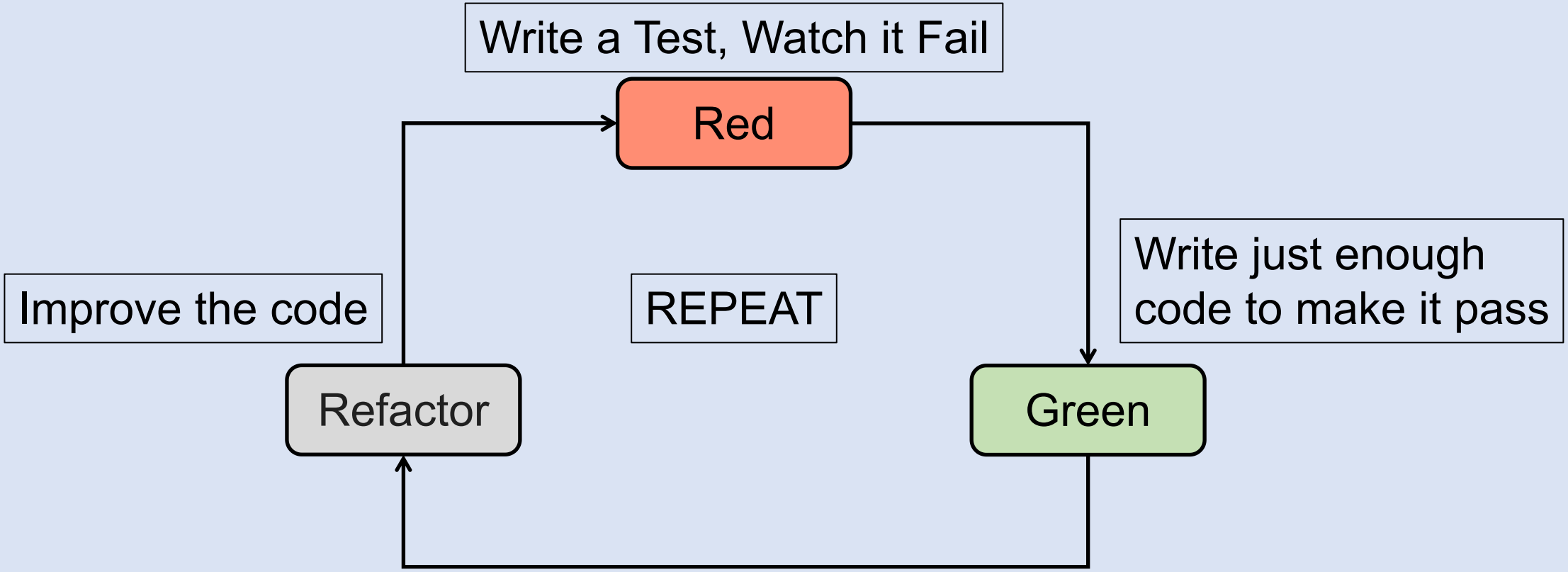
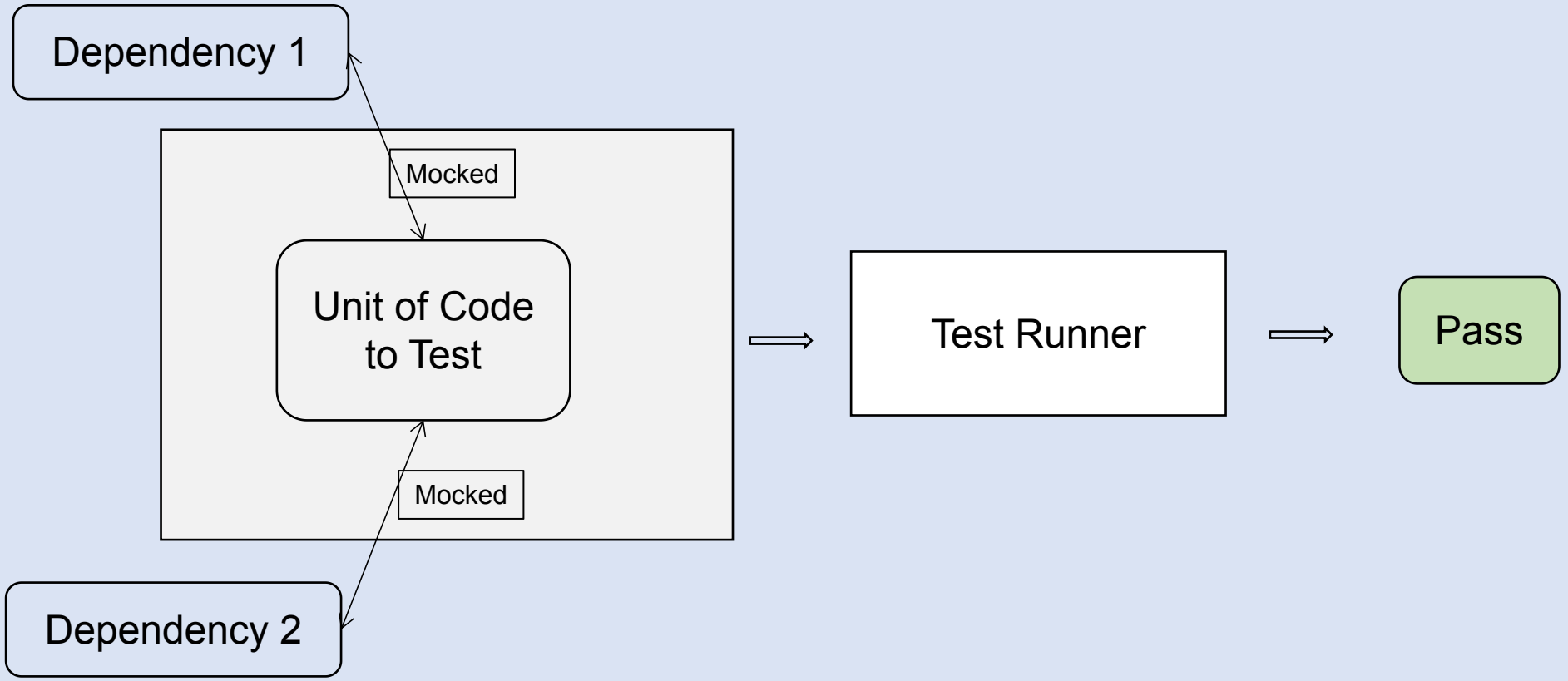
Elevate Your WordPress Plugin Development with TDD

Sudip Limbu
Full-Stack Developer

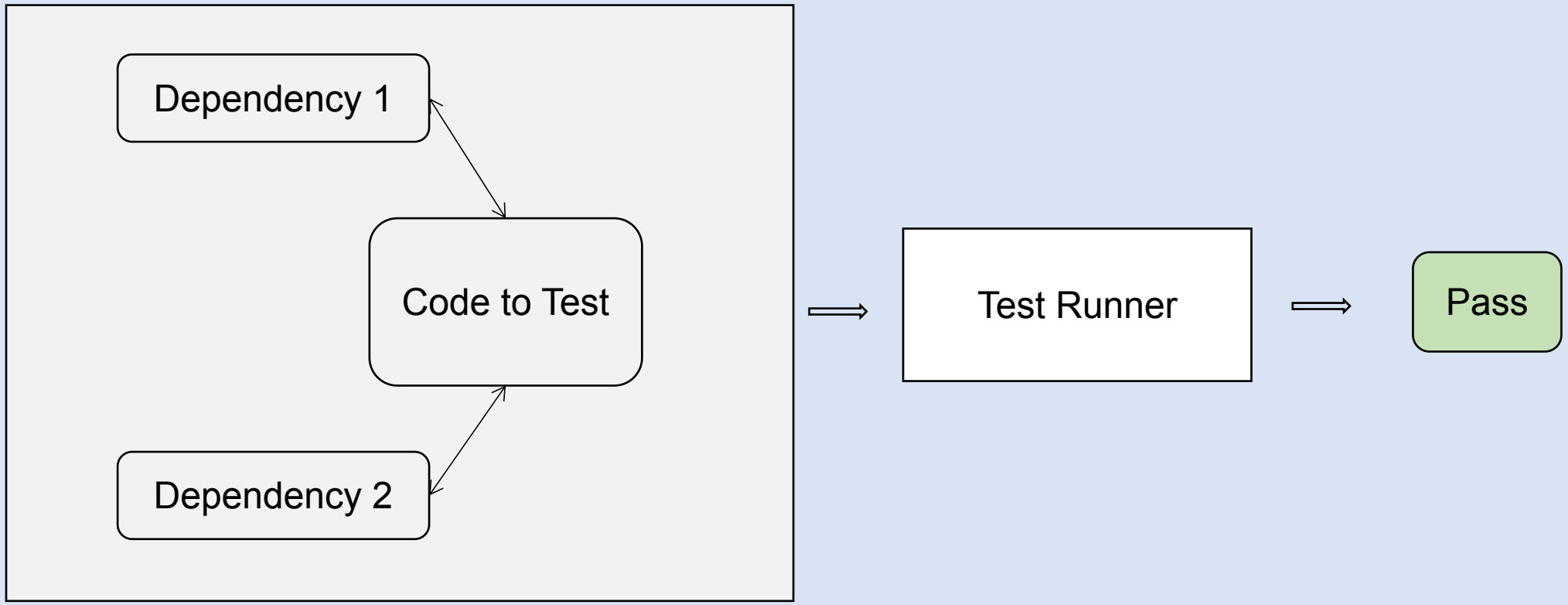
What is TDD?


Write a Test, Watch it Fail
Red
Write just enough code to make it pass
Green
Refactor
Improve the code
REPEAT

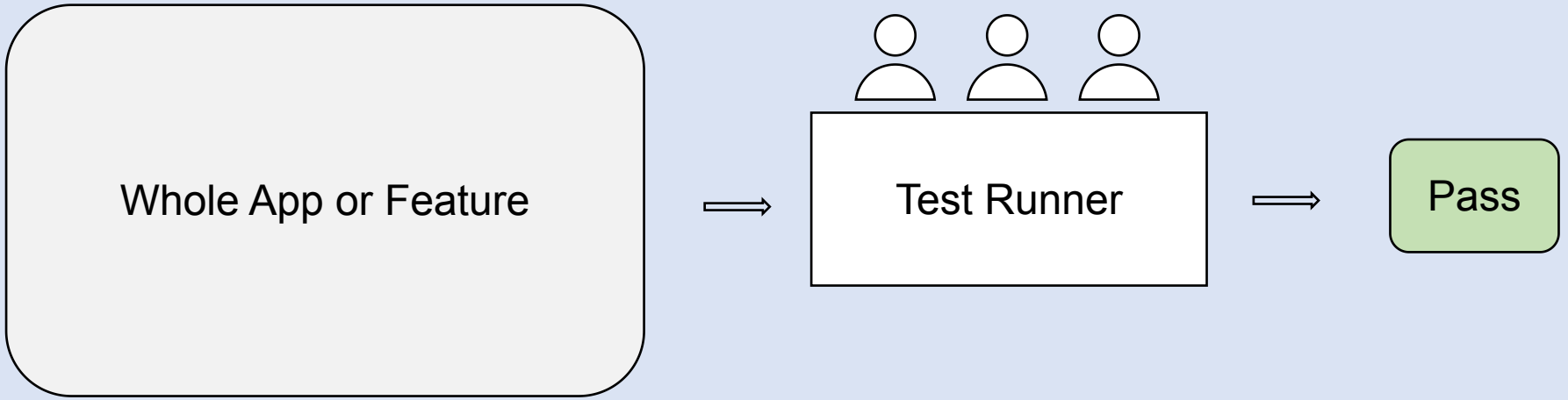
Unit Test


Dependency 1
Mocked
Unit of Code to Test
Mocked
Dependency 2
Test Runner
Pass

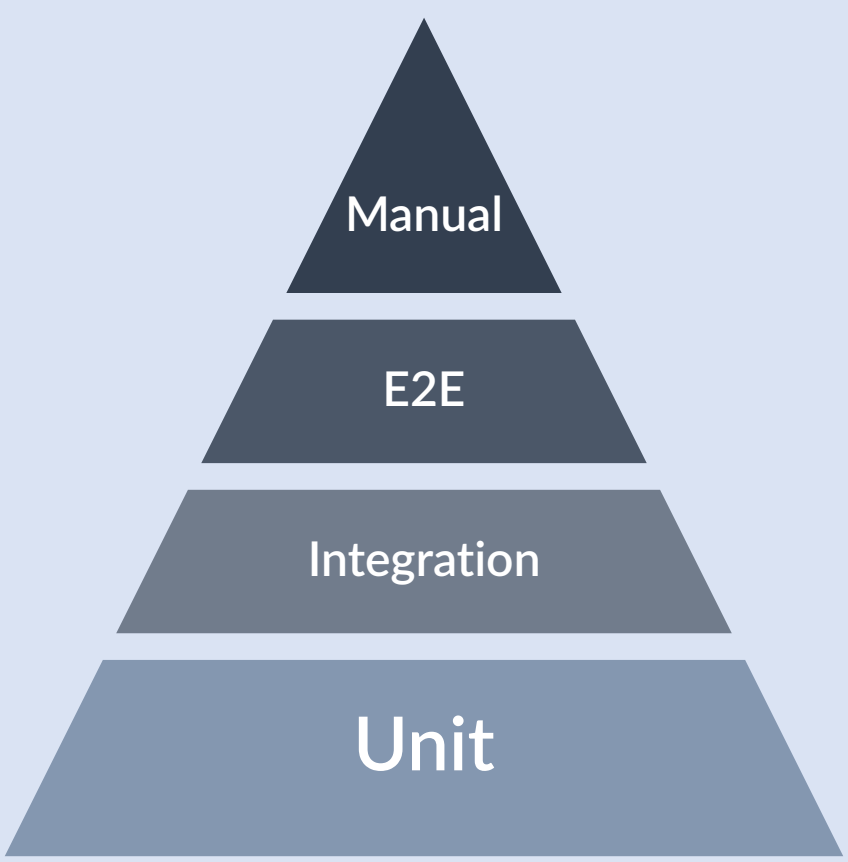
Integration Test



E2E Test



Testing Pyramid



- Unreliable
- Slow

- Automated
- Fast
- More Reliable

Setup Environment for plugin testing

- Plugin Scaffolding (using WP-CLI)

wp scaffold plugin

Generates starter code for a plugin.

View Open Issues (3) | View Closed Issues (51) | Create New Issue

The following files are always generated:

- `plugin-slug.php` is the main PHP plugin file.
- `readme.txt` is the readme file for the plugin.
- `package.json` needed by NPM holds various metadata relevant to the project. Packages: `grunt`, `grunt-wp-i18n` and `grunt-wp-readme-to-markdown`. Scripts: `start`, `readme`, `i18n`.
- `Gruntfile.js` is the JS file containing Grunt tasks. Tasks: `i18n` containing `addtextdomain` and `makepot`, `readme`

Setup Environment for plugin testing

- Plugin Scaffolding (using WP-CLI)
- WP-ENV

Search Block Editor

CHAPTERS

Block Editor Handbook
Getting Started
How-to Guides
Reference Guides
Block API Reference
Core Blocks Reference

@wordpress/env

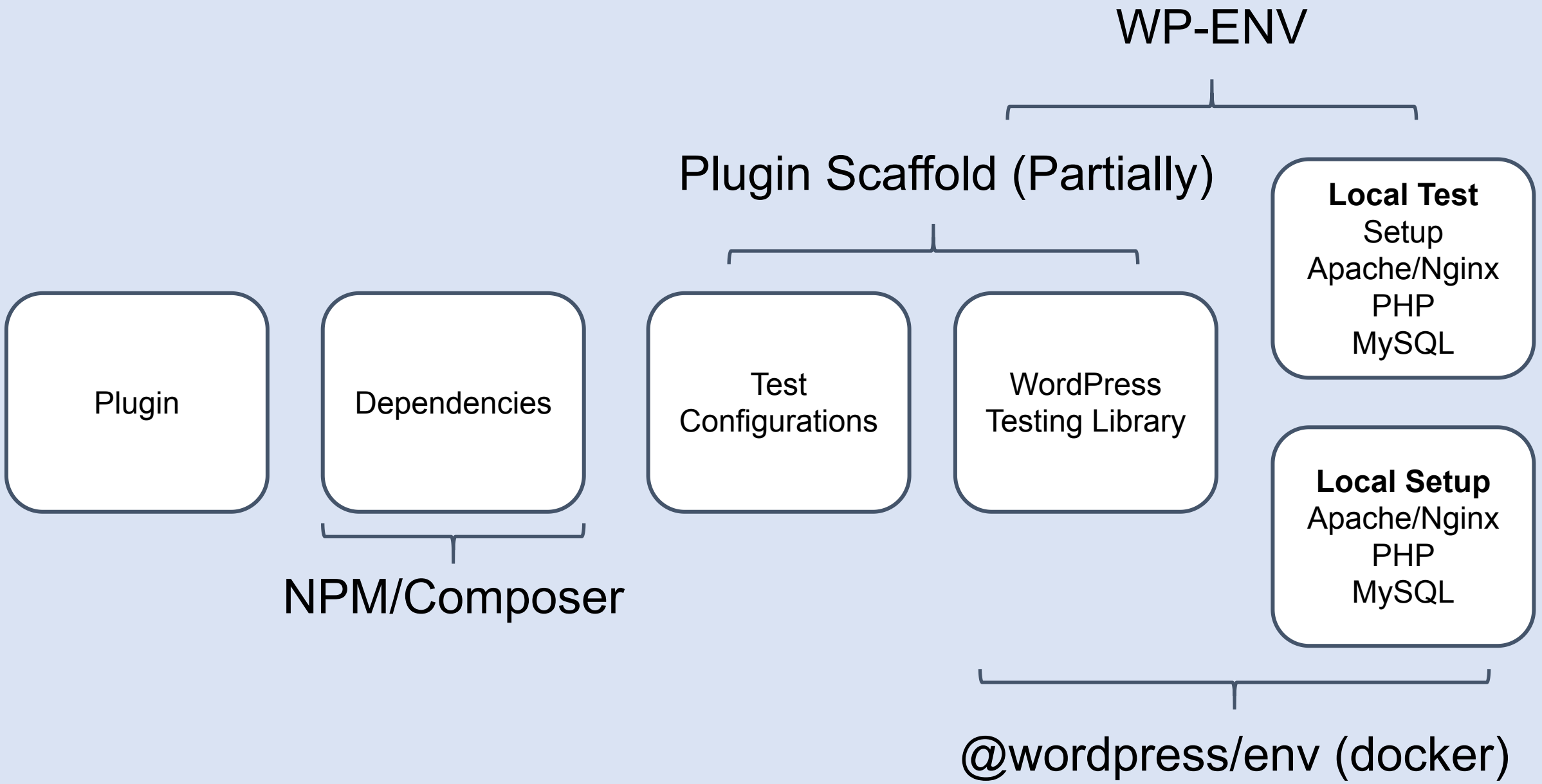
Edit

`wp-env` lets you easily set up a local WordPress environment for building and testing plugins and themes. It's simple to install and requires no configuration.

Quick (tl;dr) instructions

Ensure that Docker is running, then:

```
$ cd /path/to/a/wordpress/plugin
$ npm -g i @wordpress/env
$ wp-env start
```


WP-ENV
Plugin Scaffold (Partially)
Local Test
Setup
Apache/Nginx
PHP
MySQL
Plugin
Dependencies
Test
Configurations
WordPress
Testing Library
Local Setup
Apache/Nginx
PHP
MySQL
NPM/Composer
@wordpress/env (docker)

```
~/wp-env/..
 be4c0dc59f7ae6d3aba26b489efc6fad/
   tests-WordPress/
   tests-WordPress-PHPUnit/
     .git/
     tests/
   WordPress/
   WordPress-PHPUnit/
   CLI.Dockerfile
   docker-compose.yml
   Tests-CLI.Dockerfile
   Tests-WordPress.Dockerfile
   WordPress.Dockerfile
   wp-env-cache.json
```



```
13   tests-mysql:
14     image: mariadb
15     ports:
16       - '3306'
17     environment:
18       MYSQL_ROOT_HOST: '%'
19       MYSQL_ROOT_PASSWORD: password
20       MYSQL_DATABASE: tests-wordpress
21     volumes:
22       - 'mysql-test:/var/lib/mysql'
23   wordpress:
24     depends_on:
25       - mysql
26     build:
27       context: .
28       dockerfile: WordPress.Dockerfile
29       args: &ref_0
30         HOST_USERNAME: sudip
31         HOST_UID: '1000'
32         HOST_GID: '1000'
       --> build:
33     ports:
```

```
sudip@l:~/wp-env/be4c0dc59f7ae6d3aba26b489efc6fad$ docker-compose ps
                    Name                                  Command               State                    Ports
-------------------------------------------------------------------------------------------------------------------
be4c0dc59f7ae6d3aba26b489efc6fad_cli_1              docker-entrypoint.sh /bin/ ...   Up
be4c0dc59f7ae6d3aba26b489efc6fad_mysql_1            docker-entrypoint.sh mariadbd    Up      0.0.0.0:32768->3306/tcp,:
::32768->3306/tcp
be4c0dc59f7ae6d3aba26b489efc6fad_tests-cli_1         docker-entrypoint.sh /bin/ ...   Up
be4c0dc59f7ae6d3aba26b489efc6fad_tests-mysql_1       docker-entrypoint.sh mariadbd    Up      0.0.0.0:32769->3306/tcp,:
::32769->3306/tcp
be4c0dc59f7ae6d3aba26b489efc6fad_tests-wordpress_1   docker-entrypoint.sh apach ...   Up      0.0.0.0:8889->80/tcp,:::8
889->80/tcp
be4c0dc59f7ae6d3aba26b489efc6fad_wordpress_1         docker-entrypoint.sh apach ...   Up      0.0.0.0:8888->80/tcp,:::8
888->80/tcp
sudip@l:~/wp-env/be4c0dc59f7ae6d3aba26b489efc6fad$
```

Unit Test/Integration Requirements

- phpunit/phpunit
- yoast/phpunit-polyfills

Unit/Integration Tests

- wp-env creates isolated enviroments
- wp-env installs WordPress testing library along with the required packages
- use phpunit.xml for configuration
- use .wp-env.json manages WordPress and PHP versions
- use composer.json manages dependencies
- run phpunit

E2E test Requirements

- @playwright/test
- @wordpress/e2e-test-utils-playwright

E2E tests

- wp-env creates isolated enviroments
- wp-env serves app at localhost:8888/8889
- use playwright.config.ts for configuration
- use .wp-env.json manages WordPress and PHP versions
- package.json manages dependencies
- install testing browsers (chromium/firefox/webkit)
- run playwright

Test in multiple WordPress and PHP Versions

- use .github/workflows/main.yml to configure
- push your changes to github

Resources

- https://github.com/WordPress/wordpress-develop
- https://github.com/WordPress/gutenberg
- https://github.com/woocommerce/woocommerce
- https://github.com/elementor/elementor

Why TDD?

- Improves Code Quality
- Self Documentation
- Decreases Technical Debt
- Improves Developer Confidence
- Agile Methodology (CI/CD)
- Happy Customer

Sudip Limbu

Full-Stack Developer
WordPress/PHP/JavaScript

@4slimbu