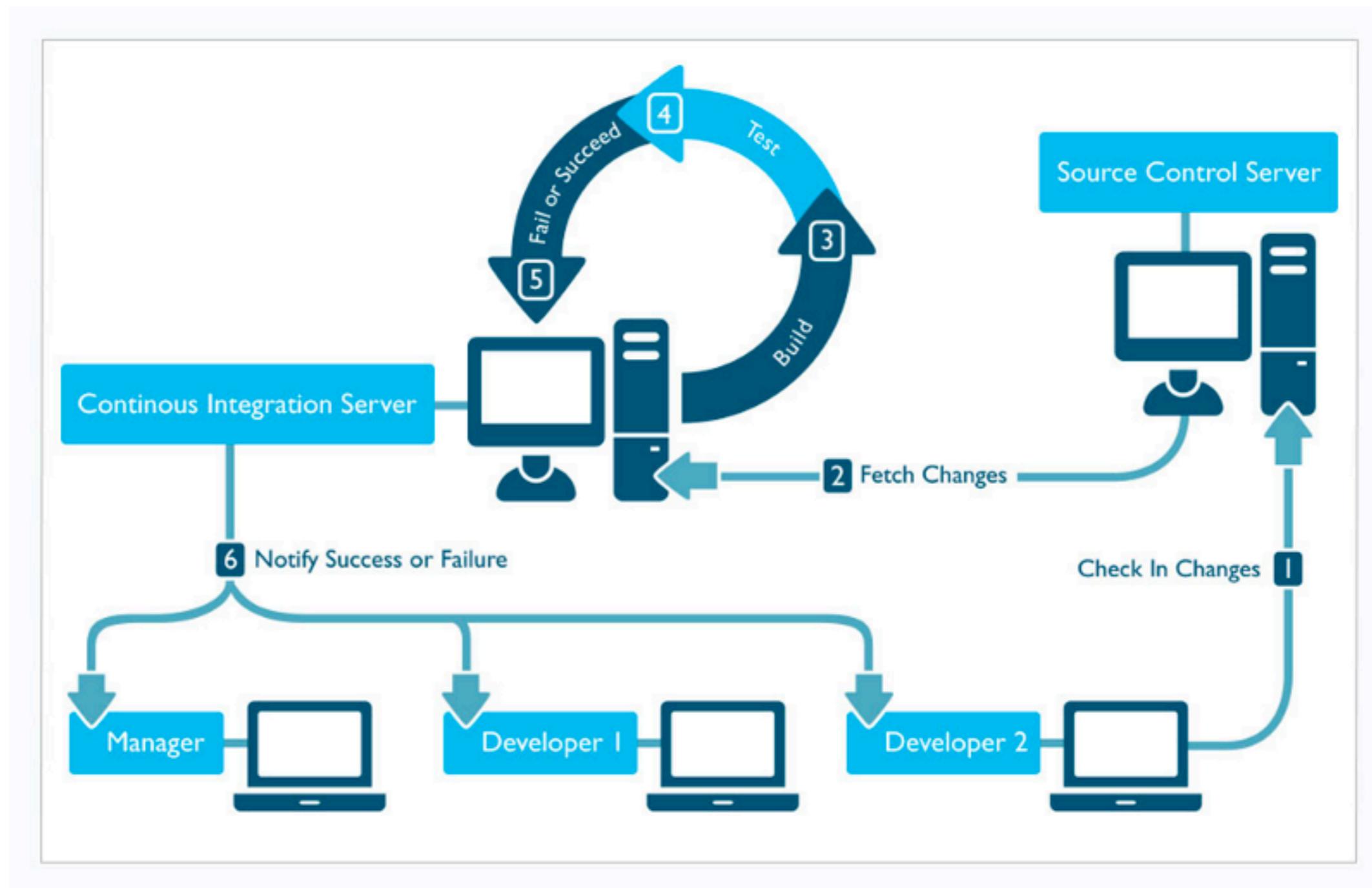




Continuous Integration



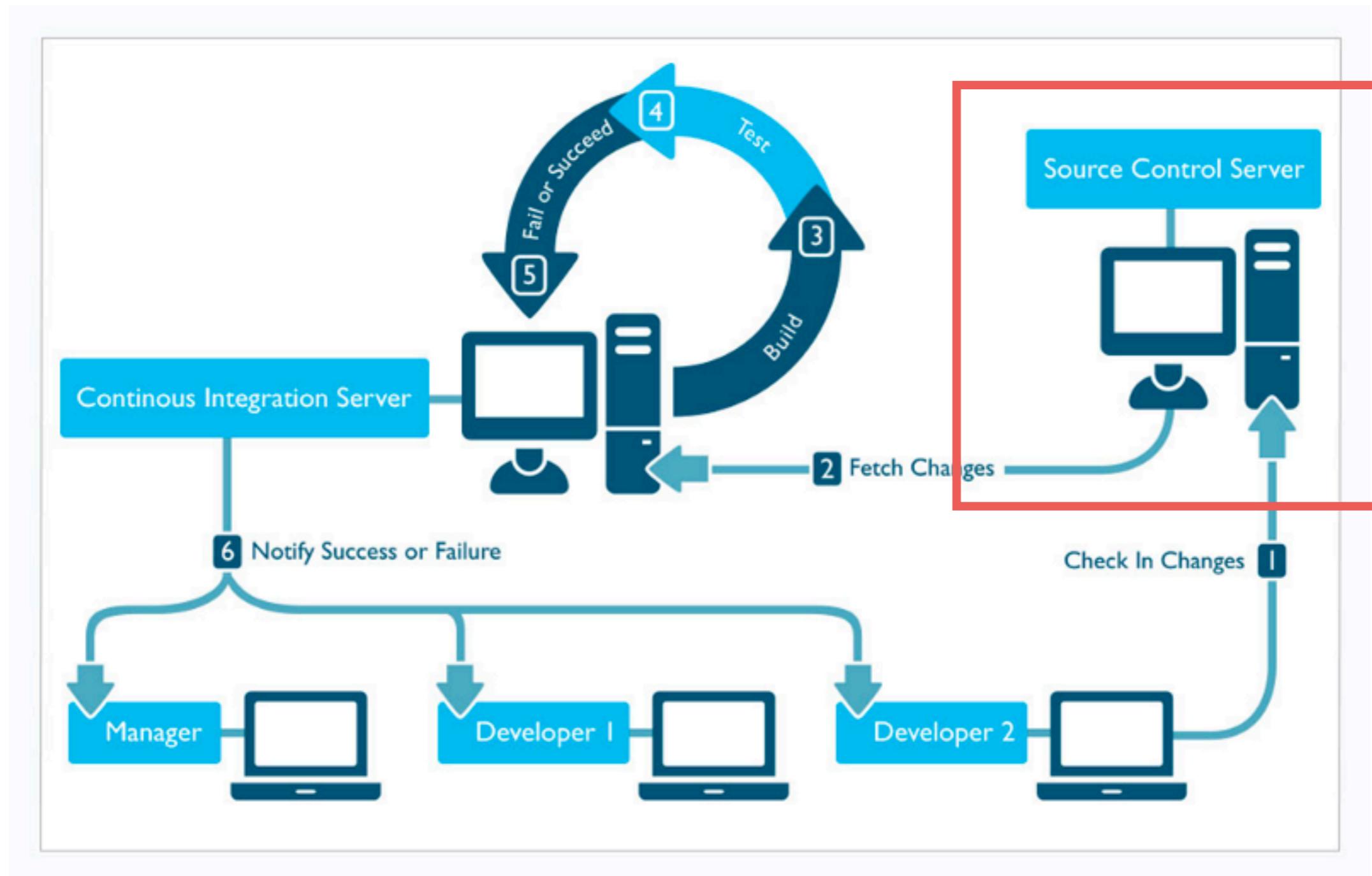
Continuous Integration process



Version Control System



Version Control System



Version Control System

Git
Subversion (svn)
CVS



Git

Download from git-scm.org

The screenshot shows the official Git website at git-scm.com. The header features the Git logo and the tagline "local-branching-on-the-cheap". A search bar is located in the top right corner. The main content area includes a brief introduction to Git's features, a diagram illustrating its distributed nature, and a sidebar with links for About, Documentation, Downloads, and Community.

git --local-branching-on-the-cheap

Search entire site...

Git is a **free and open source** distributed version control system designed to handle everything from small to very large projects with speed and efficiency.

Git is **easy to learn** and has a **tiny footprint with lightning fast performance**. It outclasses SCM tools like Subversion, CVS, Perforce, and ClearCase with features like **cheap local branching**, convenient **staging areas**, and **multiple workflows**.

A 3D wireframe diagram showing seven computer stacks connected by a network of colored lines (red, green, blue) forming a mesh-like structure, representing the distributed nature of Git's architecture.

About
The advantages of Git compared to other source control systems.

Documentation
Command reference pages, Pro Git book content, videos and other material.

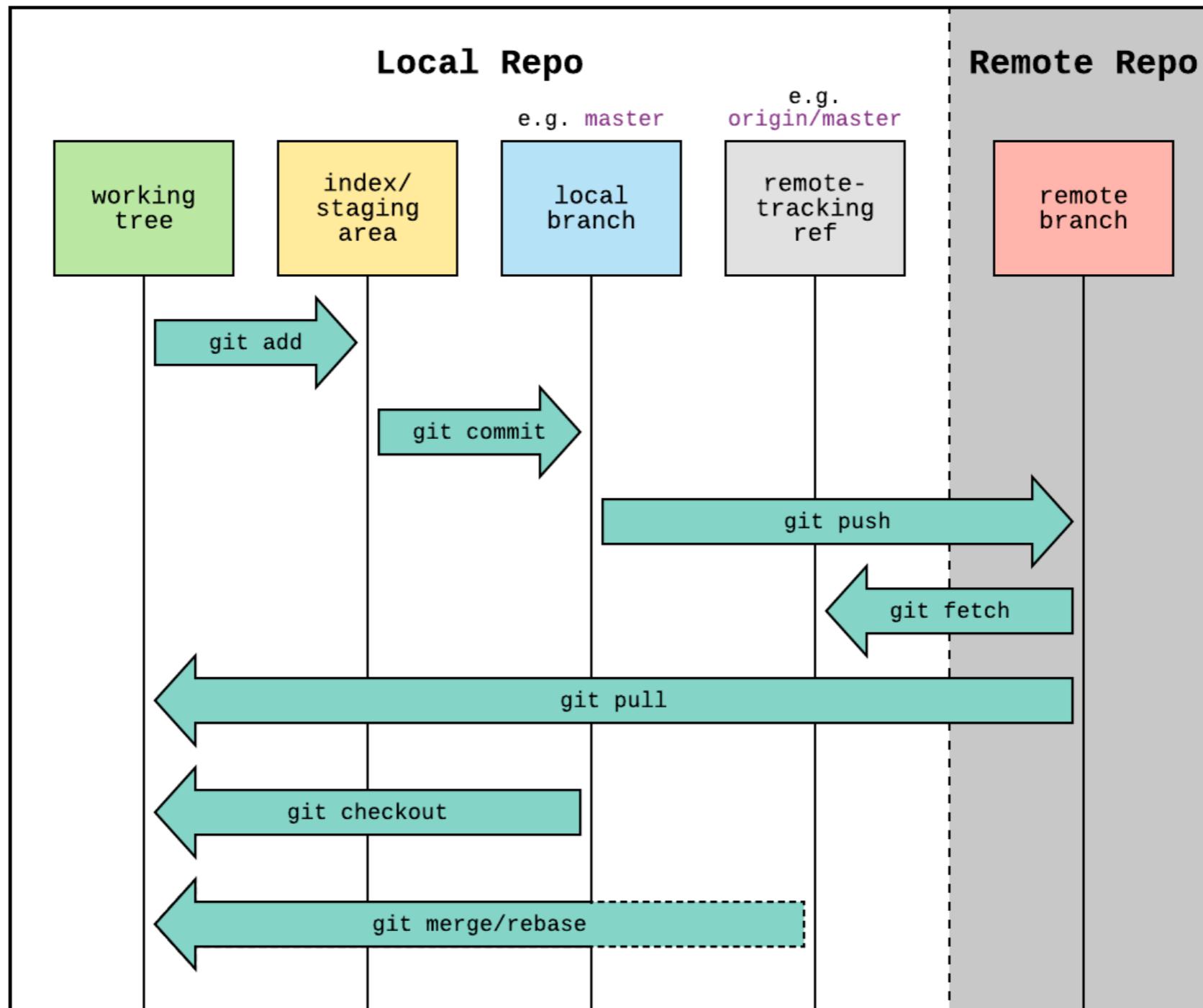
Downloads
GUI clients and binary releases for all major platforms.

Community
Get involved! Bug reporting, mailing list, chat, development and more.

Latest source Release
2.22.0
Release Notes (2019-06-07)
Download 2.22.0 for Mac



Git workflow



<https://github.com/up1/course-advance-robotframework/wiki/Git-101>

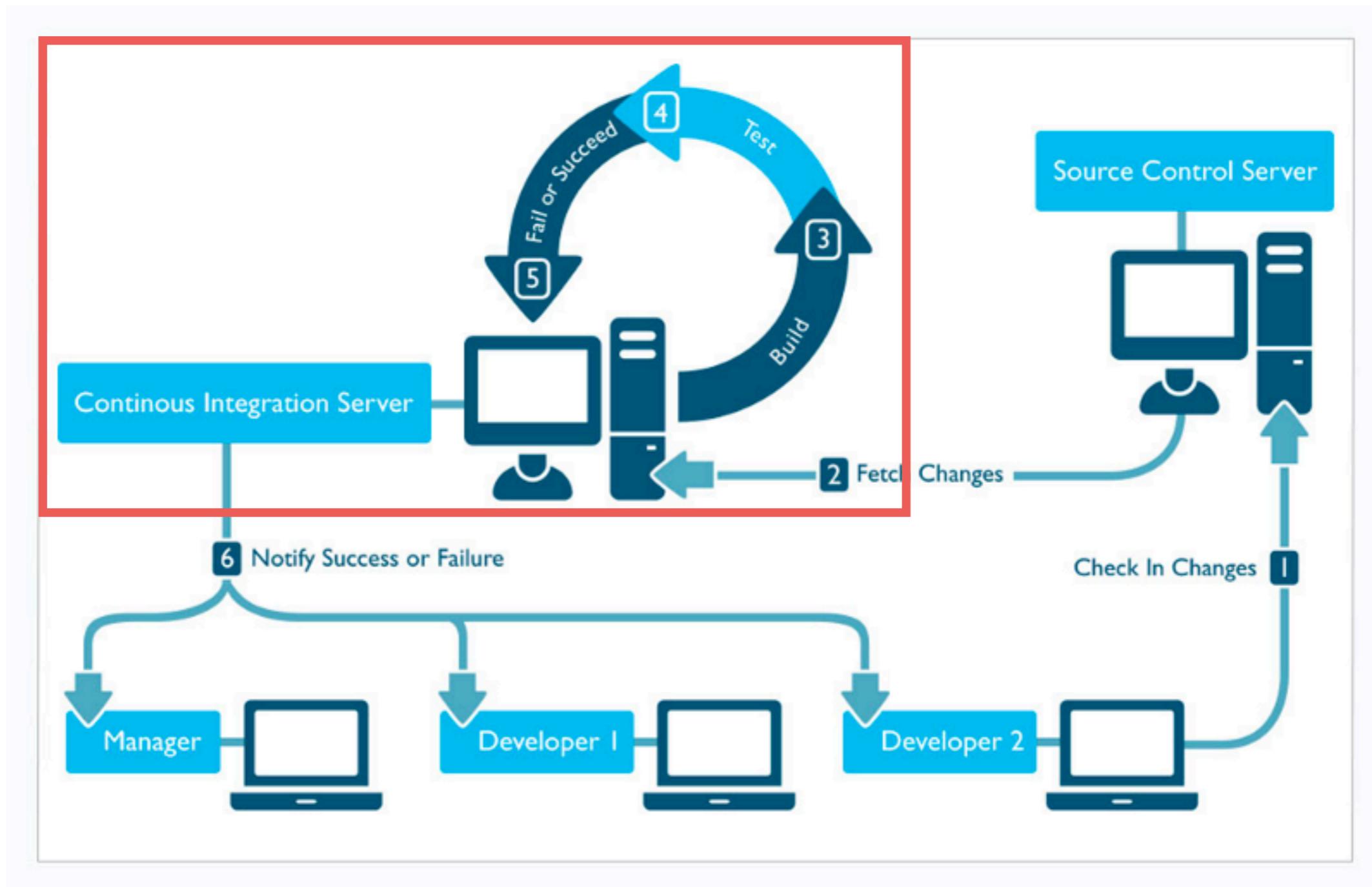


Continuous Testing





Continuous Testing





Jenkins

<https://jenkins.io/>



Start Jenkins

\$java -jar Jenkins.war

Open url=<http://localhost:8080> in browser



Start Jenkins with port

\$java -jar Jenkins.war --httpPort=9090

Open url=<http://localhost:9090> in browser



Create new job with freestyle

Enter an item name

robot

» Required field



Freestyle project

This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.



Pipeline

Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.



External Job

This type of job allows you to record the execution of a process run outside Jenkins, even on a remote machine. This is designed so that you can use Jenkins as a dashboard of your existing automation system.



Source code management

General **Source Code Management** Build Triggers Build Environment Build Post-build Actions

Source Code Management

None
 CVS
 CVS Projectset
 Git

Repositories

Repository URL	<input type="text" value="https://github.com/up1/robot-swpark.git"/>	?
Credentials	<input type="button" value="- none -"/> <input type="button" value="Add"/>	Advanced...
Add Repository		

Branches to build

Branch Specifier (blank for 'any')	<input type="text" value="*/master"/>	X	?
Add Branch			



Build trigger (every minute)

Build Triggers

- Trigger builds remotely (e.g., from scripts) ?
- Build after other projects are built ?
- Build periodically ?
- GitHub hook trigger for GITScm polling ?
- Poll SCM ?

Schedule

Do you really mean "every minute" when you say "*****"? Perhaps you meant "H * * * *" to poll once per hour

Would last have run at Friday, July 19, 2019 12:00:38 AM ICT; would next run at Friday, July 19, 2019 12:00:38 AM ICT.

Ignore post-commit hooks



Build => run with robot

Build

Execute shell

Command `export PATH=.:~/Users/somkiat/data/slide/robotframework/swpark-20190718/workshop:$PATH
robot *.robot`

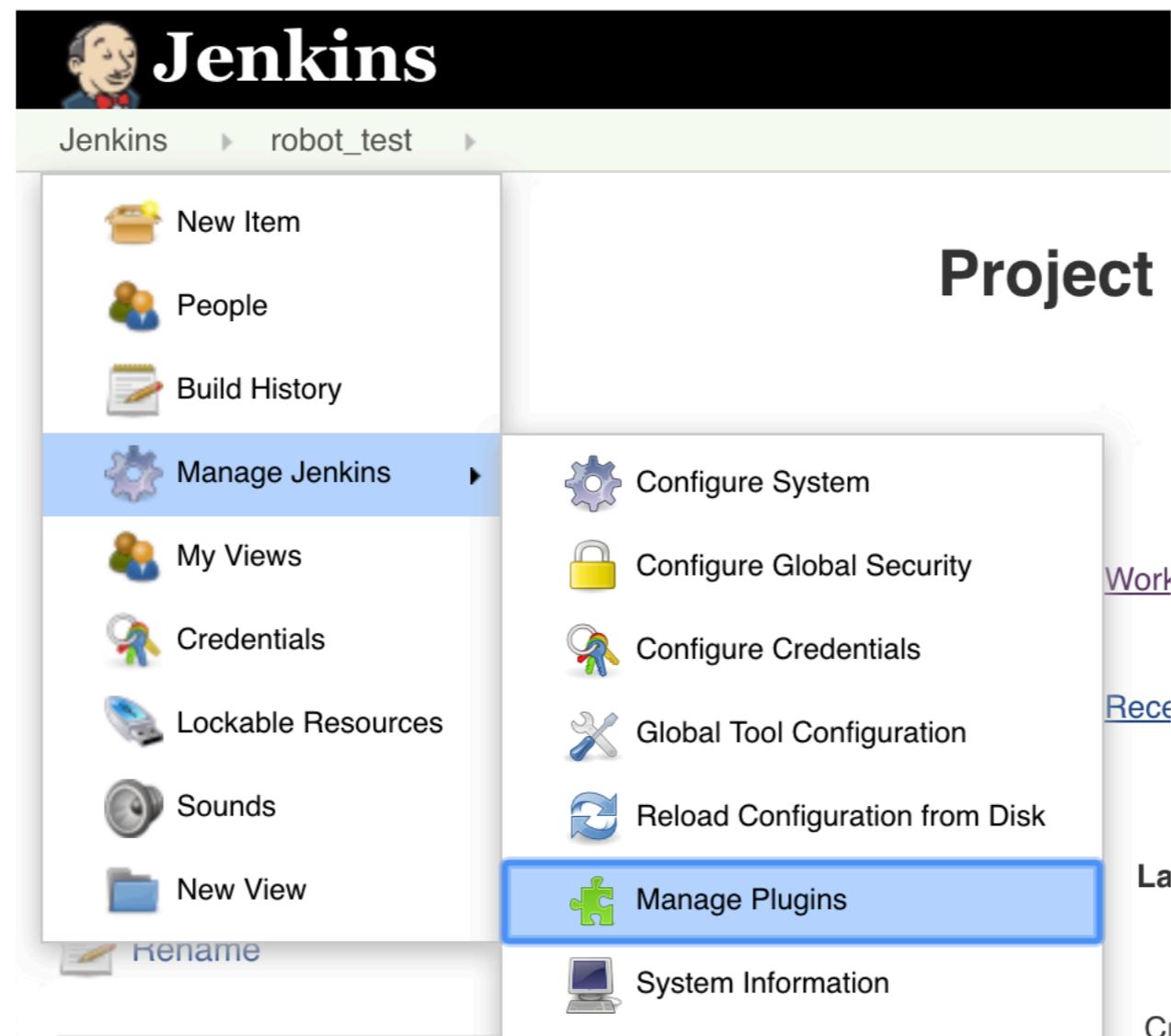
See [the list of available environment variables](#)

[Advanced...](#)

Add build step ▾



Install RobotFramework report plugin



Install RobotFramework report plugin

Filter:

Updates Available Installed Advanced

Install ↓	Name	Version
<input checked="" type="checkbox"/> Robot Framework This publisher stores Robot Framework test reports for builds and shows summaries of them in project and build views along with trend graph.		1.6.5

[Install without restart](#) [Download now and install after restart](#) Update information obtained: 8 hr 22 min ago [Check now](#)



Install RobotFramework report plugin

Installing Plugins/Upgrades

Preparation

- Checking internet connectivity
- Checking update center connectivity
- Success

Robot Framework



Success

→ [Go back to the top page](#)

(you can start using the installed plugins right away)

→ Restart Jenkins when installation is complete and no jobs are running



Post Build with RobotFramework report

The screenshot shows the Jenkins post-build actions configuration. A red box highlights the 'Publish Robot Framework test results' option, which is highlighted with a blue background. Below it is a list of other actions:

- Aggregate downstream test results
- Archive the artifacts
- Build other projects
- Publish JUnit test result report
- Publish Robot Framework test results** (highlighted)
- Record fingerprints of files to track usage
- Git Publisher
- E-mail Notification
- Editable Email Notification
- Jenkins Sounds
- Set GitHub commit status (universal)
- Set build status on GitHub commit [deprecated]
- Delete workspace when build is done

Add post-build action ▾



Post Build with RobotFramework report

Post-build Actions

Publish Robot Framework test results

Directory of Robot output

Path to directory containing robot xml and html files (relative to build workspace)

Advanced... 

Thresholds for build result

% 100.0 

% 100.0 

Use thresholds for critical tests only



Run and see report

Project robot_test

 [Workspace](#)

 [Recent Changes](#)

 **Latest Robot Results:**

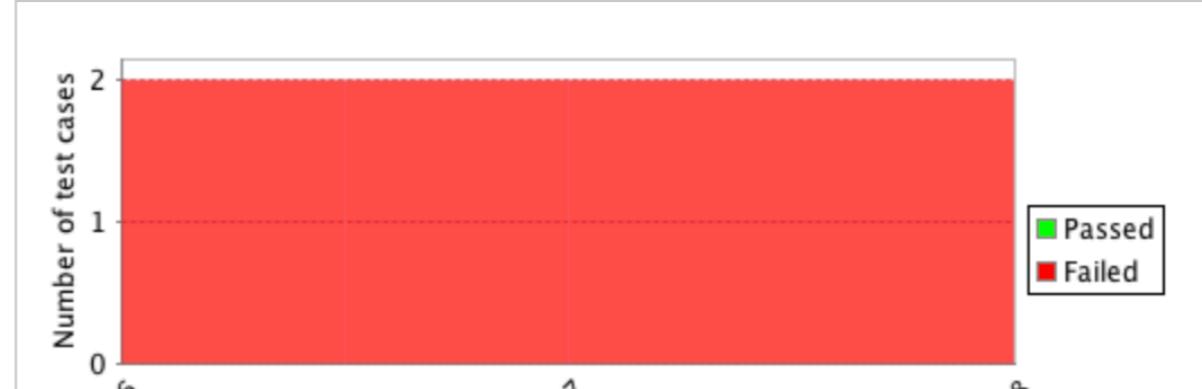
	Total	Failed	Passed	Pass %
Critical tests	2	2	0	0.0
All tests	2	2	0	0.0

- [Browse results](#)
- [Open report.html](#)
- [Open log.html](#)

 [add description](#)

[Disable Project](#)

Robot Framework Tests Trend (all tests)



The chart shows the number of test cases (Y-axis, 0 to 2) versus build numbers (X-axis, #6, #7, #8). All three builds show 2 failed test cases each. A legend indicates green for Passed and red for Failed.

Number of test cases

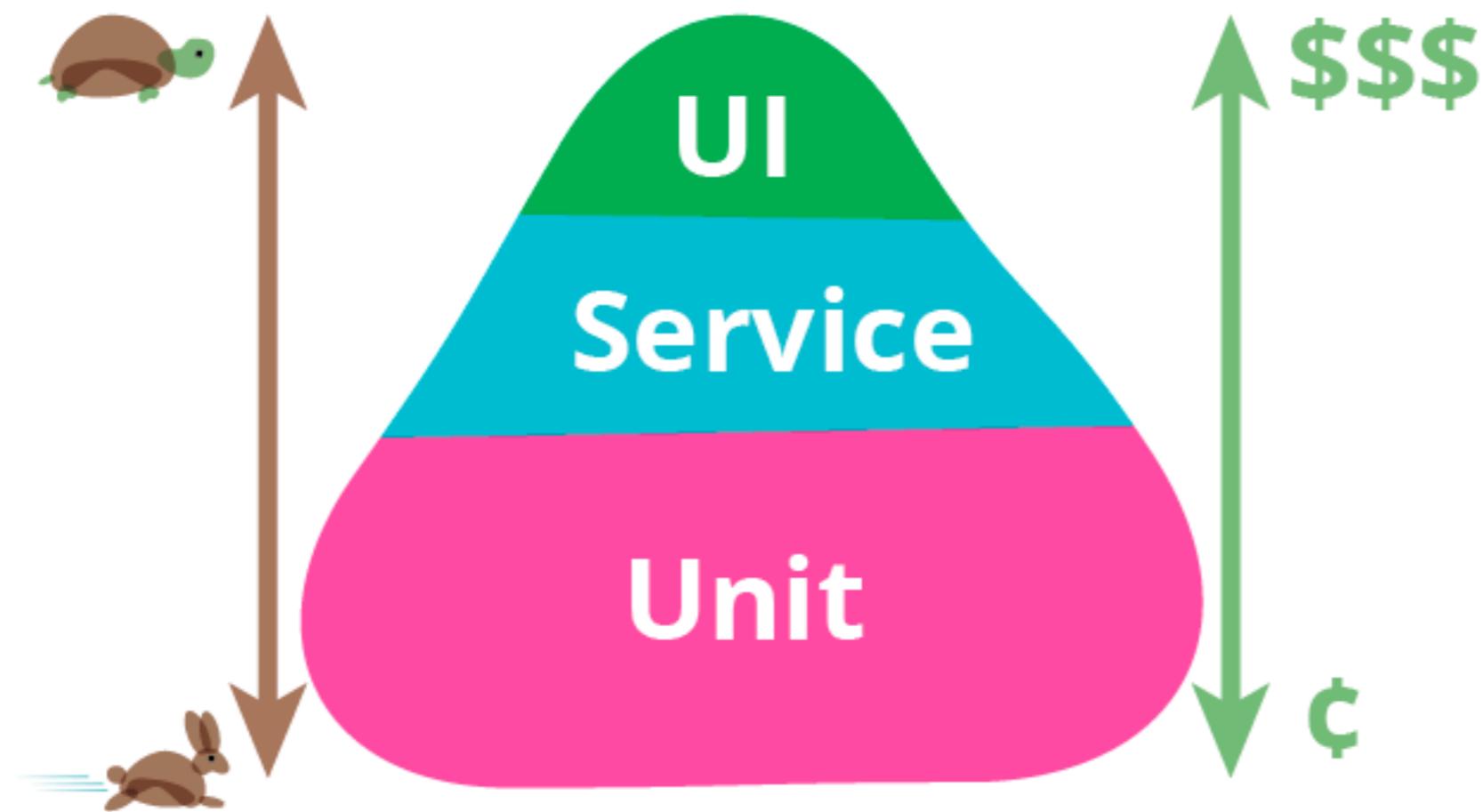
Build number

Zoom to changes Show only failed Show only critical all Max builds [Show bigger image](#)

Permalinks

- [Last build \(#8\), 7 hr 12 min ago](#)
- [Last stable build \(#2\), 7 hr 51 min ago](#)
- [Last successful build \(#2\), 7 hr 51 min ago](#)
- [Last failed build \(#8\), 7 hr 12 min ago](#)
- [Last unsuccessful build \(#8\), 7 hr 12 min ago](#)
- [Last completed build \(#8\), 7 hr 12 min ago](#)





Scaling Testing



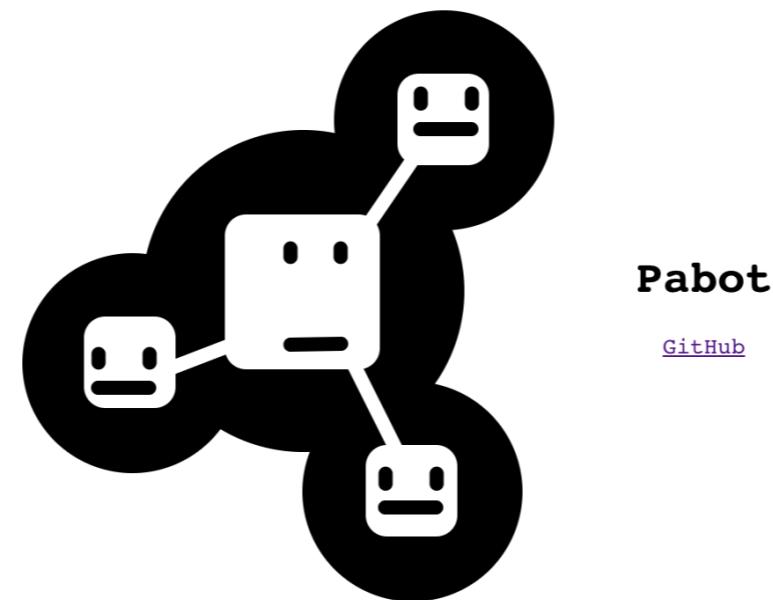
Scaling Testing

- Pabot
- Selenium Grid

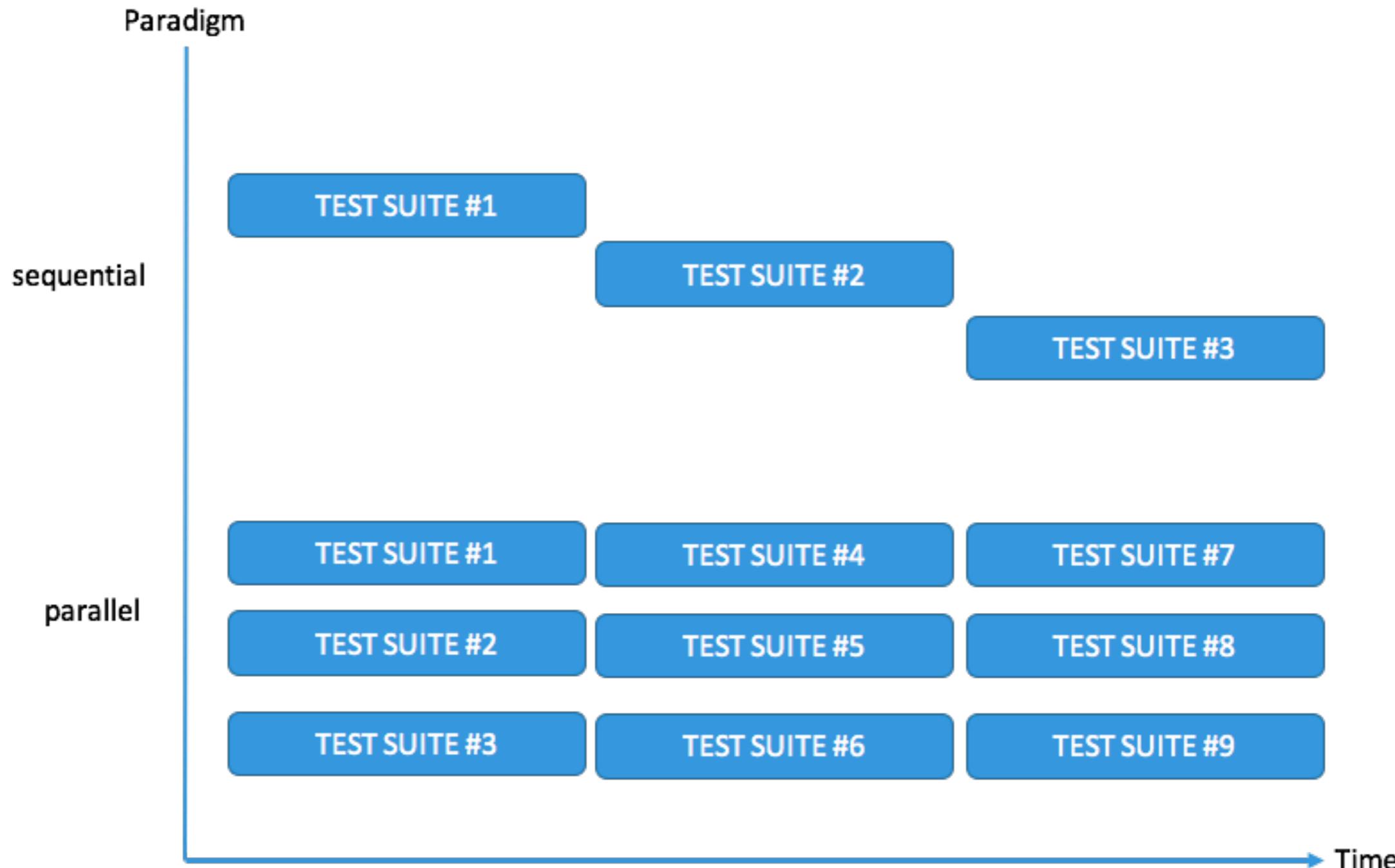


Pabot

Parallel executor for Robot Framework
Split one execution into multiple
<https://pabot.org/>



Test execution



Using Pabot

```
$pip install -U robotframework-pabot  
$pabot
```



Parallel test suites

\$pabot flow_dress_sorting.robot



Parallel test cases

```
$pabot --testlevelspli  
flow_dress_sorting.robot
```



Workshop with Pobot



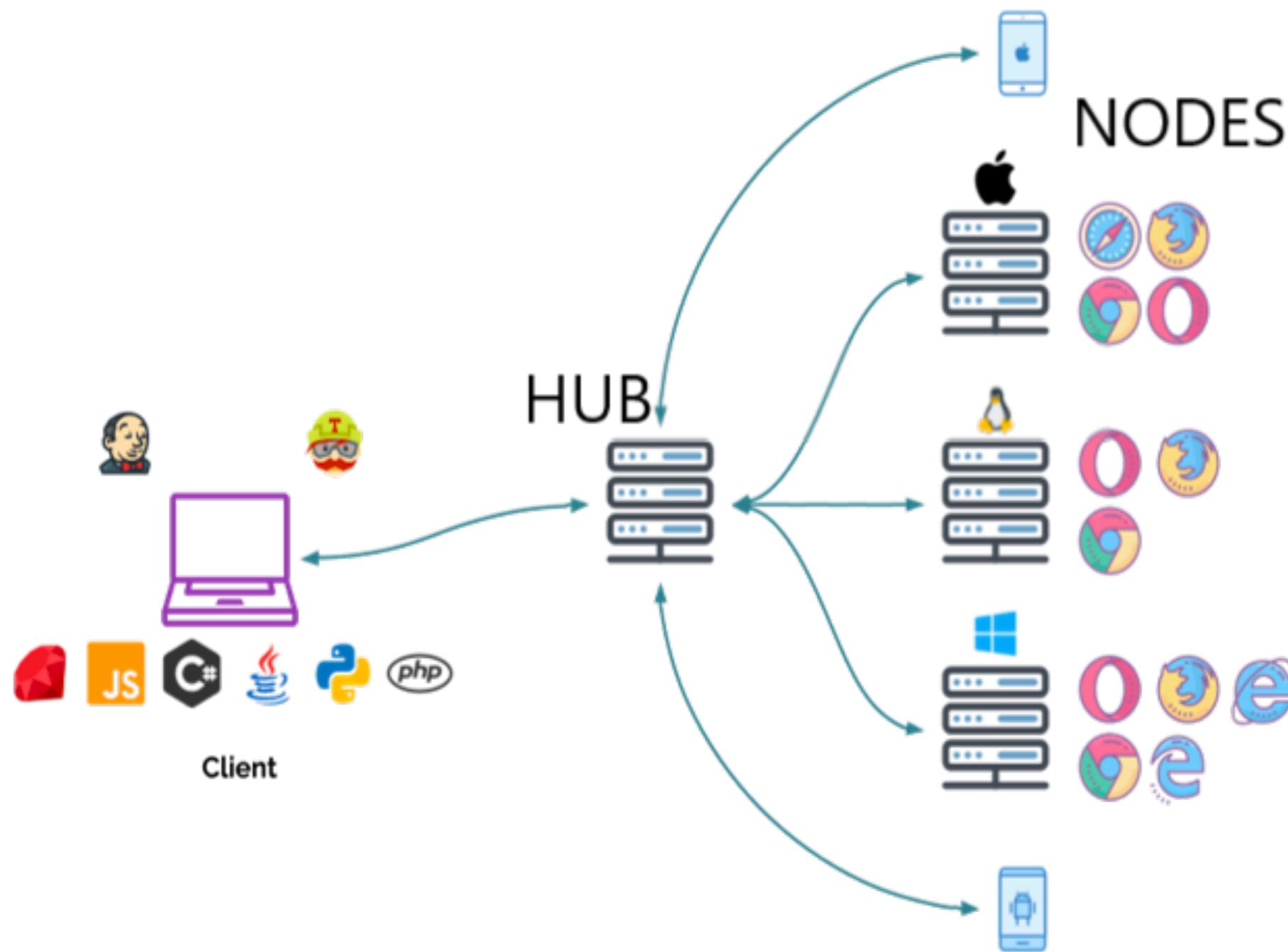
Selenium grid

Run test cases on different machines
Parallel testing

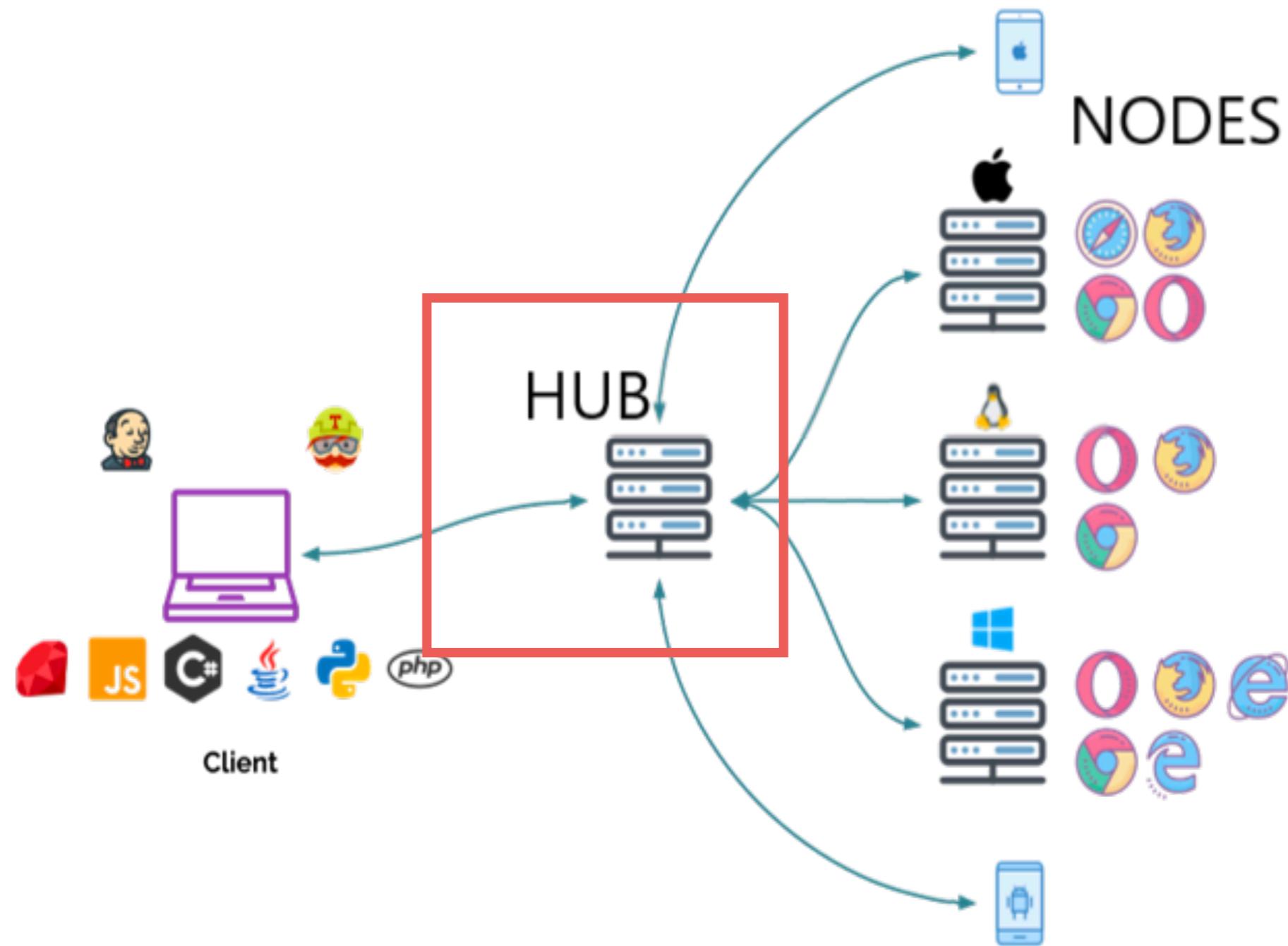
<https://github.com/SeleniumHQ/selenium/wiki/Grid2>



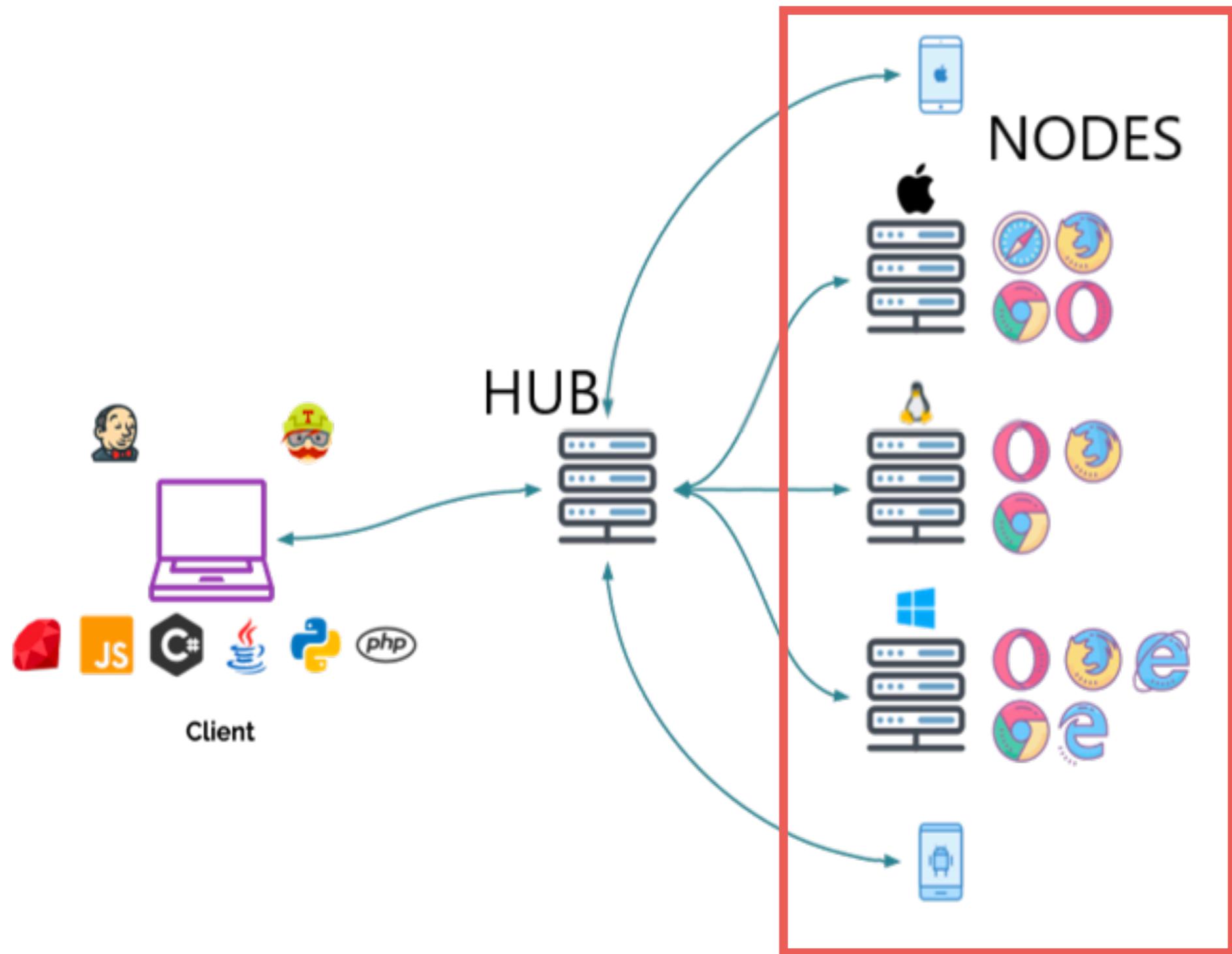
Selenium grid architecture



Selenium grid architecture



Selenium grid architecture



Selenium grid architecture

Hub
Nodes



Download selenium grid

← → C ⓘ Not Secure | selenium-release.storage.googleapis.com/index.html

Index of /

Name	Last modified	Size	ETag
 2.39	-	-	-
 2.40	-	-	-
 2.41	-	-	-
 2.42	-	-	-
 2.43	-	-	-
 2.44	-	-	-
 2.45	-	-	-
 2.46	-	-	-
 2.47	-	-	-
 2.48	-	-	-
 2.49	-	-	-

<http://selenium-release.storage.googleapis.com/index.html>



Download selenium grid 4.0

← → ⌂ ⓘ Not Secure | selenium-release.storage.googleapis.com/index.html?path=4.0/

Index of /4.0/

	<u>Name</u>	Last modified	Size	ETag
	Parent Directory		-	
	selenium-html-runner-4.0.0-alpha-1.jar	2019-04-24 16:17:02	13.52MB	2eca35318710f46d1ba5ed5543a906c9
	selenium-html-runner-4.0.0-alpha-2.jar	2019-07-01 21:32:41	13.76MB	346d72e4f425bfec91c7073a46c96208
	selenium-java-4.0.0-alpha-1.zip	2019-04-24 16:17:01	8.46MB	db9ed262a07c1cd2bb6098263c7f1e7b
	selenium-java-4.0.0-alpha-2.zip	2019-07-01 21:32:33	8.74MB	2d31929580c3d829197eea97ade5f4f0
	selenium-server-4.0.0-alpha-1.jar	2019-04-24 16:16:58	10.62MB	c32b1dd1c12cdb42b48f345d65d657fb
	selenium-server-4.0.0-alpha-1.zip	2019-04-24 16:16:59	10.20MB	7f0bc4bb4fc2a5a7f0a262f62bf782d3
	selenium-server-4.0.0-alpha-2.jar	2019-07-01 21:32:04	10.79MB	d0676e6b3ee508b48416aba603662573
	selenium-server-4.0.0-alpha-2.zip	2019-07-01 21:32:11	10.47MB	fb19d62db44a7b163f1fbc2fff9dff0a
	selenium-server-standalone-4.0.0-alpha-1.jar	2019-04-24 16:17:00	11.98MB	ac553ec987d16d2af8c8e3ef9061772c
	selenium-server-standalone-4.0.0-alpha-1.zip	2019-04-24 16:17:00	11.77MB	1974b11f970bad6e15c84e3840ec3897
	selenium-server-standalone-4.0.0-alpha-2.jar	2019-07-01 21:32:19	12.33MB	d000d97d24389fde5bfb94f450ede780
	selenium-server-standalone-4.0.0-alpha-2.zip	2019-07-01 21:32:27	12.26MB	2466773c71eeddea02004371a5e32324

<http://selenium-release.storage.googleapis.com/index.html>



Start Hub

```
$java -jar selenium-server-standalone-  
<version>.jar -role hub
```



Start Hub

Open url=http://localhost:4444



Whoops! The URL specified routes to this help page.

For more information about Selenium Grid Hub please see the [docs](#) and/or visit the [wiki](#). Or perhaps you are looking for the Selenium Grid Hub [console](#).

Happy Testing!

Selenium is made possible through the efforts of our open source community, contributions from these [people](#), and our [sponsors](#).



Start Node

```
$java -jar selenium-server-standalone-  
      <version>.jar -role node  
-hub http://localhost:4444/grid/register
```



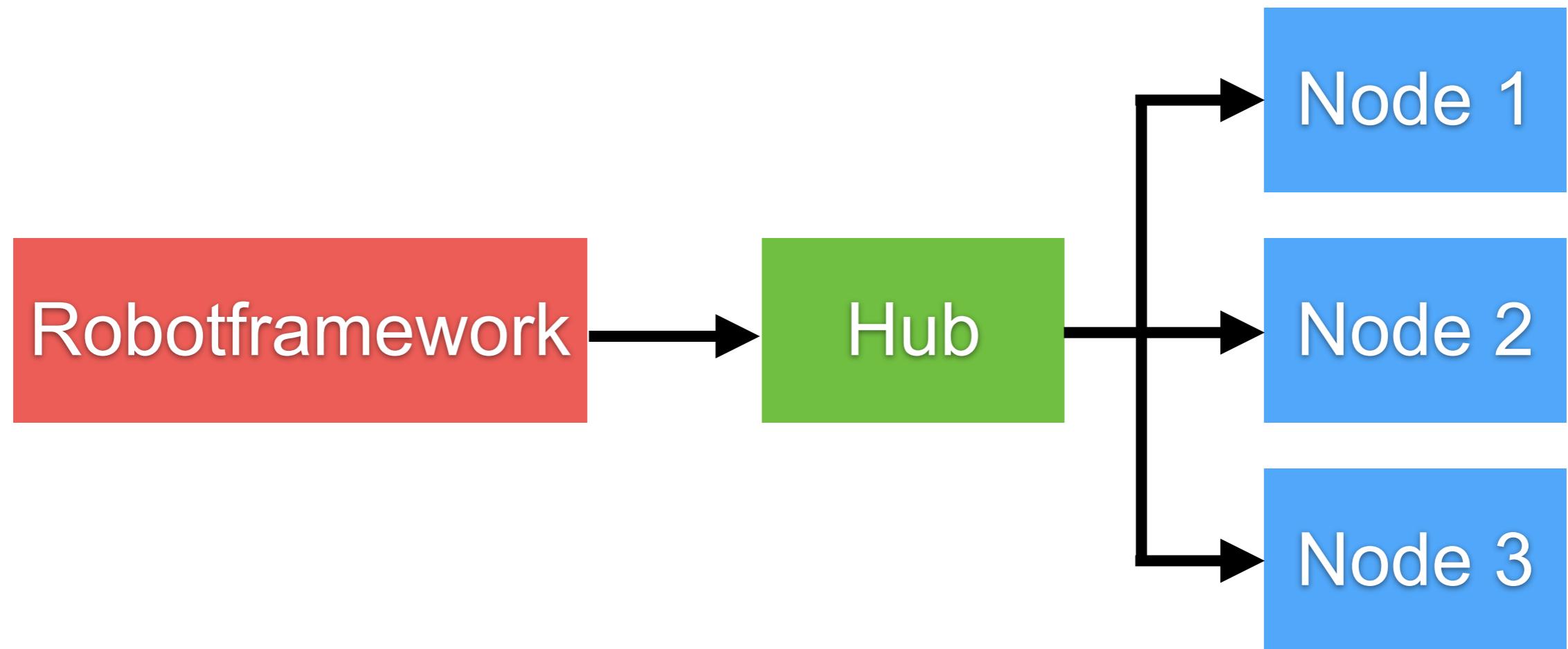
Start Node

Open url=http://localhost:4444/grid/console

The screenshot shows the Grid Console interface. At the top, there is a logo consisting of a stylized 'S' and 'e' icon followed by the text "Grid Console v.4.0.0-alpha-2". Below this, a dark bar displays the information: "DefaultRemoteProxy (version : 4.0.0-alpha-2)" and "id : http://192.168.1.33:25112, OS : MAC". The main area has two tabs: "Browsers" (which is active) and "Configuration". Under the "Browsers" tab, there is a section titled "WebDriver" with three rows of browser icons. The first row contains five Firefox icons (v:). The second row contains one Safari icon (v:). The third row contains five Chrome icons (v:). At the bottom left of the main area, there is a link labeled "View Config".



Testing with Robotframework



Testing with Robotframework

```
*** Keywords ***
```

```
Open with selenium grid
```

```
  Open Browser    ${URL}
    ...  browser=${BROWSER}
    ...  remote_url=http://localhost:4444/wd/hub
    ...  desired_capabilities=browserName:chrome
```



State of nodes in selenium grid

The screenshot shows the Selenium Grid Console interface. At the top, there is a logo consisting of three colored squares (orange, purple, yellow) arranged in a triangle, followed by the letters 'Se'. To the right of the logo, the text 'Grid Console v.4.0.0-alpha-2' is displayed. Below this, a dark blue header bar contains the text 'DefaultRemoteProxy (version : 4.0.0-alpha-2)' and 'id : http://192.168.1.33:29618, OS : MAC'. The main content area has two tabs: 'Browsers' (which is active) and 'Configuration'. Under the 'Browsers' tab, there is a section titled 'WebDriver' with three rows of browser icons. The first row contains five Firefox icons (v:). The second row contains one Internet Explorer icon (v:). The third row contains five Google Chrome icons (v:), with the entire row highlighted by a red rectangular border. At the bottom left of the main content area, there is a link labeled 'View Config'.



Close browser in each test case

```
*** Settings ***
Resource  ./pages/welcome.robot
Resource  ./pages/catalog.robot
Test Teardown  Close Browser
```



Testing life cycle

Test case 01

Test case 02



Testing life cycle

Suite Setup

Test case 01

Test case 02

Suite Teardown



Testing life cycle

Suite Setup

Test Setup

Test case 01

Test Teardown

Test case 02

Suite Teardown



Testing life cycle

Suite Setup

Test Setup

Test case 01

Test Teardown

Test Setup

Test case 02

Test Teardown

Suite Teardown



Workshop with Selenium grid



API testing



API testing

Robot framework
Postman



API testing with Robot

Using RequestsLibrary

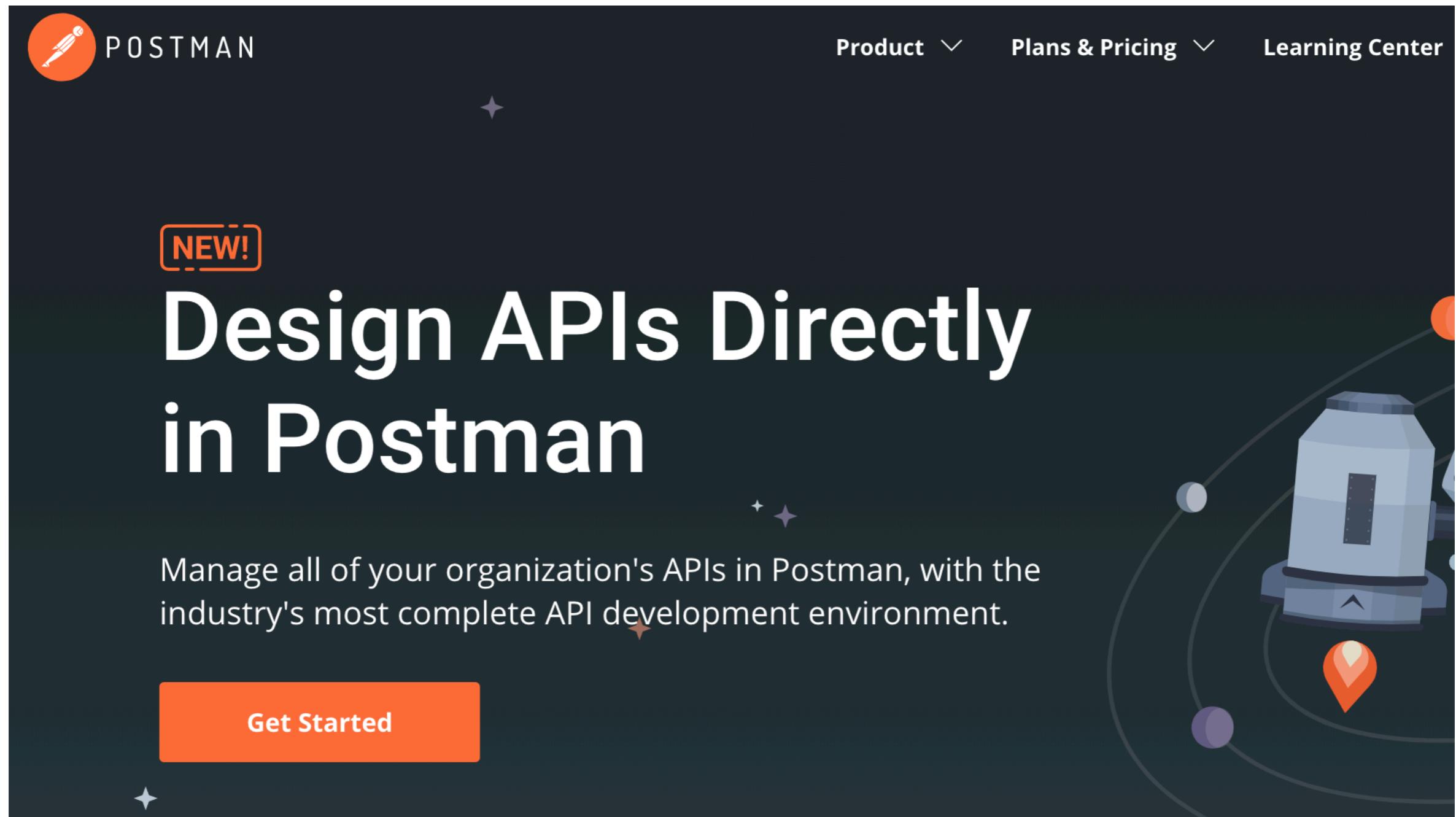
```
$pip install -U requests
```

```
$pip install -U robotframework-requests
```

<https://github.com/bulkan/robotframework-requests>



API testing with Postman



The screenshot shows the Postman homepage. At the top left is the Postman logo (a stylized orange circle with a white pen icon). At the top right are navigation links: 'Product' with a dropdown arrow, 'Plans & Pricing' with a dropdown arrow, and 'Learning Center'. Below the navigation is a large 'NEW!' badge. The main title 'Design APIs Directly in Postman' is displayed in large white font. A descriptive text below it reads: 'Manage all of your organization's APIs in Postman, with the industry's most complete API development environment.' To the right of the text is a graphic of a satellite in space, connected by lines to several colored dots (blue, purple, red) on a dark background. At the bottom left is an orange 'Get Started' button.

<https://www.getpostman.com/>



Postman in command line

\$npm install -g newman

\$newman run <collection file>

\$newman run <collection file> -r cli,junit

\$newman run <collection file> -r cli,junit

--reporter-junit-export outfile.xml

<https://www.npmjs.com/package/newman>



Workshop API testing



Workshop API testing

<https://jsonplaceholder.typicode.com/>

JSONPlaceholder

Fake Online REST API for Testing and Prototyping

Serving ~350M requests per month

Powered by [JSON Server](#) + [LowDB](#)

 [BECOME A PATRON](#)



Workshop API testing

<https://jsonplaceholder.typicode.com/users>

```
[  
  - {  
      id: 1,  
      name: "Leanne Graham",  
      username: "Bret",  
      email: "Sincere@april.biz",  
      - address: {  
          street: "Kulas Light",  
          suite: "Apt. 556",  
          city: "Gwenborough",  
          zipcode: "92998-3874",  
          - geo: {  
              lat: "-37.3159",  
              lng: "81.1496"  
          }  
      },  
  },
```



Agenda day 1

- Basic of python
- How to develop test library with python ?
- Type of test library
- Scope of test library
- How to use and publish test library ?
- Workshop



Agenda day 2

- Create keywords of test library
- How to generate documentation of test library ?
- Develop dynamic test library
- Develop test library by use case
- Workshop



Create library of Robot Framework



Programming language

Python
Java



Basic of Python



Python 3+



Hello with Python

```
def sayHi(name):  
    return 'Hi, %s' %(name)  
  
if __name__ == '__main__':  
    result = sayHi('somkiat')  
    print(result)
```



OOP with Python



Hello with OOP

```
class Hello:  
    def __init__(self):  
        self._result = ''  
  
    def set_name(self, name):  
        self._name = name  
  
    def get_result(self):  
        return 'Hi, %s' %(self._name)  
  
if __name__ == '__main__':  
    h = Hello()  
    h.set_name('somkiat')  
    print(h.get_result())
```



Custom Library with Robot Framework



Types of Library

Static library

Dynamic library

Hybrid library



Static Library



Hello World (1)

Create a new library with python

```
1 *** Settings ***
2 Library    HelloWorld.py
3
4 *** Testcases ***
5 First library
6     Say Hi
7
8 Second library with argument
9     Say Hi    somkiat
```



Hello World (2)

Create file **HelloWorld.py** and method **say_hi()**

```
1  def say_hi(name = ""):  
2      print("Say hi " + name)  
3
```



Hello World (3)

\$pybot test.robot

```
=====
```

Test

```
=====
```

First library	PASS
---------------	------

Second library with argument	PASS
------------------------------	------

Test	PASS
------	------

2 critical tests, 2 passed, 0 failed

2 tests total, 2 passed, 0 failed

```
=====
```



Hello World (4)

See in report.html

Test Details

Totals **Tags** **Suites** **Search**

Name: Test

Status: 2 critical test, 2 passed, 0 failed
2 test total, 2 passed, 0 failed

Start / End Time: 20180603 22:28:28.540 / 20180603 22:28:28.578

Elapsed Time: 00:00:00.038

Log File: log.html#s1

Name	Documentation
Test. First library	
Test. Second library with argument	



Improve naming of Library



Improve name of library

Need to change to HelloWorld

```
1 *** Settings ***
2 Library    HelloWorld
3
4 *** Testcases ***
5 First library
6     Say Hi
7
8 Second library with argument
9     Say Hi    somkiat
```



Run with python path

```
$pybot --pythonpath . test.robot
```

```
=====
```

```
Test
```

```
=====
```

```
First library | PASS |
```

```
Second library with argument | PASS |
```

```
Test | PASS |
```

```
2 critical tests, 2 passed, 0 failed
```

```
2 tests total, 2 passed, 0 failed
```



Working with OOP



Hello World (2)

Create file **HelloWorld.py** and method **say_hi()**

```
1  class HelloWorld:  
2      def say_hi(self, name = ""):  
3          print("Say hi " + name)  
4  
5
```



Scope of test library



Scope of Test Library

TEST CASE (default)
TEST SUITE
GLOBAL



TEST CASE

Create a new instance for every test case



TEST SUITE

Create a new instance for every test suite



GLOBAL

Only one instance and shared by all test cases
and test suites



Scopes (1)

GLOBAL



Scopes (2)

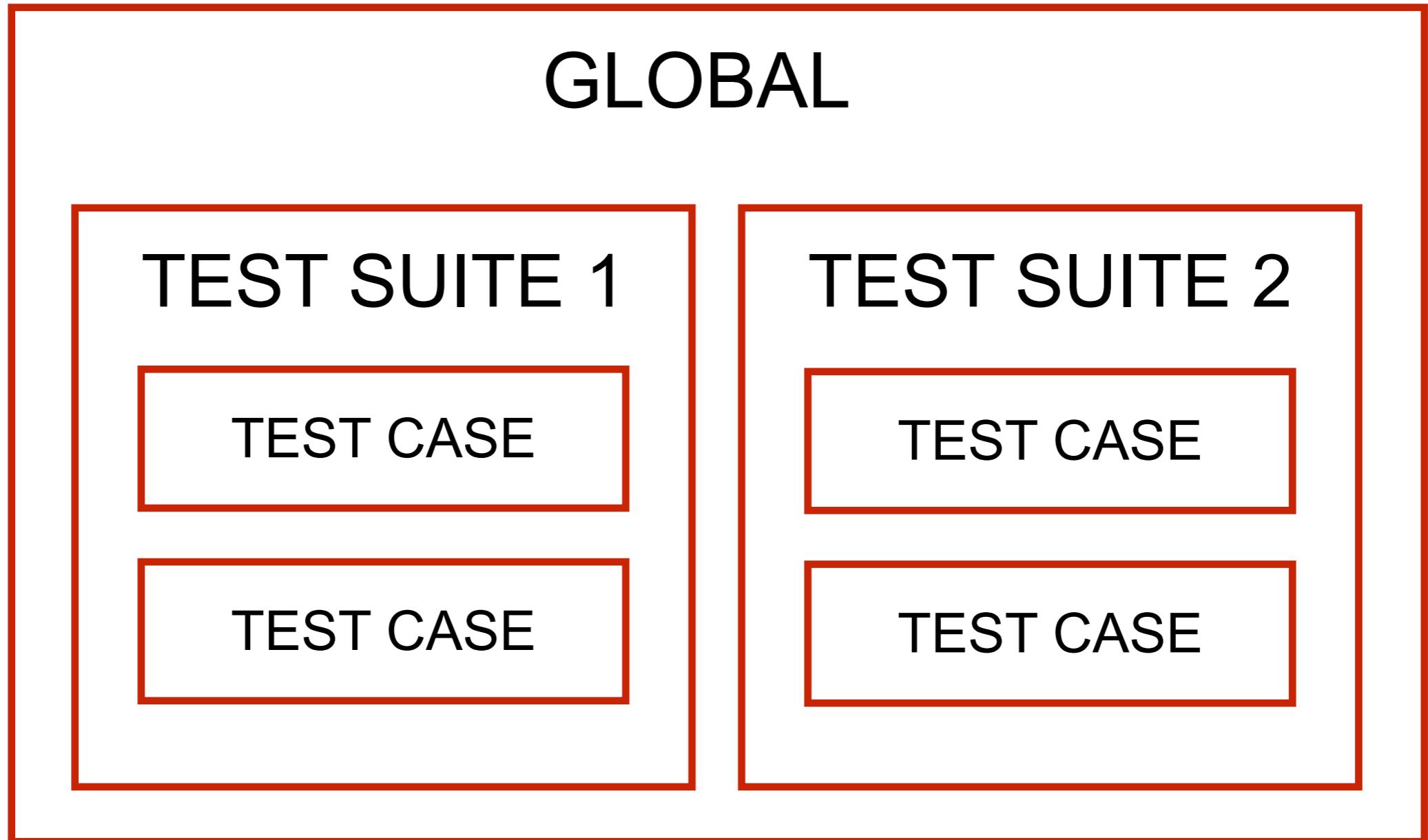
GLOBAL

TEST SUITE 1

TEST SUITE 2



Scopes (3)



Using TEST CASE scope

Create attribute ROBOT_LIBRARY_SCOPE

```
1  class HelloWorld:  
2      ROBOT_LIBRARY_SCOPE = 'TEST CASE'  
3  
4      def __init__(self):  
5          self.name = "Noname"  
6  
7      def say_hi(self):  
8          print("Say hi " + self.name)  
9  
10     def say_hi2(self, name):  
11         self.name = name  
12         print("Say hi " + self.name)  
13
```



My test cases

```
1 *** Settings ***
2 Library    HelloWorld
3
4 *** Testcases ***
5 First library
6     Say Hi
7
8 Second library with argument
9     Say Hi2    somkiat
10
11 Third library
12     Say Hi
```



Run with python path

- TEST First library

Full Name: Test.First library
Start / End / Elapsed: 20180603 23:22:02.490 / 20180603 23:22:02.491 / 00:00:00.001
Status: PASS (critical)

- KEYWORD HelloWorld.Say Hi

Start / End / Elapsed: 20180603 23:22:02.491 / 20180603 23:22:02.491 / 00:00:00.000
23:22:02.491 INFO Say hi Noname

Name = “Noname”

- TEST Second library with argument

Full Name: Test.Second library with argument
Start / End / Elapsed: 20180603 23:22:02.492 / 20180603 23:22:02.493 / 00:00:00.001
Status: PASS (critical)

- KEYWORD HelloWorld.Say Hi2 somkiat

Start / End / Elapsed: 20180603 23:22:02.492 / 20180603 23:22:02.493 / 00:00:00.001
23:22:02.492 INFO Say hi somkiat

Name = “somkiat”

- TEST Third library

Full Name: Test.Third library
Start / End / Elapsed: 20180603 23:22:02.493 / 20180603 23:22:02.494 / 00:00:00.001
Status: PASS (critical)

- KEYWORD HelloWorld.Say Hi

Start / End / Elapsed: 20180603 23:22:02.494 / 20180603 23:22:02.494 / 00:00:00.000
23:22:02.494 INFO Say hi Noname

Name = “Noname”



Using TEST SUITE scope

```
1  class HelloWorld:  
2      ROBOT_LIBRARY_SCOPE = 'TEST SUITE'  
3  
4      def __init__(self):  
5          self.name = "Noname"  
6  
7      def say_hi(self):  
8          print("Say hi " + self.name)  
9  
10     def say_hi2(self, name):  
11         self.name = name  
12         print("Say hi " + self.name)
```



Run with python path

-	TEST First library
Full Name:	Test.First library
Start / End / Elapsed:	20180603 23:16:43.825 / 20180603 23:16:43.826 / 00:00:00.001
Status:	PASS (critical)
-	KEYWORD HelloWorld.Say Hi
Start / End / Elapsed:	20180603 23:16:43.825 / 20180603 23:16:43.826 / 00:00:00.001 23:16:43.826 INFO Say hi Noname
Name = “Noname”	
-	TEST Second library with argument
Full Name:	Test.Second library with argument
Start / End / Elapsed:	20180603 23:16:43.826 / 20180603 23:16:43.827 / 00:00:00.001
Status:	PASS (critical)
-	KEYWORD HelloWorld.Say Hi2 somkiat
Start / End / Elapsed:	20180603 23:16:43.827 / 20180603 23:16:43.827 / 00:00:00.000 23:16:43.827 INFO Say hi somkiat
Name = “somkiat”	
-	TEST Third library
Full Name:	Test.Third library
Start / End / Elapsed:	20180603 23:16:43.828 / 20180603 23:16:43.829 / 00:00:00.001
Status:	PASS (critical)
-	KEYWORD HelloWorld.Say Hi
Start / End / Elapsed:	20180603 23:16:43.828 / 20180603 23:16:43.828 / 00:00:00.000 23:16:43.828 INFO Say hi somkiat
Name = “somkiat”	



Using GLOBAL scope

```
1  class HelloWorld:  
2      ROBOT_LIBRARY_SCOPE = 'GLOBAL'  
3  
4      def __init__(self):  
5          self.name = "Noname"  
6  
7      def say_hi(self):  
8          print("Say hi " + self.name)  
9  
10     def say_hi2(self, name):  
11         self.name = name  
12         print("Say hi " + self.name)  
13
```



Run with python path

-	TEST First library
Full Name:	Test.First library
Start / End / Elapsed:	20180603 23:16:43.825 / 20180603 23:16:43.826 / 00:00:00.001
Status:	PASS (critical)
-	KEYWORD HelloWorld.Say Hi
Start / End / Elapsed:	20180603 23:16:43.825 / 20180603 23:16:43.826 / 00:00:00.001 23:16:43.826 INFO Say hi Noname
Name = “Noname”	
-	TEST Second library with argument
Full Name:	Test.Second library with argument
Start / End / Elapsed:	20180603 23:16:43.826 / 20180603 23:16:43.827 / 00:00:00.001
Status:	PASS (critical)
-	KEYWORD HelloWorld.Say Hi2 somkiat
Start / End / Elapsed:	20180603 23:16:43.827 / 20180603 23:16:43.827 / 00:00:00.000 23:16:43.827 INFO Say hi somkiat
Name = “somkiat”	
-	TEST Third library
Full Name:	Test.Third library
Start / End / Elapsed:	20180603 23:16:43.828 / 20180603 23:16:43.829 / 00:00:00.001
Status:	PASS (critical)
-	KEYWORD HelloWorld.Say Hi
Start / End / Elapsed:	20180603 23:16:43.828 / 20180603 23:16:43.828 / 00:00:00.000 23:16:43.828 INFO Say hi somkiat
Name = “somkiat”	



Run with another test suite

Create new test suite => test2.robot

```
1 *** Settings ***
2 Library    HelloWorld
3
4 *** Testcases ***
5 Another test case
6 Say Hi
```



Run with python path

\$pybot --pythonpath . *.robot

- SUITE Test2

Full Name:	Test & Test2.Test2
Source:	/Users/somkiat/data/slide/robot-framework/advanc
Start / End / Elapsed:	20180603 23:31:04.939 / 20180603 23:31:04.942
Status:	1 critical test, 1 passed, 0 failed 1 test total, 1 passed, 0 failed

- TEST Another test case

Full Name:	Test & Test2.Test2.Another test case
Start / End / Elapsed:	20180603 23:31:04.941 / 20180603 23:31:04.942
Status:	PASS (critical)

- KEYWORD HelloWorld.Say Hi

Start / End / Elapsed:	20180603 23:31:04.941 / 20180603 23:31:04.942
23:31:04.942	INFO Say hi somkiat

Name = “somkiat”



Publish Library



Publish Library

Git provider => Github
pypi.org



Publish Library with pypi.org



<https://packaging.python.org/guides/migrating-to-pypi-org/#uploading>



Step 1

Register account at <https://pypi.org/>

Help Donate Log in Register

Find, install and publish Python packages
with the Python Package Index

Search projects

Or browse projects

140,809 projects 985,505 releases 1,320,823 files 279,892 users

python™
Package
Index

The Python Package Index (PyPI) is a repository of software for the Python programming language. PyPI helps you find and install software developed and shared by the Python community. [Learn about installing packages.](#)



Step 2

Verify and see your project

The screenshot shows the Python Package Index (PyPI) dashboard. At the top, there is a blue header bar with a logo, a search bar containing "Search projects" and a magnifying glass icon, and a welcome message "Welcome back, somkiat ▾". Below the header, there are two main sections: "Your account" on the left and "Your projects (0)" on the right. The "Your account" section contains links for "Your projects" (which is highlighted with a red border) and "Account settings". The "Your projects (0)" section contains a message: "You have not uploaded any projects to PyPI, yet. To learn how to get started, visit the [Python Packaging User Guide](#)". At the bottom of the page, there is a footer with four columns: "Get help", "About PyPI", "Contributing to PyPI", and "Using PyPI". Each column lists several links related to its category.

Get help	About PyPI	Contributing to PyPI	Using PyPI
Installing packages	Status: All Systems Operational	Bugs & feedback	Code of conduct
Uploading packages	Package index name retention	Contribute on GitHub	Security policy
User guide	Our sponsors	Development credits	Privacy policy
FAQs			Terms of use



Step 3

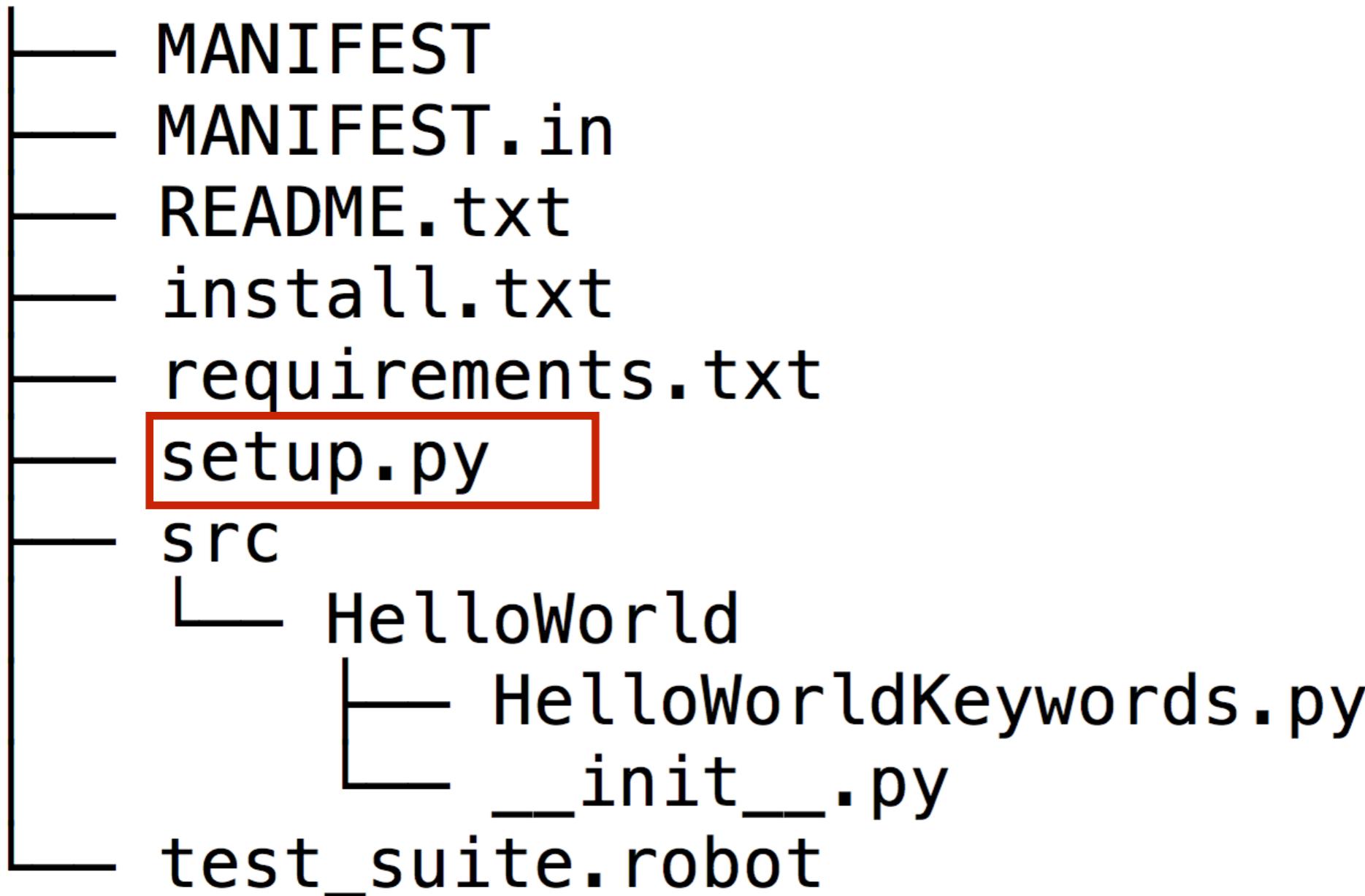
Start to develop your package

```
├── MANIFEST  
├── MANIFEST.in  
├── README.txt  
├── install.txt  
├── requirements.txt  
├── setup.py  
└── src  
    └── HelloWorld  
        ├── HelloWorldKeywords.py  
        └── __init__.py  
└── test_suite.robot
```



Step 4 (1)

Create file `setup.py` to configure test library



Step 4 (2)

Specify name and version of library

```
1 from setuptools import setup  
2  
3 setup(  
4     name="helloworld-library",  
5     version='0.1',  
6     package_dir={'': 'src'},  
7     packages=['HelloWorld'],  
8     url='https://github.com/up1/demo-helloworld-library',  
9     author='Somkiat',  
10    author_email='somkiat.p@gmail.com',  
11 )
```



Step 4 (3)

Specify package structure and name

```
1 from setuptools import setup  
2  
3 setup(  
4     name="helloworld-library",  
5     version='0.1',  
6     package_dir={'': 'src'},  
7     packages=['HelloWorld'],  
8     url='https://github.com/up1/demo-helloworld-library',  
9     author='Somkiat',  
10    author_email='somkiat.p@gmail.com',  
11 )
```



Step 4 (4)

Required metadata of test library

```
1 from setuptools import setup  
2  
3 setup(  
4     name="helloworld-library",  
5     version='0.1',  
6     package_dir={'': 'src'},  
7     packages=['HelloWorld'],  
8     url='https://github.com/up1/demo-helloworld-library',  
9     author='Somkiat',  
10    author_email='somkiat.p@gmail.com',  
11 )
```



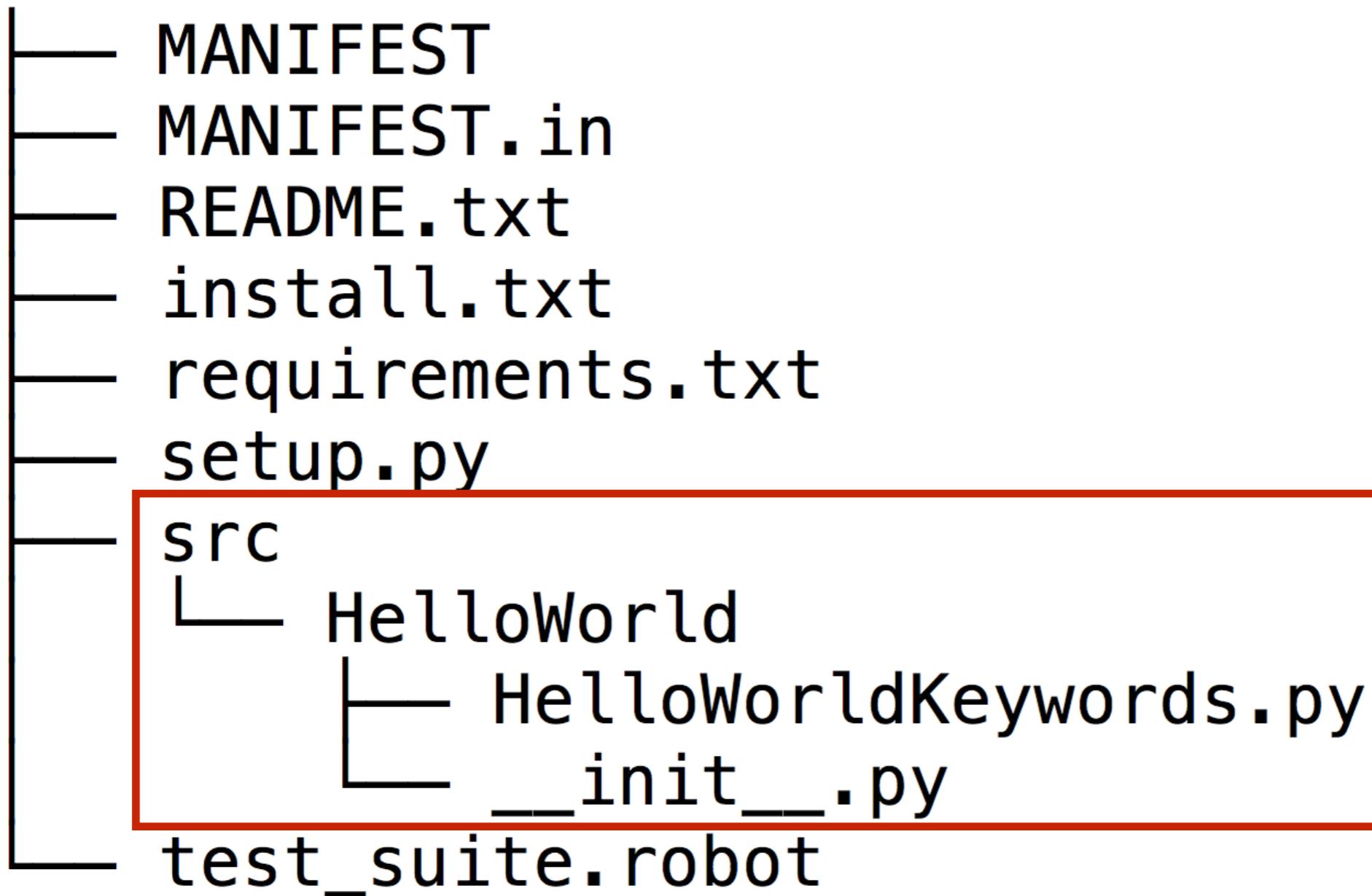
Step 5

Develop HelloWorld library



Structure of package

Create directory src/HelloWorld



Define keywords of library

Create file HelloWorldKeywords.py

```
1  class HelloWorldKeywords(object):
2      def __init__(self):
3          self.name = "Noname"
4
5      def say_hi(self):
6          print("Say hi " + self.name)
7
8      def say_hi2(self, name):
9          self.name = name
10         print("Say hi " + self.name)
11
```



Define keywords of library

Create file `__init__.py`

```
1  from HelloWorldKeywords import HelloWorldKeywords  
2  
3  class HelloWorld(HelloWorldKeywords):  
4      ROBOT_LIBRARY_SCOPE = 'TEST_CASE'  
5
```



Step 6

Publish library to pypi.org



Create file `~/.pypirc`

Configuration for publish library to pypi.org

```
1 [distutils]
2 index-servers =
3   pypi
4
5 [pypi]
6 #repository=https://pypi.python.org/pypi
7 username=<your username>
8 password=<your password>
```



Publish library to pypi.org

```
$pip install -U pip setuptools twine
```

```
$python setup.py sdist
```

```
$twine upload dist/*
```

```
Writing helloworld-library-0.2/setup.cfg
```

```
Creating tar archive
```

```
removing 'helloworld-library-0.2' (and everything under it)
```

```
Uploading distributions to https://upload.pypi.org/legacy/
```

```
Uploading helloworld-library-0.2.tar.gz
```

```
100%|██████████| 3.54k/3.54k [00:01<00:00, 2.86kB/s]
```



Check your library (1)

Go to pypi.org

The screenshot shows the PyPI (Python Package Index) website interface. At the top, there is a blue header bar with a logo on the left, a search bar containing "Search projects" with a magnifying glass icon, and a welcome message "Welcome back, somkiat ▾" on the right. Below the header, the main content area has two sections: "Your account" on the left and "Your projects (1)" on the right. The "Your account" section contains two buttons: "Your projects" (which is highlighted in blue) and "Account settings". The "Your projects (1)" section displays a single project card for "helloworld-library", which was last released on Jun 3, 2018. The card includes a small cube icon, the project name, the release date, and two buttons: "Manage" (blue) and "View" (white).



Check your library (2)

helloworld-library 0.1

pip install helloworld-library 

 Latest version

Last released: About 5 hours ago.

No project description provided

Manage project

Navigation

- [Project description](#)
- [Release history](#)
- [Download files](#)

Project links

- [Homepage](#)

Project description

The author of this package has not provided a project description



Use HelloWorld library

```
$pip install helloworld-library
```

```
Collecting helloworld-library
```

```
  Downloading https://files.pythonhosted.org/packages/8f/a2  
92e220deb5cb908b1d6f358f1d36e8e8307e9211a8c382b91a0225/hell  
library-0.2.tar.gz
```

```
Building wheels for collected packages: helloworld-library
```

```
  Running setup.py bdist_wheel for helloworld-library ... c
```

```
  Stored in directory: /Users/somkiat/Library/Caches/pip/wk  
/b4/6b/db550e3f32243f1d2397f064d34ed13b3178cb7b90b29f4c5e
```

```
Successfully built helloworld-library
```

```
Installing collected packages: helloworld-library
```

```
Successfully installed helloworld-library-0.2
```



Use HelloWorld library

```
1 *** Settings ***
2 Library    HelloWorld
3
4 *** Testcases ***
5 First library
6     Say Hi
7
8 Second library with argument
9     Say Hi2    somkiat
10
```

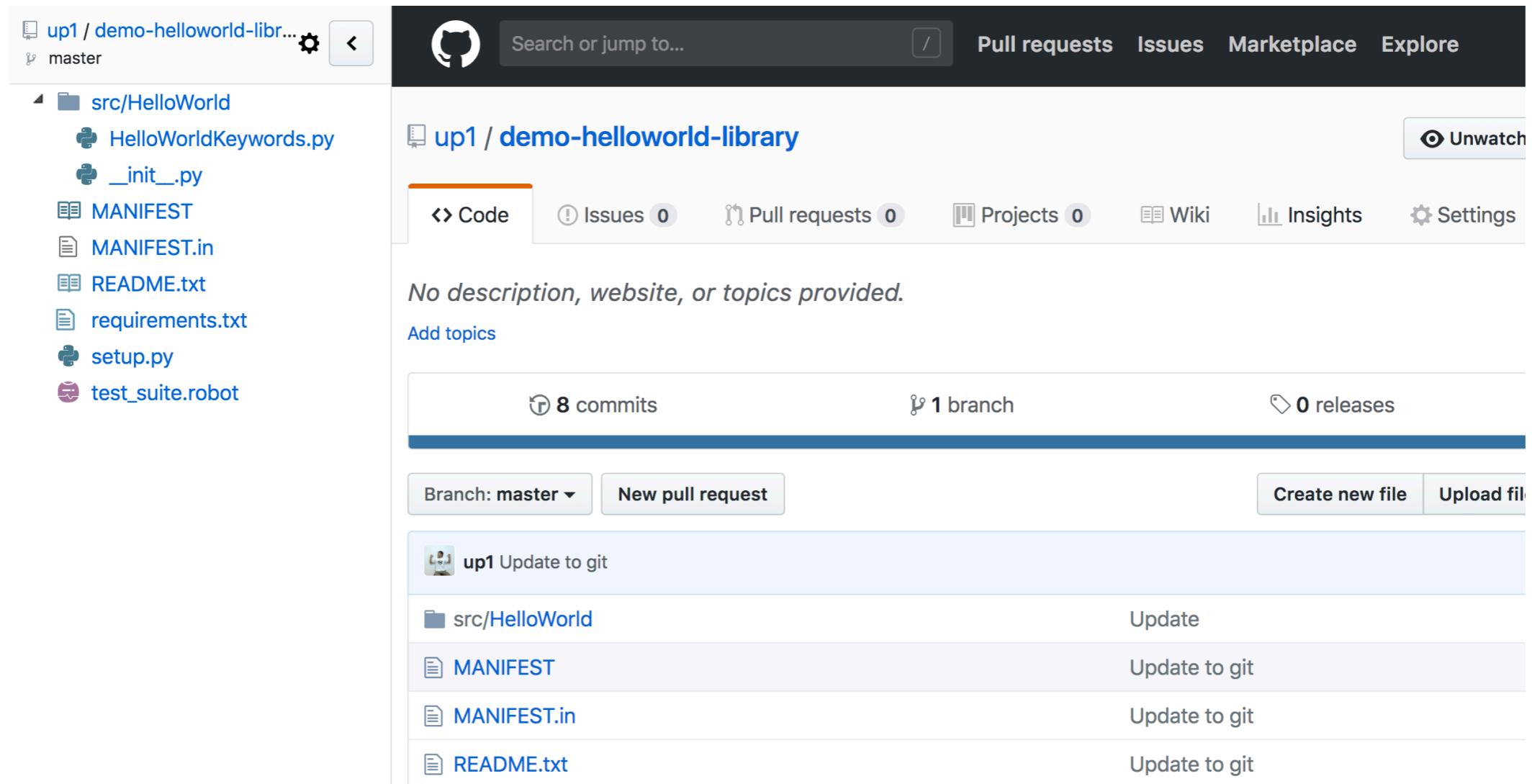


Publish Library with github



Publish library to github

1. Push your code to your Github repository



<https://github.com/up1/demo-helloworld-library>



2. Install library from Github (1)

\$pip install -r requirements.txt

\$pip uninstall -r requirements.txt

git+https://github.com/up1/demo-helloworld-library.git#egg=helloworld-library



Name of library



2. Install library from Github (2)

\$pip install -r requirements.txt

```
Collecting helloworld-library from git+https://github.com/up1  
library.git#egg=helloworld-library (from -r requirements.txt  
  Cloning https://github.com/up1/demo-helloworld-library.git  
    Olders/t5/8kg23s_97z9dw44tfc1d6dqw0000gn/T/pip-install-d20dok  
      rary
```

```
Building wheels for collected packages: helloworld-library  
  Running setup.py bdist_wheel for helloworld-library ... done  
  Stored in directory: /private/var/folders/t5/8kg23s_97z9dw4  
T/pip-ephem-wheel-cache-svb7x4pk/wheels/6e/77/72/2c1098f915d8  
e47d24d0c0f106fe5b667
```

```
Successfully built helloworld-library  
Installing collected packages: helloworld-library  
Successfully installed helloworld-library-0.2
```



How to generate document of test library ?

<http://robotframework.org/robotframework/latest/RobotFrameworkUserGuide.html#specifying-documentation-format>



Generate document of library

Robotframework 2.7.5+ use **Libdoc** to generate the documentation of library

<http://robotframework.org/robotframework/latest/RobotFrameworkUserGuide.html#libdoc>



Support formats

ROBOT (default)

HTML

TEXT (plain text)

reST (reStructuredText)



How to use ?

Example with ROBOT format

```
1 from HelloWorldKeywords import HelloWorldKeywords  
2  
3 class HelloWorld(HelloWorldKeywords):  
4     """ A keyword library for Robot Framework. It provides keywords for  
5     learning how to create a new library. For more information  
6     on underlying methods and documentation, see:  
7         http://eclipse.org/paho/clients/python/docs/  
8     """  
9  
10    ROBOT_LIBRARY_SCOPE = 'TEST_CASE'  
11  
12    |
```



How to use ?

Document in each keyword

```
5      def say_hi(self):
6          """ Say hi with out argument
7          Examples:
8          | Say Hi |
9          """
10         print("Say hi " + self.name)
11
12         def say_hi2(self, name):
13             """ Say hi with a argument.
14             `name` Your name
15             Examples:
16             Say hi  <name>
17             | Say Hi | somkiat |
18             """
19             self.name = name
20             print("Say hi " + self.name)
```



Generate documentation

```
$pip install -U helloworld-library
```

```
$python -m robot.libdoc HelloWorld ./docs/  
HelloWorld-Library.html
```



Documentation of Library (1)

HelloWorld

Library scope: test case
Named arguments: supported

Introduction

A keyword library for Robot Framework. It provides keywords for learning how to create a new library. For more information on underlying

Shortcuts

Say Hi · Say Hi2

Keywords

Keyword	Arguments	
Say Hi		Say hi with out argument Examples: <code>Say Hi</code>
Say Hi2	<i>name</i>	Say hi with a argument. <i>name</i> Your name Examples <code>Say Hi somkiat</code>

Altogether 2 keywords.

Generated by [Libdoc](#) on 2018-06-04 00:34:08.



Documentation of Library (2)

HelloWorld

Library scope: test case
Named arguments: supported

Introduction

A keyword library for Robot Framework. It provides keywords for learning how to create a new library. For more information on underlying

Shortcuts

Say Hi · Say Hi2

Keywords

Keyword	Arguments	
Say Hi		Say hi with out argument Examples: Say Hi
Say Hi2	name	Say hi with a argument. <i>name</i> Your name Examples Say Hi somkiat

Altogether 2 keywords.

Generated by [Libdoc](#) on 2018-06-04 00:34:08.

