使用 Slim 為 Legacy Code 重構

Miles @ PHPConf Taiwan 2016 (2016/10/29)

CURRENT

Senior Developer @ 104 Corp.
Volunteer @ DevOps Taiwan

EXPERIENCE

Speaker @ Agile Taichung Meetup 2016 June

TAG

PHP, Docker, DevOps

jangconan@gmail.com
MilesChou

Outline

- Foreword
- Concept
- Practices
- Conclusion

Foreword

很久很久以前...

爆漫王、棋靈王、遊戲王

巴 格 王

巴格王寫程式很快

假如一棵樹在森林裡倒下...

程式問題越來越多

巴格王離職，準備交接


我的 BUG?
想要的話都
給你啊！
去找吧！
我把 BUG
都藏在既
有的程式
碼裡了！


公司
耶一!!
進入了「找 BUG 時代」...
乾 ...

開發人員效率越來越差

https://en.wikipedia.org/wiki/Jenga

開發人員吵著要重構

維運人員準備要上線了


靠有Bug

自己查
自己修
!!


又有


上線!!
!!


我查我修我上！
我查我修我上！

維運人員表示


幹嘛啦？
欸？
寫那什麼鳥程式？還要我修
！
!?
ズ
ビ

維運人員要求重構要完整測過

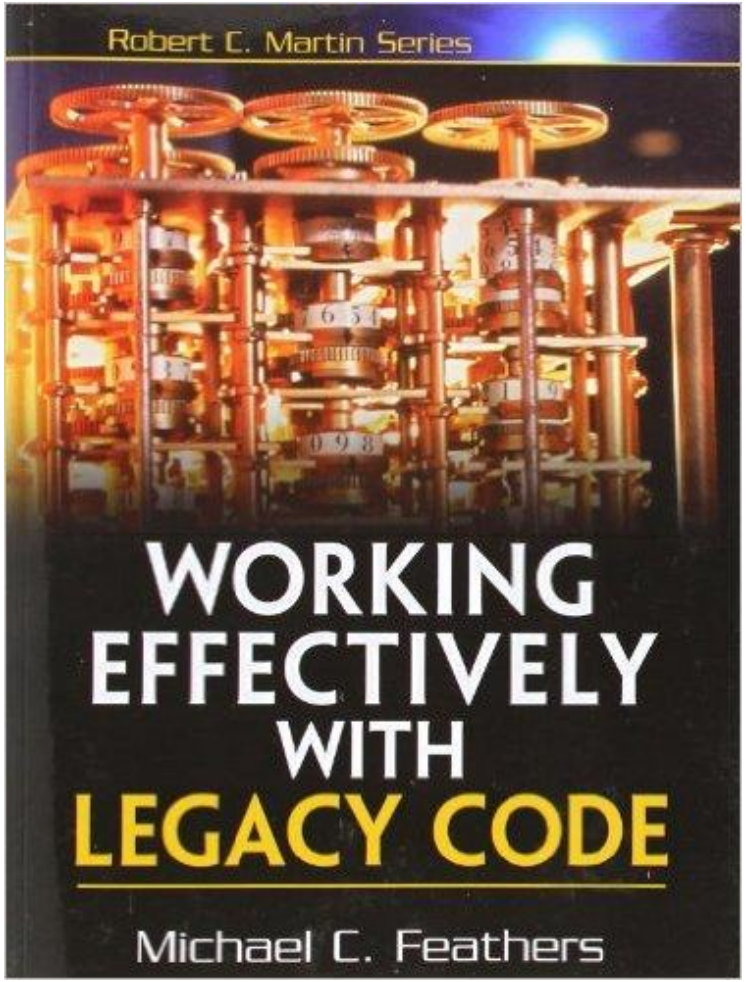

老闆：我好想重構……

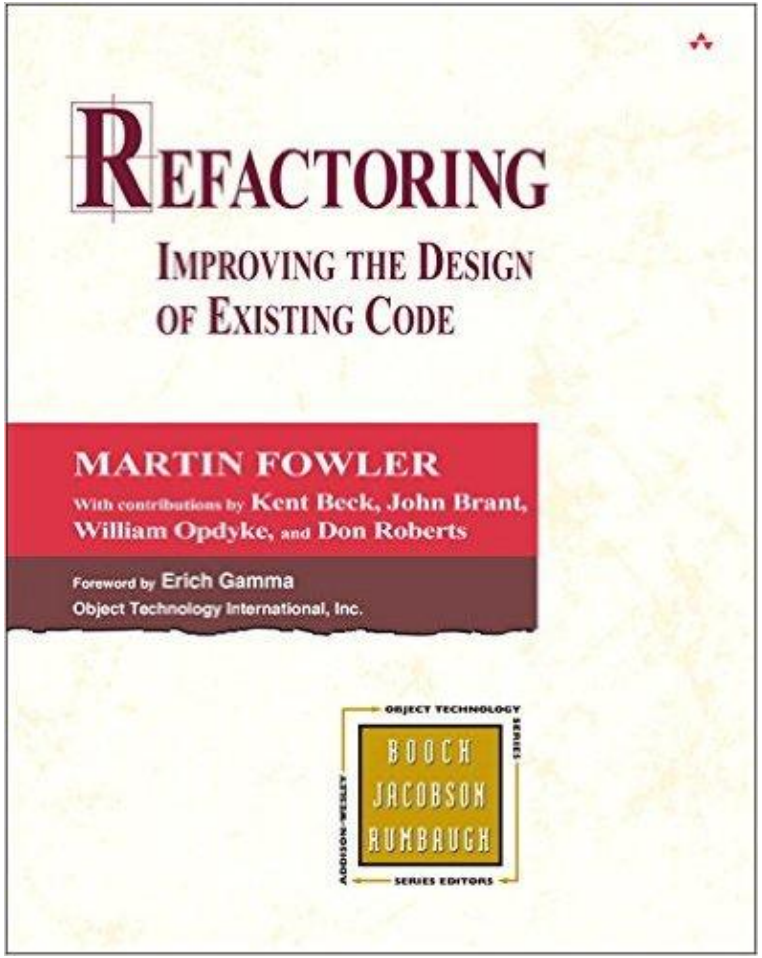

功能先上聽不懂嗎？
啊!?

讓系統繼續爛下去

- 本劇終 -

小劇場如有雷同，純屬巧合


Robert C. Martin Series
WORKING EFFECTIVELY WITH LEGACY CODE
Michael C. Feathers


REFACTORING
IMPROVING THE DESIGN OF EXISTING CODE
MARTIN FOWLER
With contributions by Kent Beck, John Brant, William Opdyke, and Don Roberts
Foreword by Erich Gamma
Object Technology International, Inc.
OBJECT TECHNOLOGY SERIES
BOOCH
JACOBSON
RUMBAUGH
SERIES EDITORS
ADDISON-WESLEY

目標:想一個重構的方法

用健康的心態，面對問題

舊系統需要維護

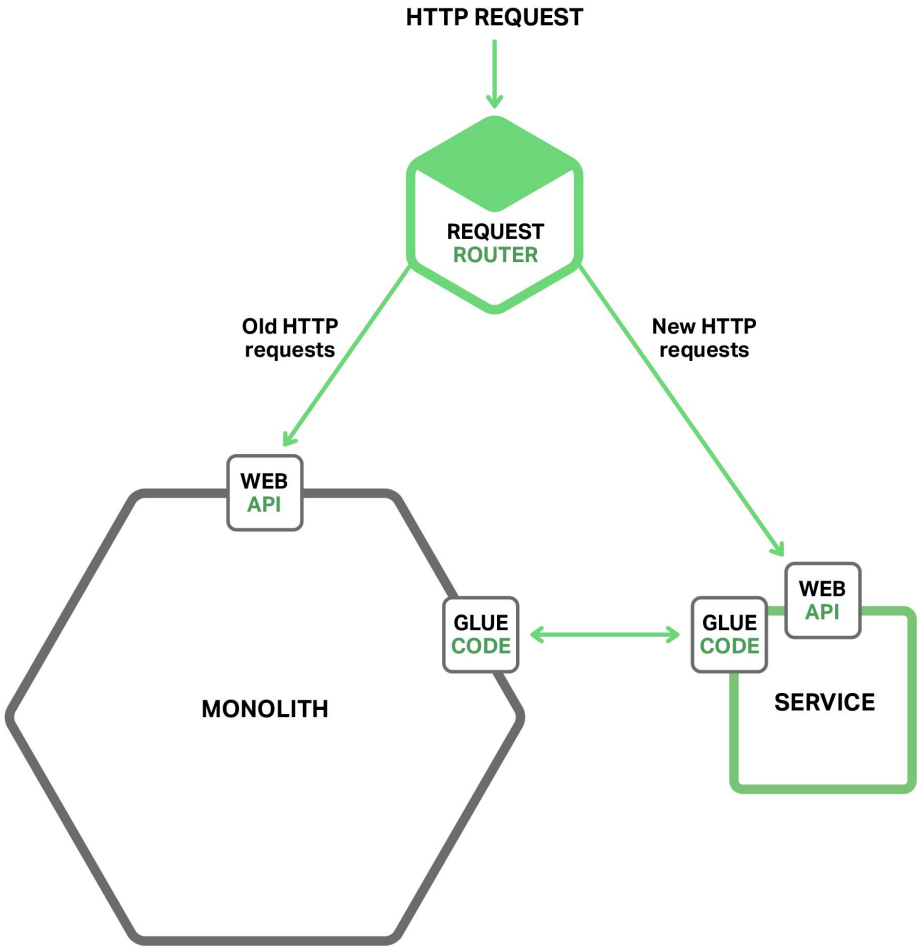
重構會有副作用

技術債必須要還

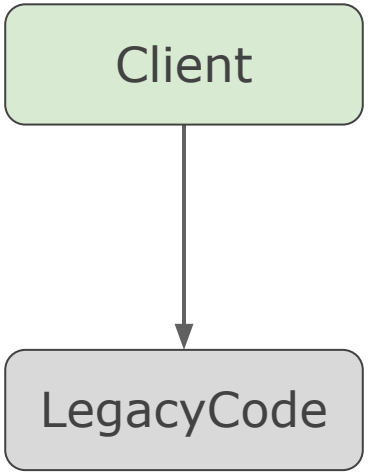
放下各自成見

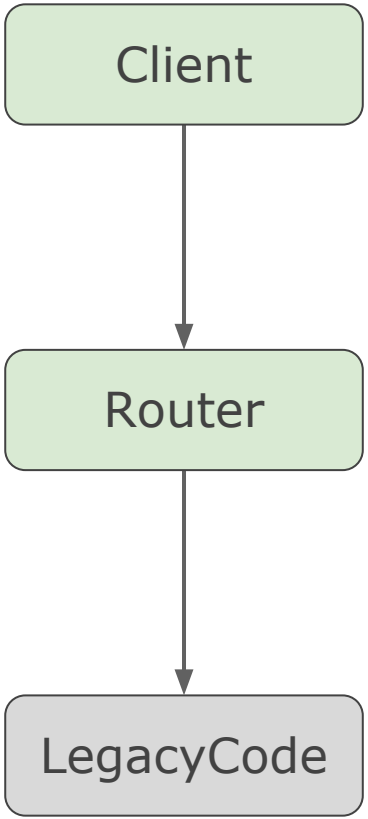
一起解決問題

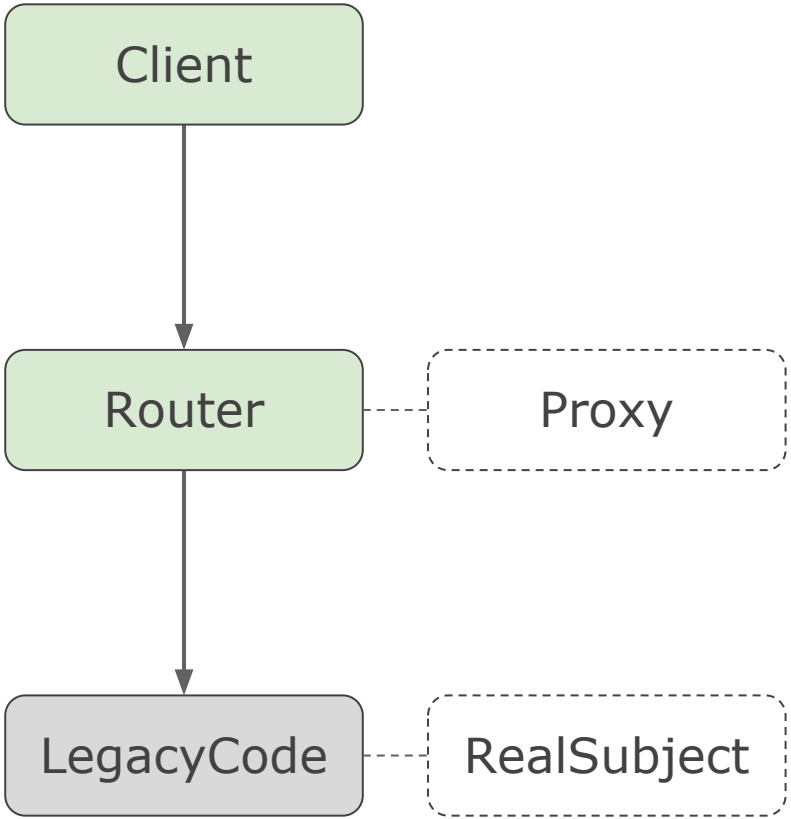
Concept



https://www.nginx.com/?p=28286


Client
LegacyCode

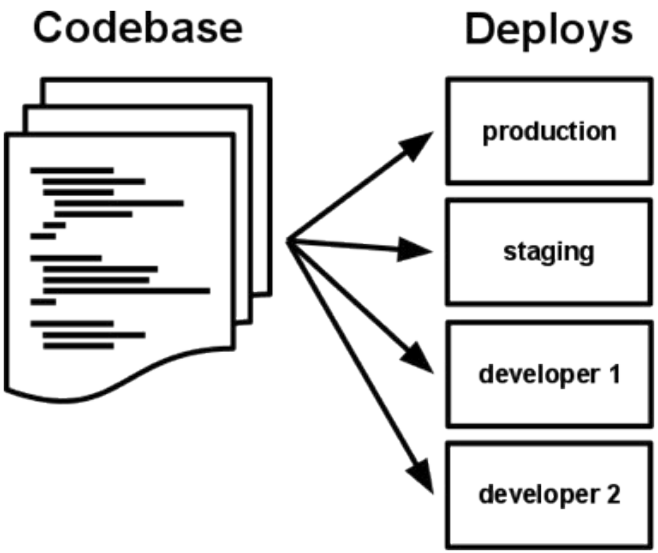

Client
Router
LegacyCode


Client
Router
Proxy
LegacyCode
RealSubject

Summary #1

- 新功能儘可能不要寫在 legacy code 裡
- 實作 proxy pattern，讓程式先正常運作

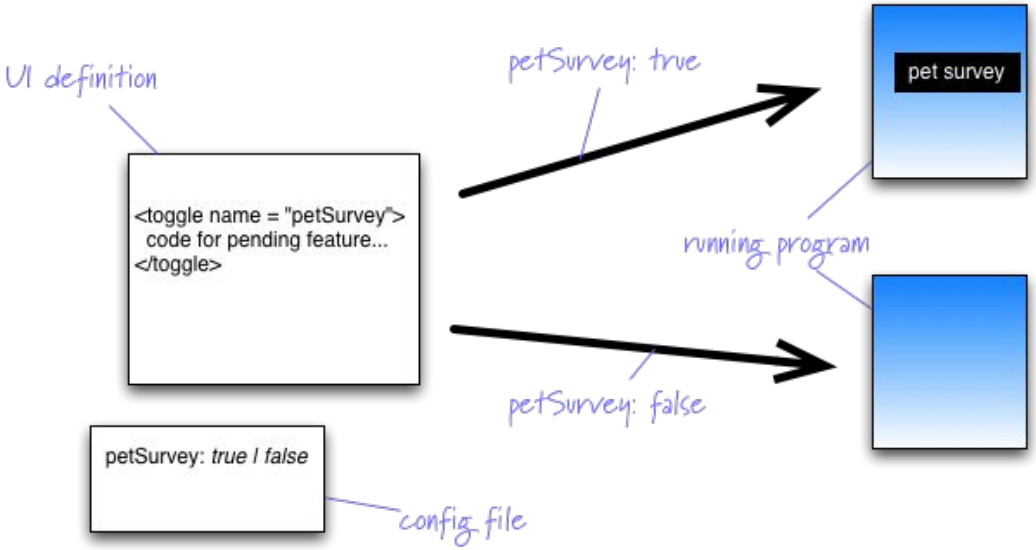
References - From 12 Factors



Summary #2

- 新功能儘可能不要寫在 legacy code 裡
- 實作 proxy pattern，讓程式先正常運作
- **程式碼放在同個 repository 裡，用 branch 切換**

References - From Martin Fowler



Summary #3

- 新功能儘可能不要寫在 legacy code 裡
- 實作 proxy pattern，讓程式先正常運作
- 程式碼放在同個 repository 裡，用 branch 切換
- **feature toggle = 程式實作類似 branch 的功能**
- **重構時間太長，可以用 feature toggle 的類 branch 切換**

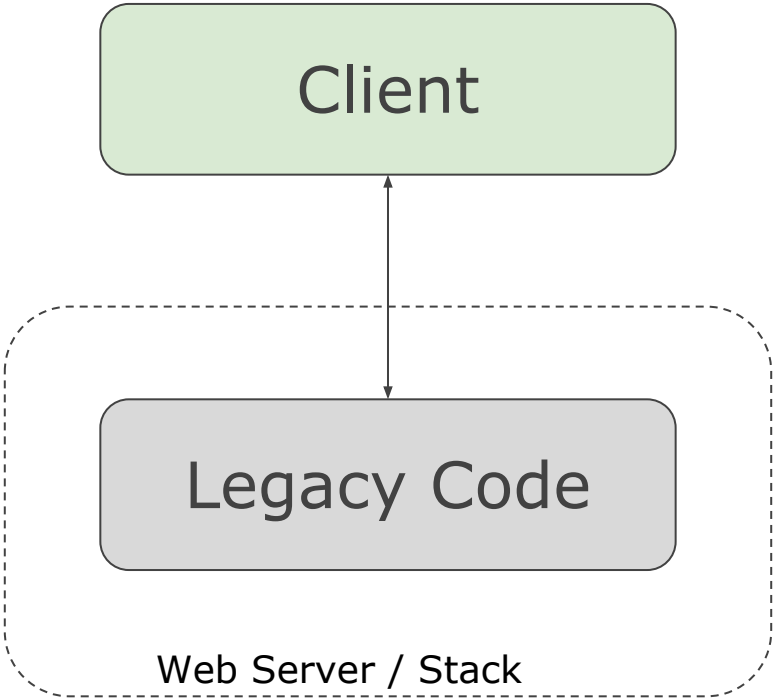
Summary #3

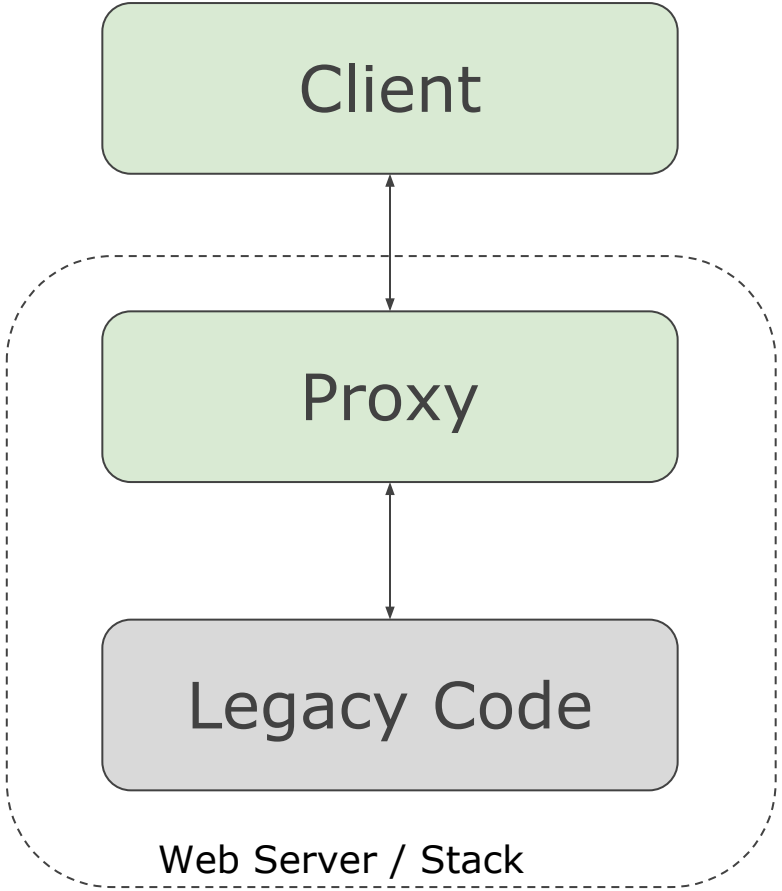
- 新功能儘可能不要寫在 legacy code 裡
- 實作 proxy pattern，讓程式先正常運作
- 程式碼放在同個 repository 裡，用 **branch 切換**
- feature toggle = 程式實作類似 branch 的功能
- 重構時間太長，可以用 feature toggle 的類 **branch 切換**

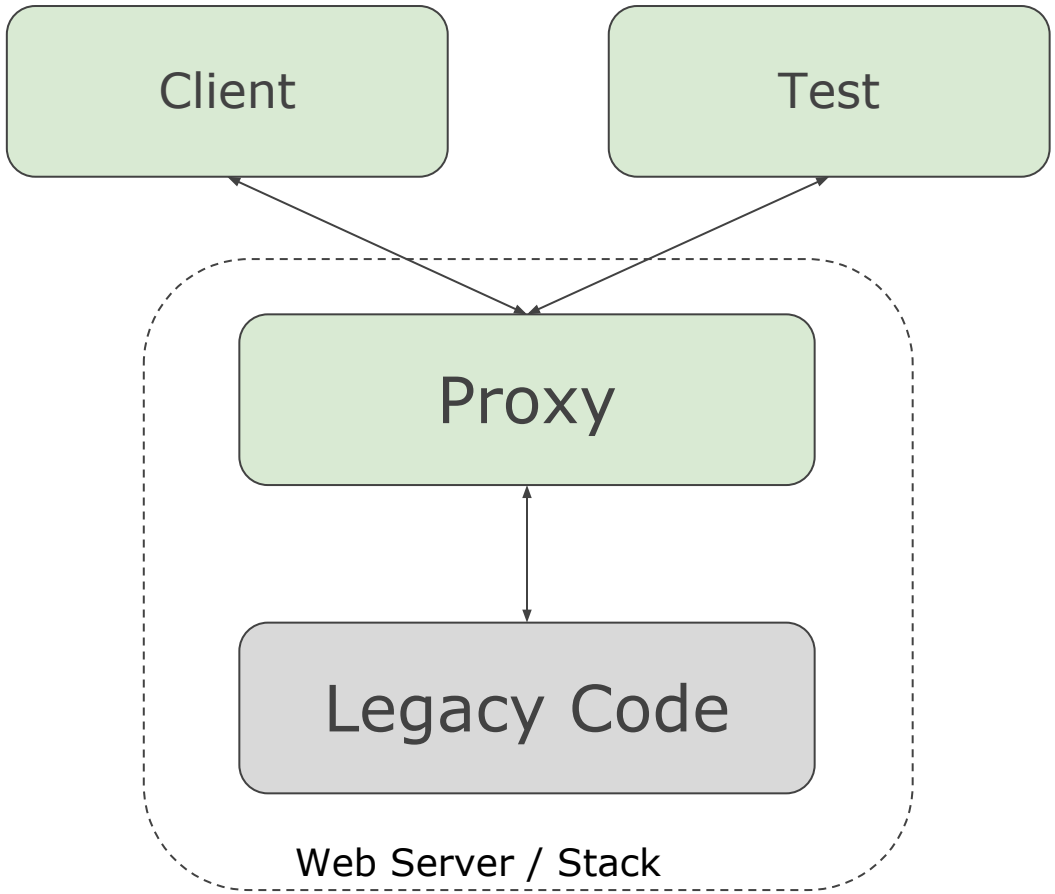
Summary #4

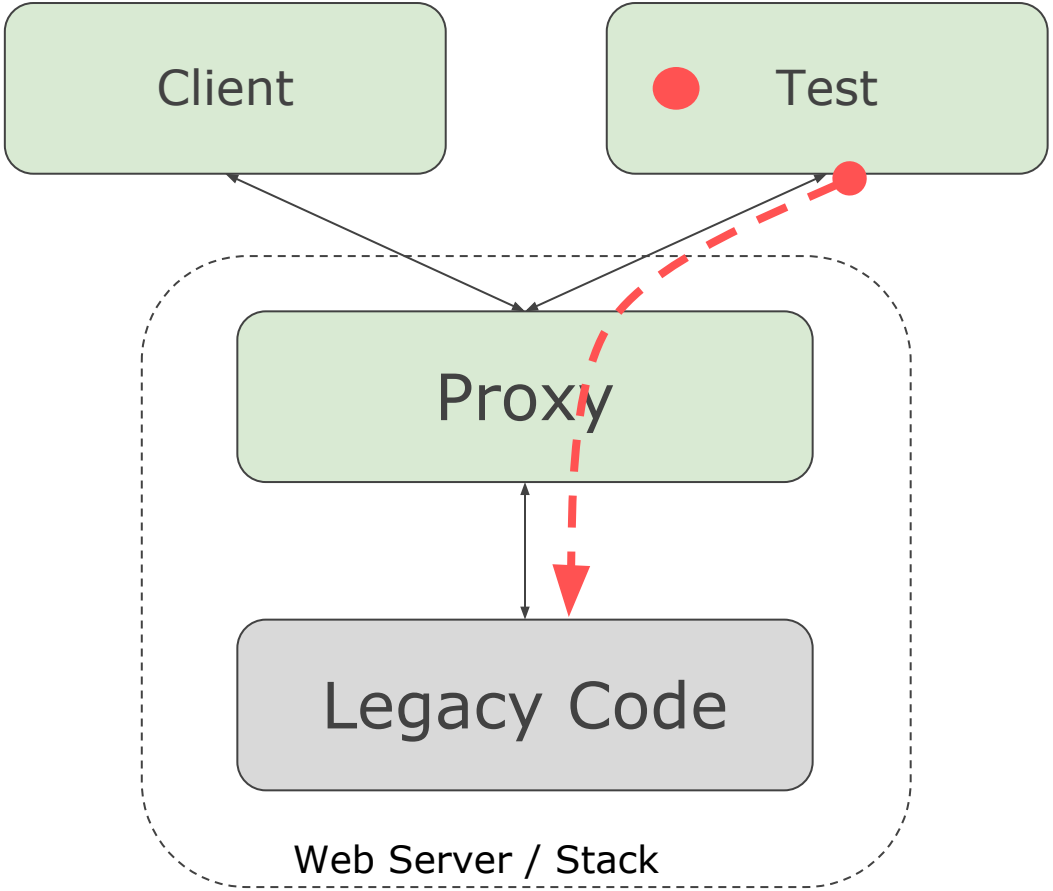
- 新功能儘可能不要寫在 legacy code 裡
- 實作 proxy pattern，讓程式先正常運作
- ~~程式碼放在同個 repository 裡，用 branch 切換~~
- ~~feature toggle = 程式實作類似 branch 的功能~~
- ~~重構時間太長，可以用 feature toggle 的類 branch 切換~~
- **程式碼放在同個 branch，用 feature toggle 切換**

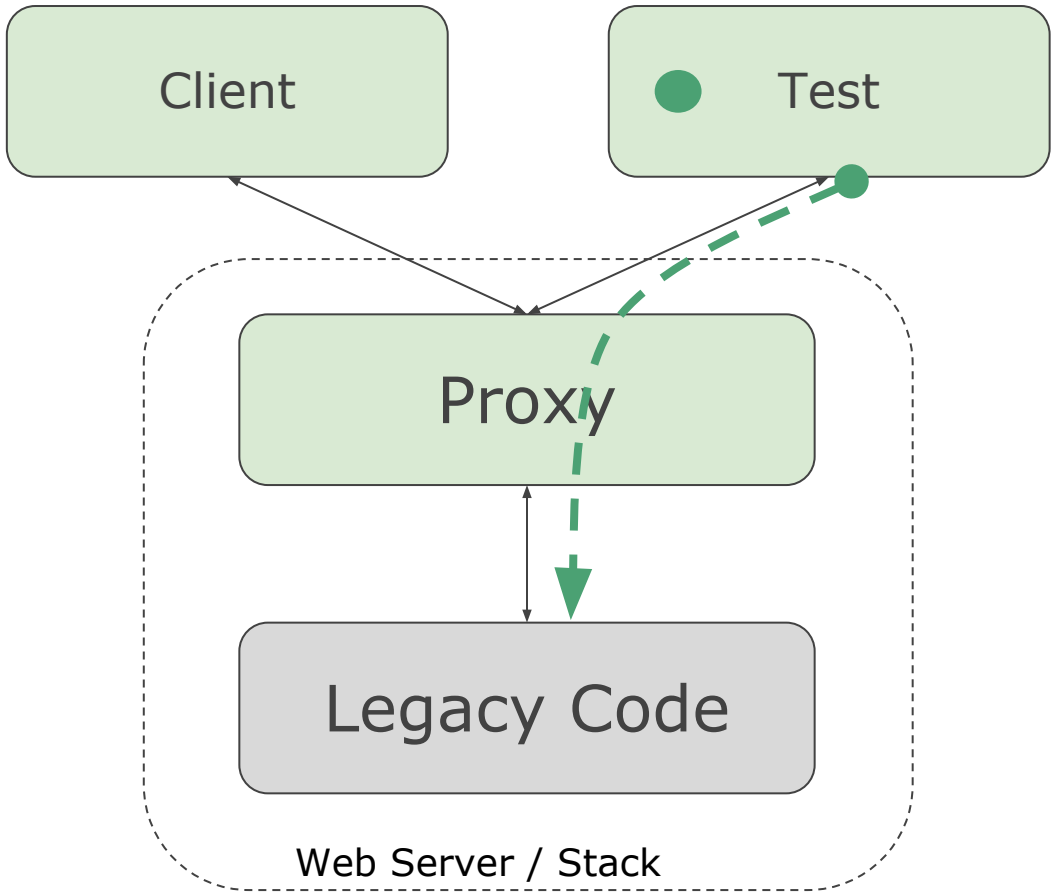
Refactoring Flow

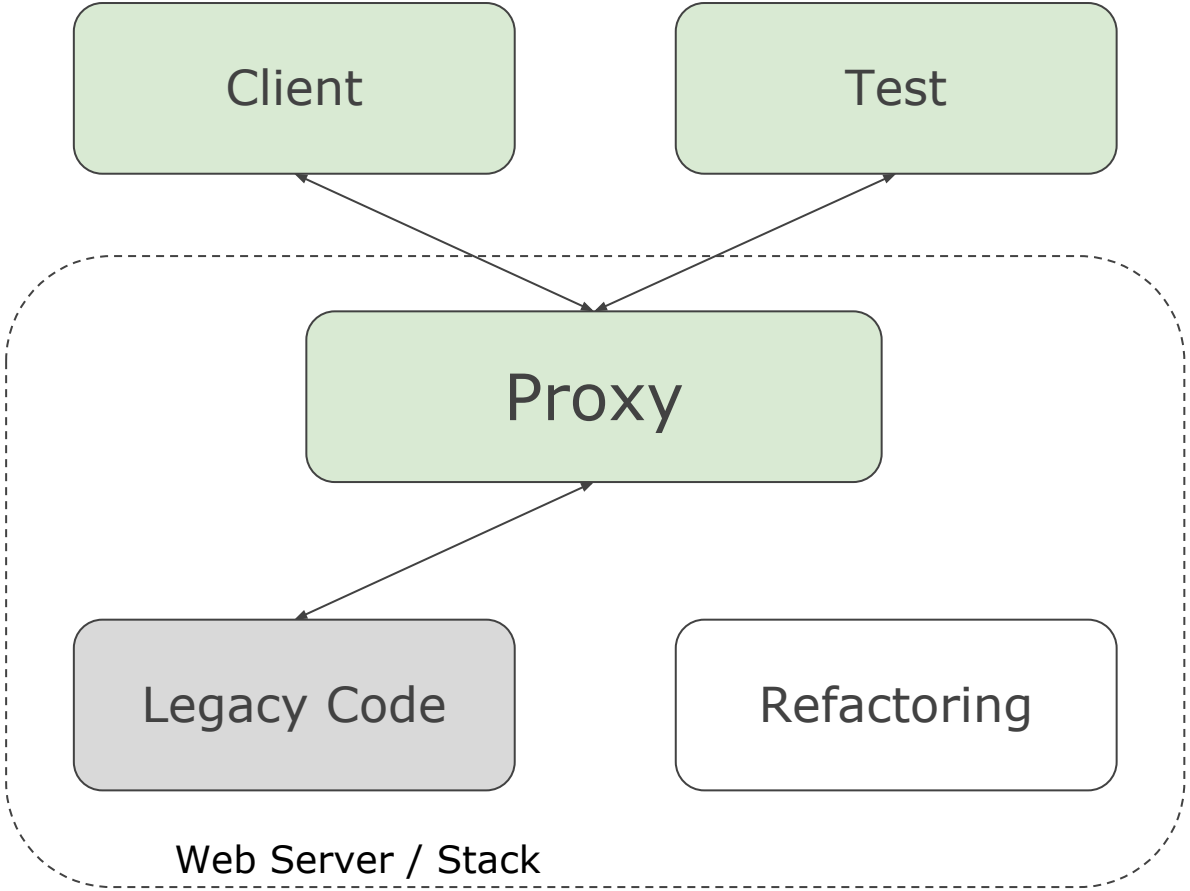

Client
Legacy Code
Web Server / Stack

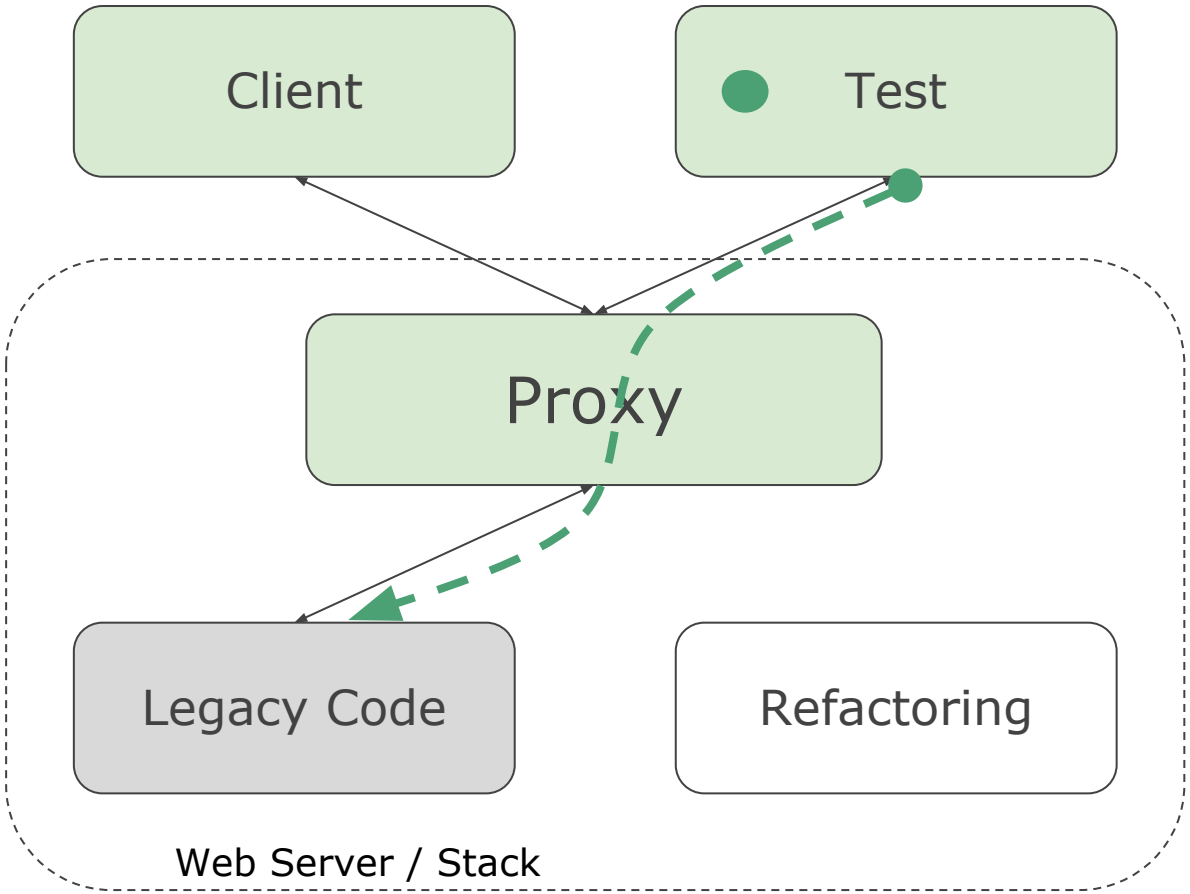

Client
Proxy
Legacy Code
Web Server / Stack

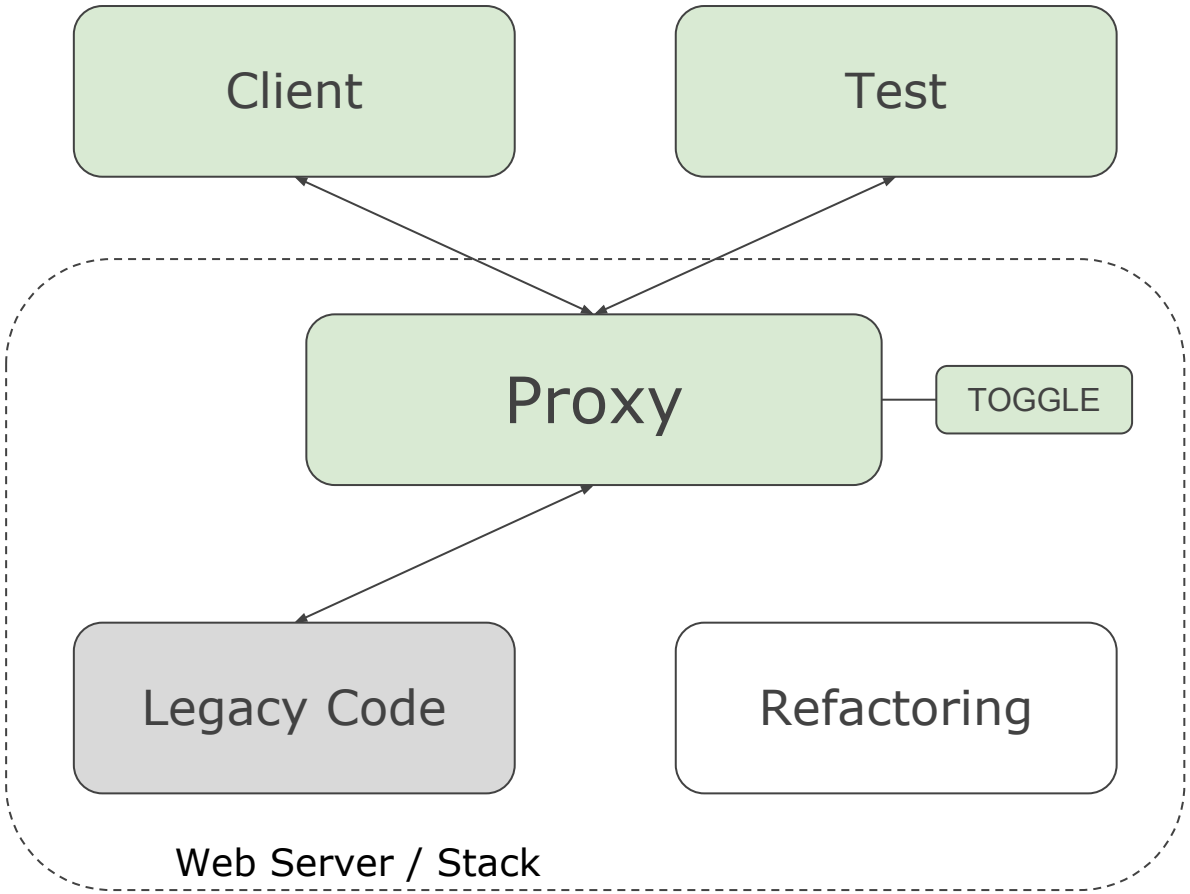

Client
Test
Proxy
Legacy Code
Web Server / Stack

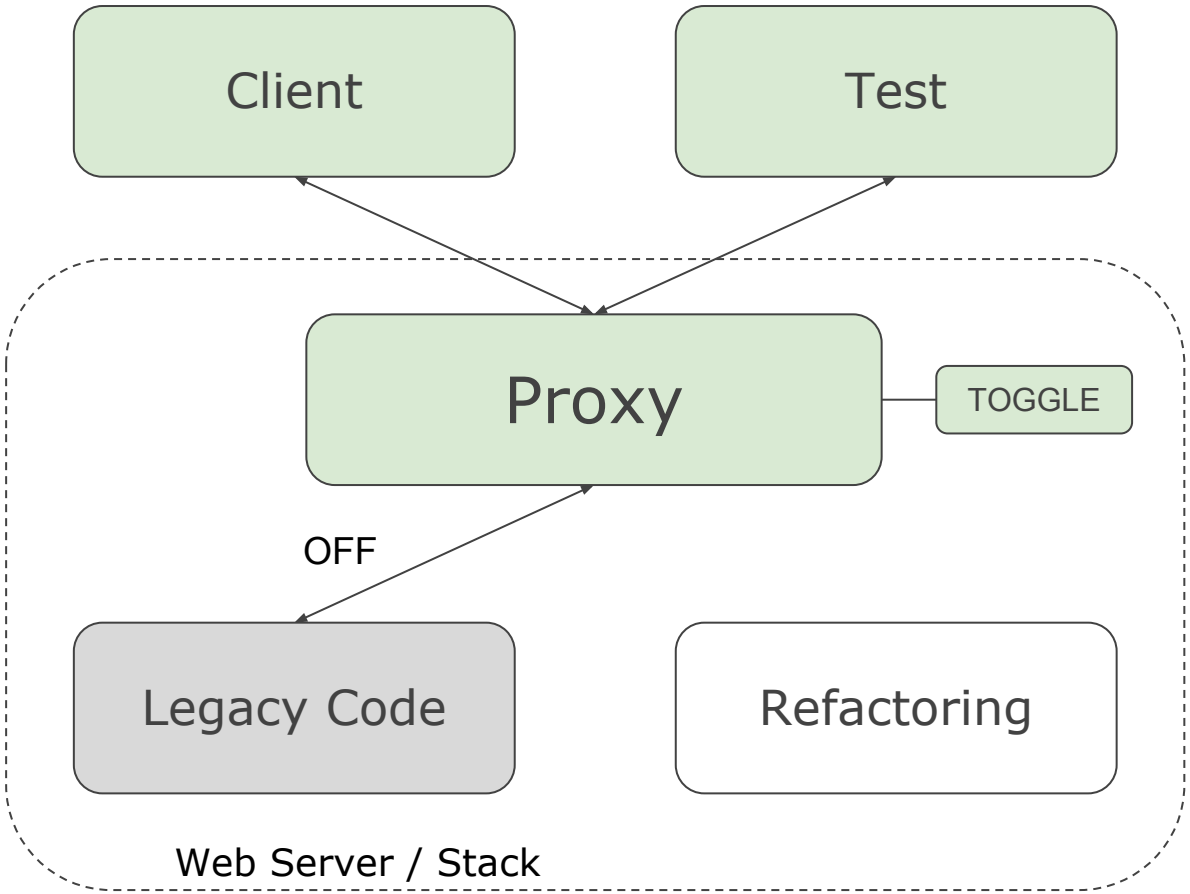

Client
Test
Proxy
Legacy Code
Web Server / Stack

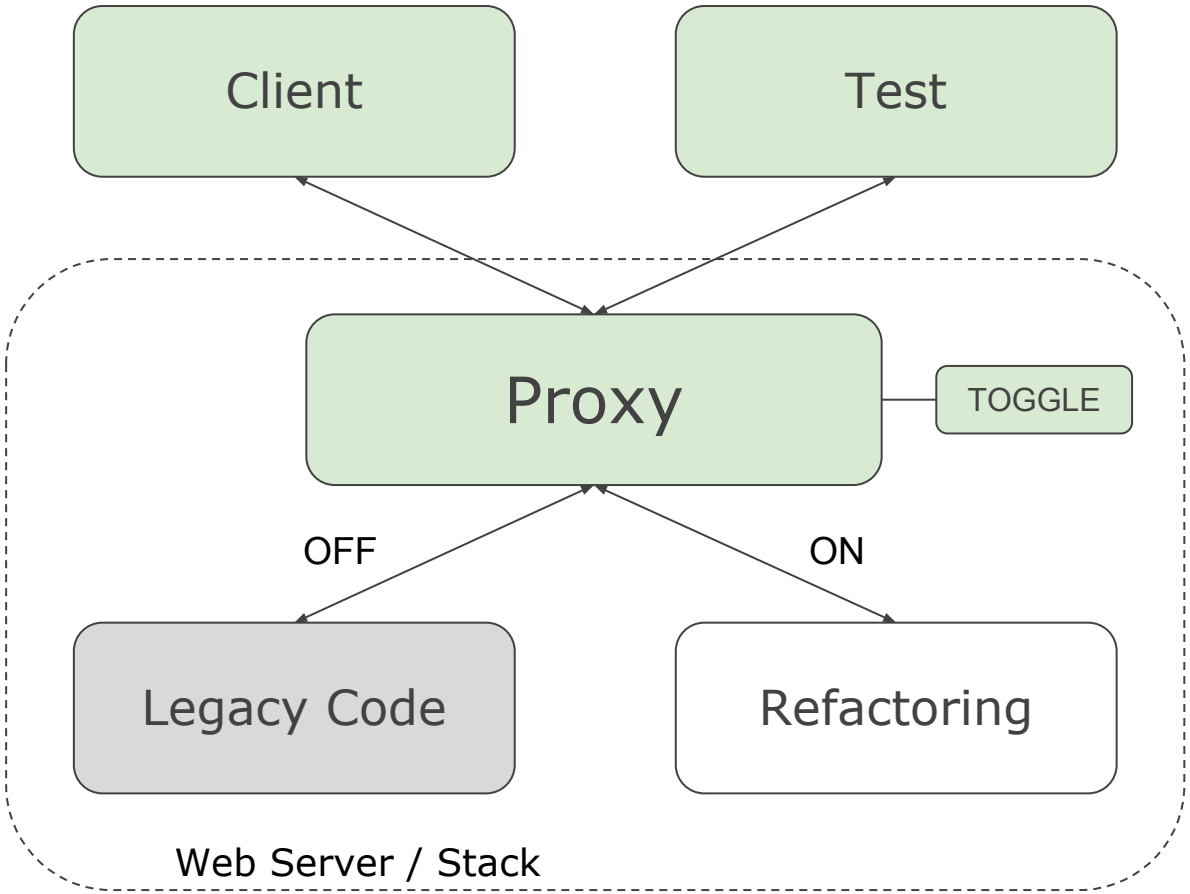

Client
Test
Proxy
Legacy Code
Web Server / Stack

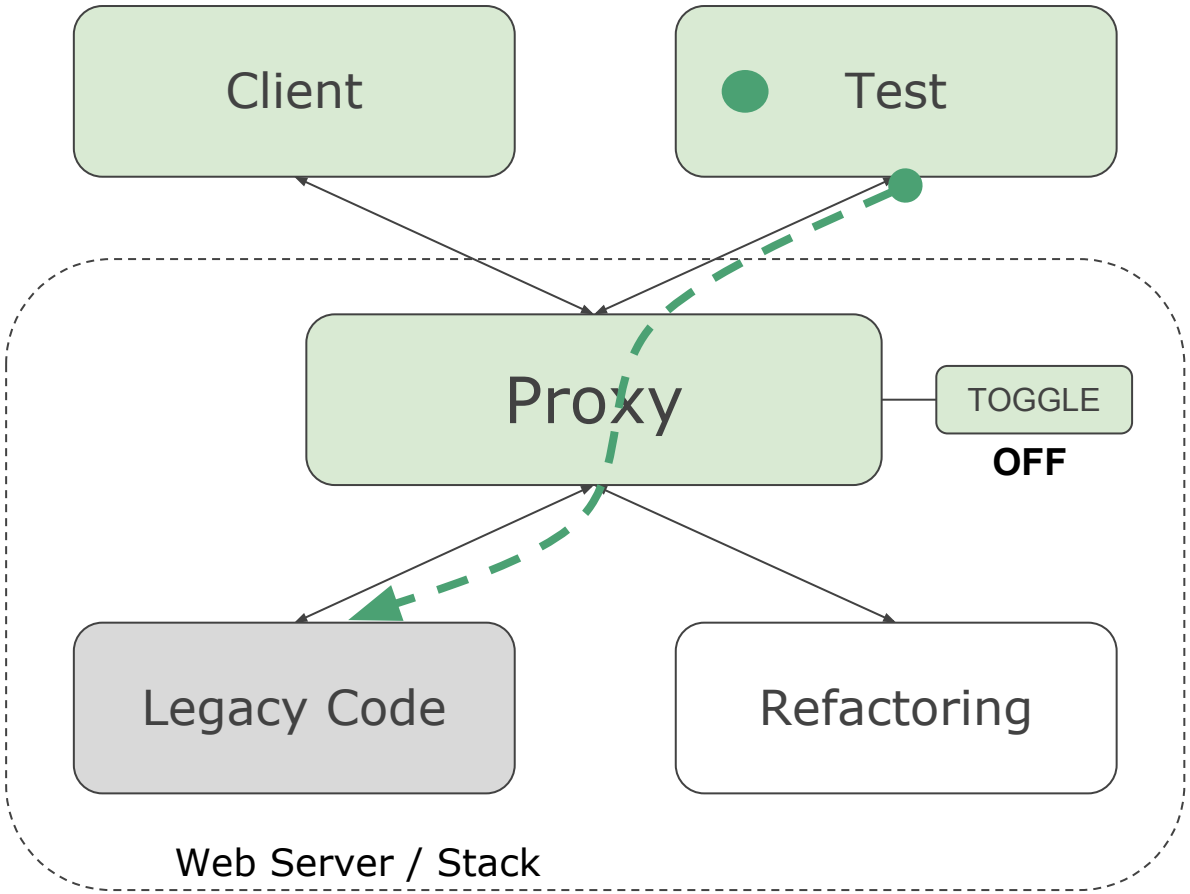

Client
Test
Proxy
Legacy Code
Refactoring
Web Server / Stack

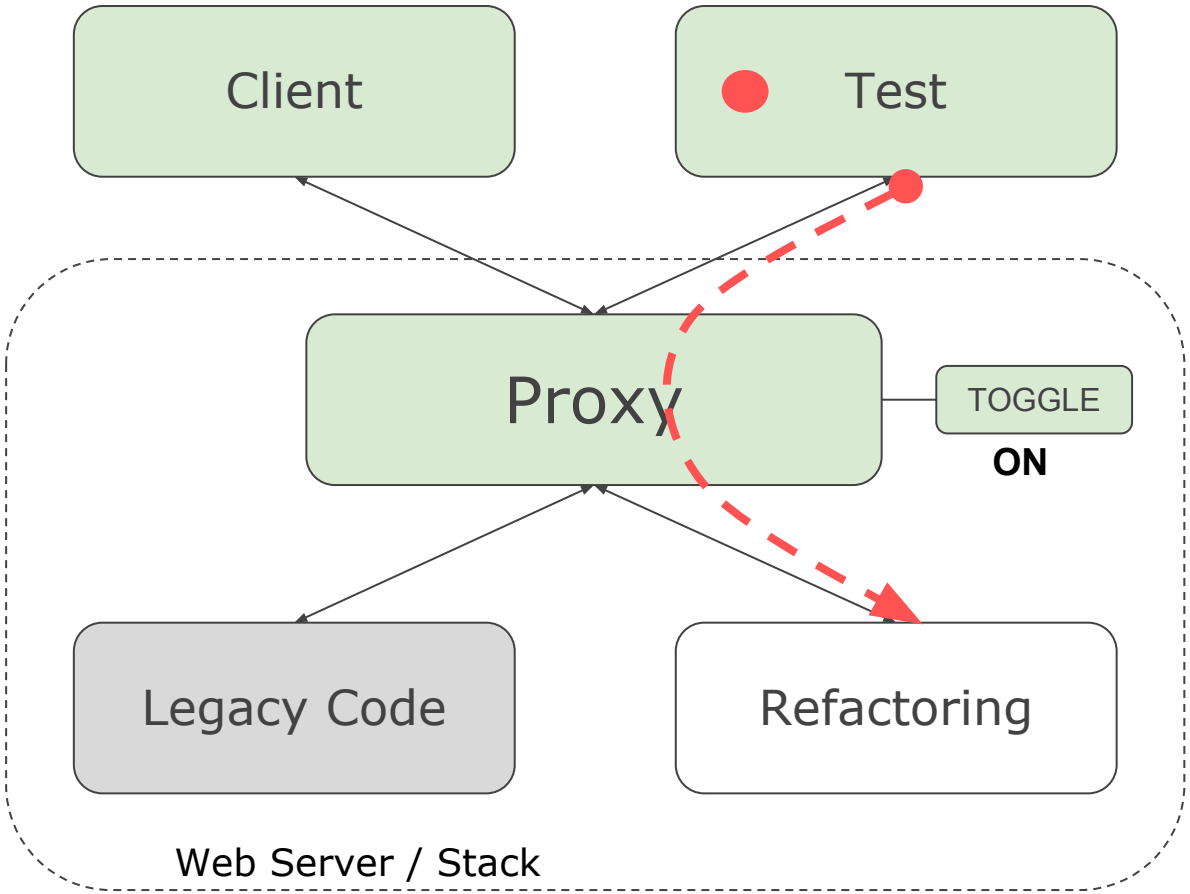

Client
Test
Proxy
Legacy Code
Refactoring
Web Server / Stack

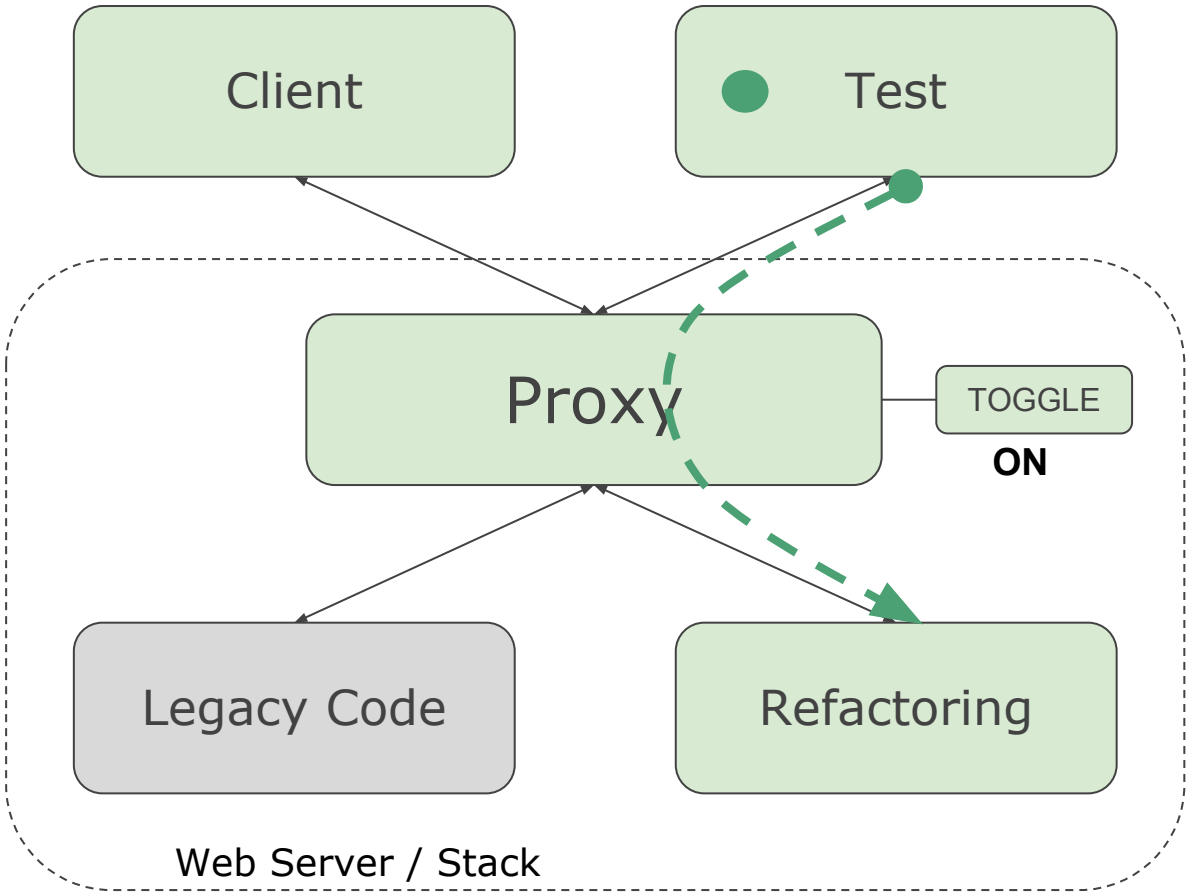

Client
Test
Proxy
TOGGLE
Legacy Code
Refactoring
Web Server / Stack

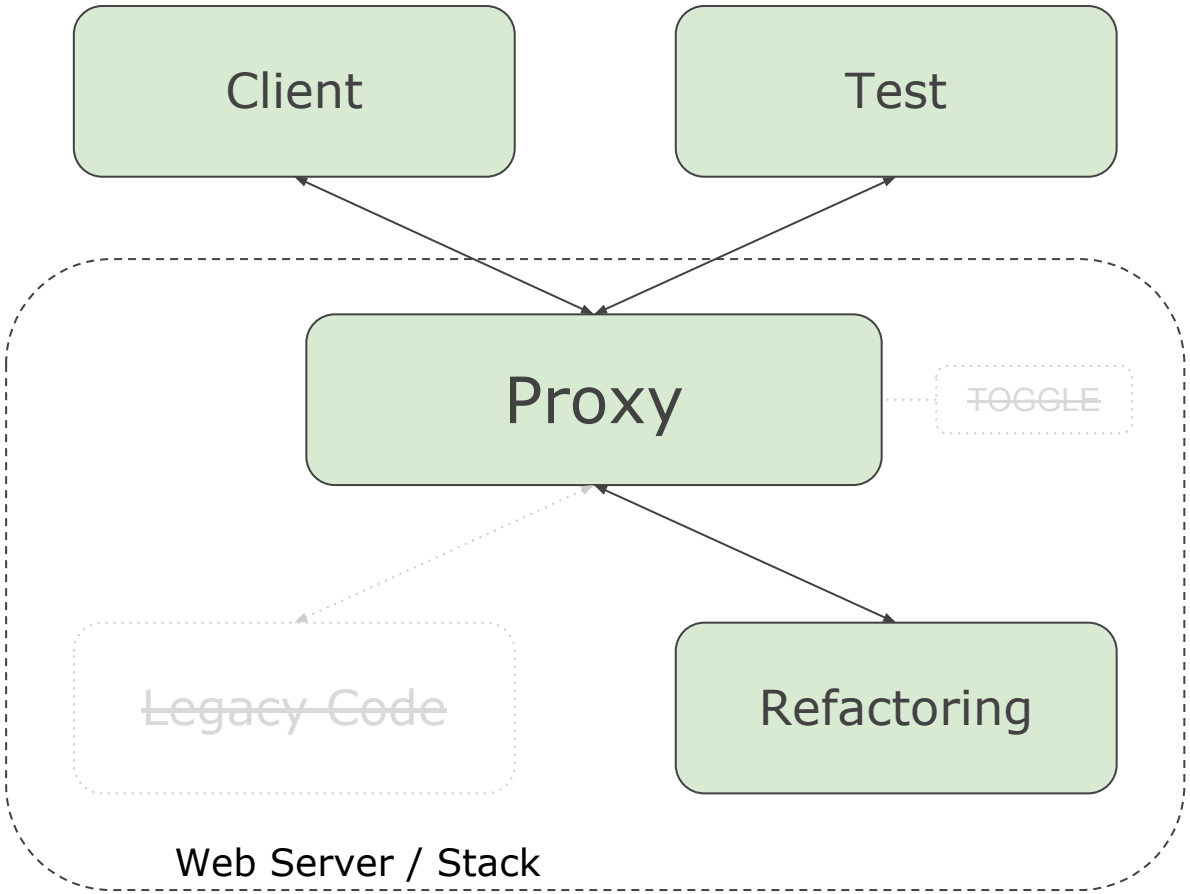

Client
Test
Proxy
TOGGLE
OFF
Legacy Code
Refactoring
Web Server / Stack

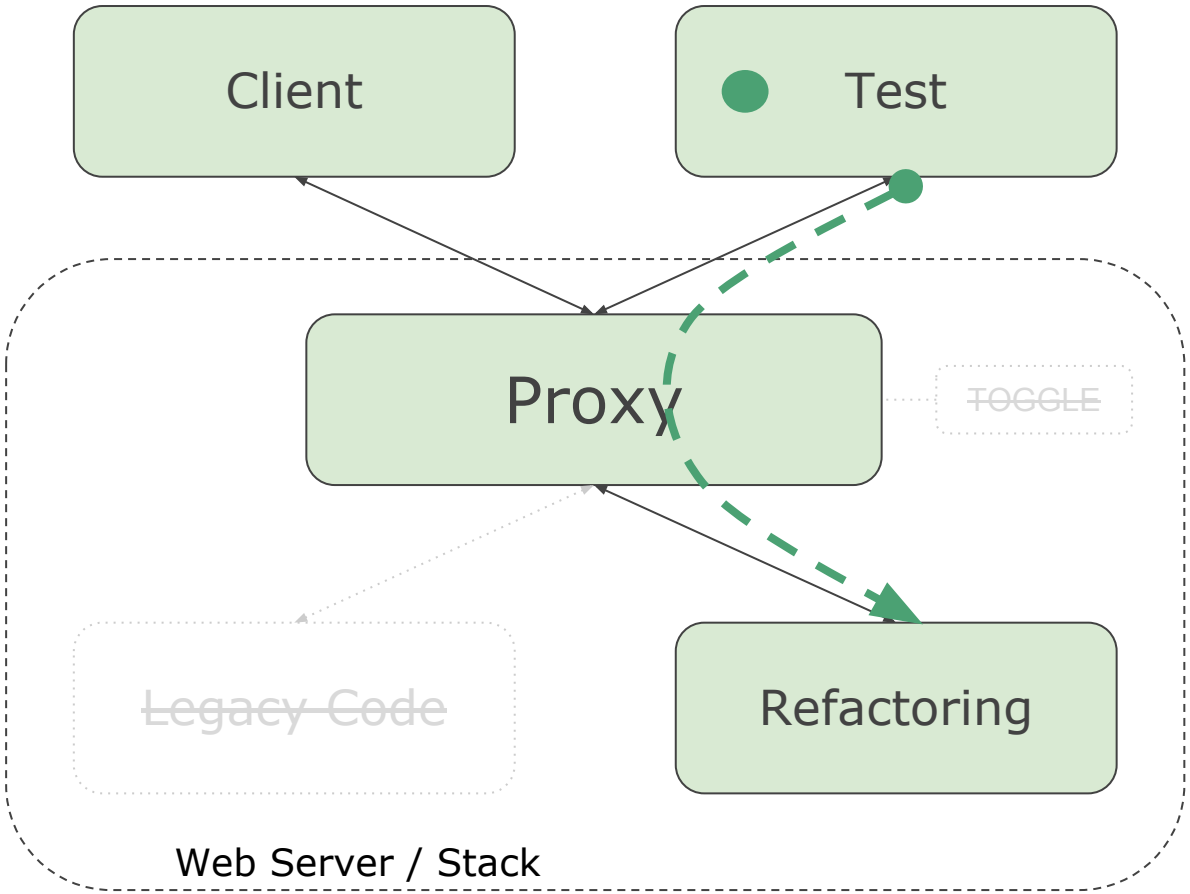

Client
Test
Proxy
TOGGLE
OFF
ON
Legacy Code
Refactoring
Web Server / Stack

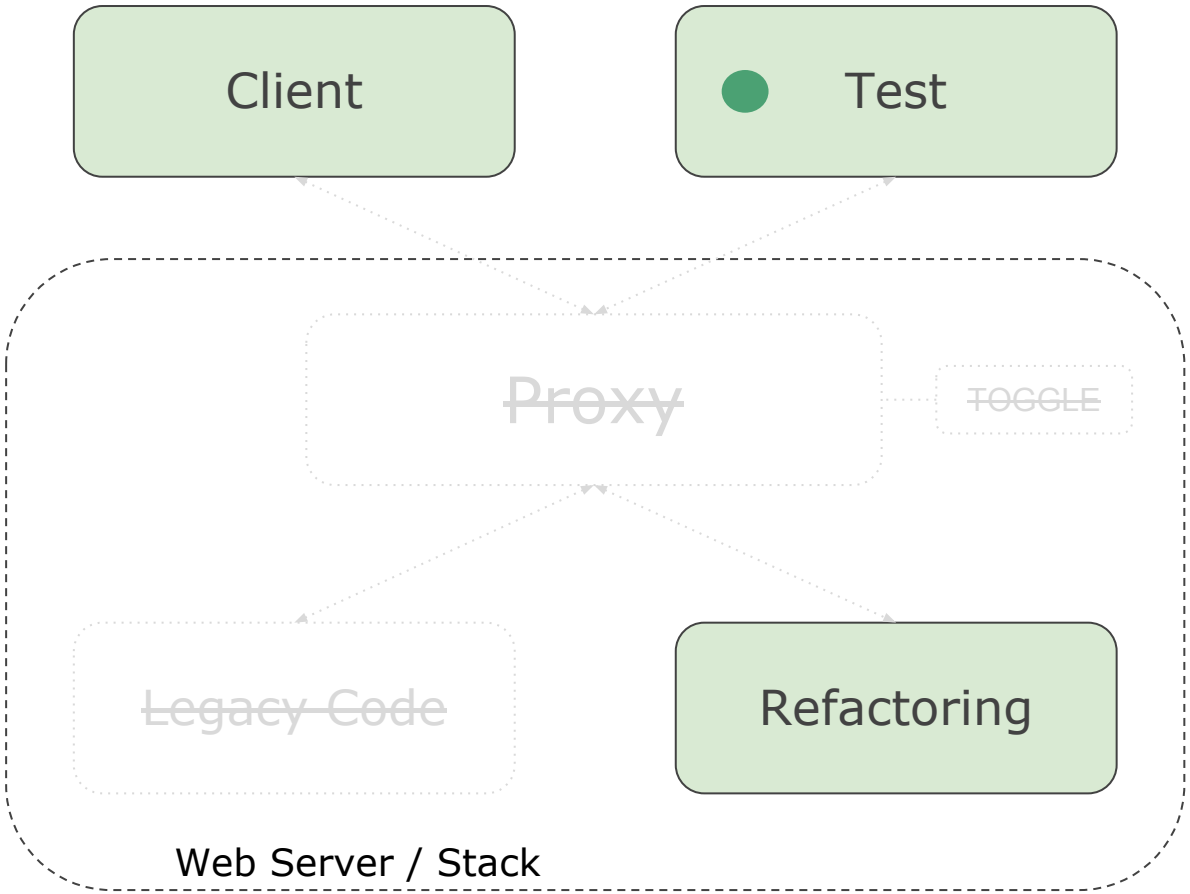

Client
Test
Proxy
TOGGLE
OFF
Legacy Code
Refactoring
Web Server / Stack

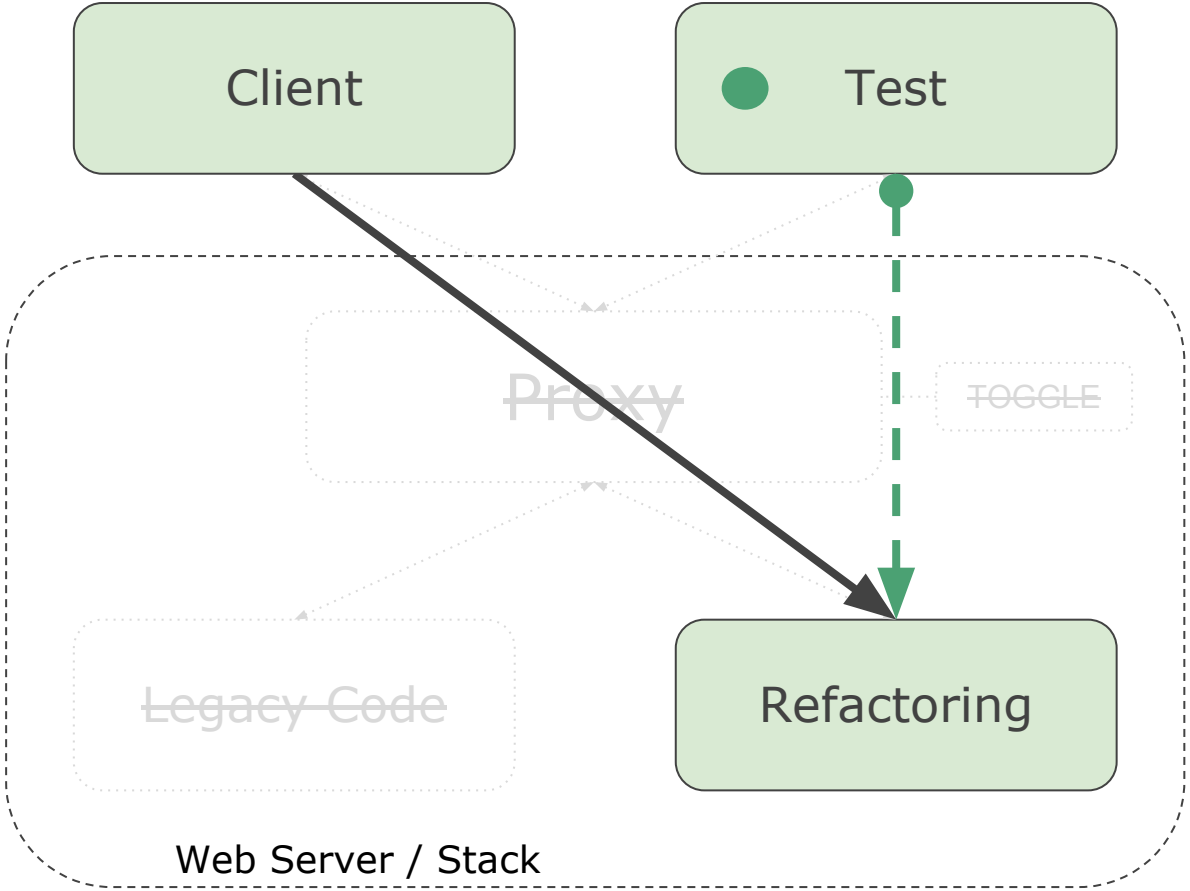

Client
Test
Proxy
TOGGLE
ON
Legacy Code
Refactoring
Web Server / Stack


Client
Test
Proxy
TOGGLE
ON
Legacy Code
Refactoring
Web Server / Stack


Client
Test
Proxy
TOGGLE
Legacy Code
Refactoring
Web Server / Stack


Client
Test
Proxy
TOGGLE
Legacy Code
Refactoring
Web Server / Stack


Client
Test
Proxy
TOGGLE
Legacy Code
Refactoring
Web Server / Stack


Client
Test
Proxy
TOGGLE
Legacy Code
Refactoring
Web Server / Stack

Requirement of Proxy

- Routable
- Easy to setting route
- Lightweight

Why Slim

Slim Feature

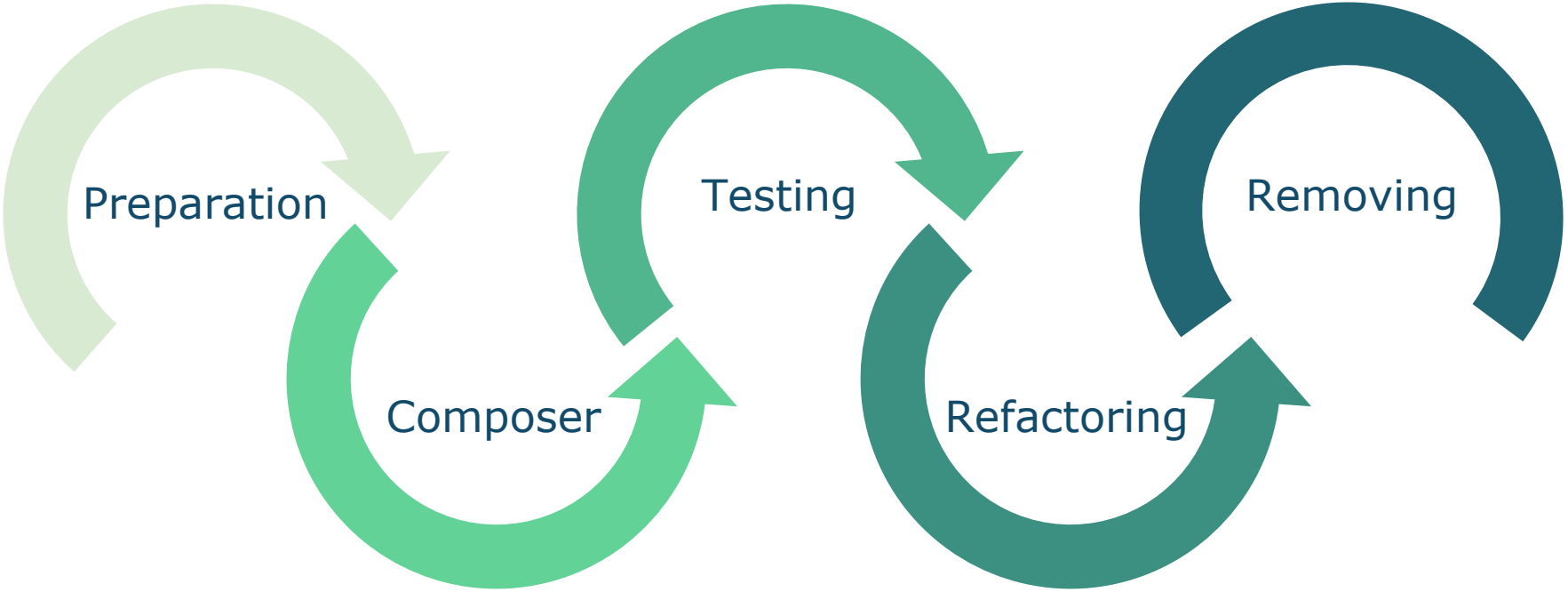
- Routable
- Easy to setting route
- Lightweight
- **Less dependence**
- **Easy to embedding**

The other PHP Micro Framework

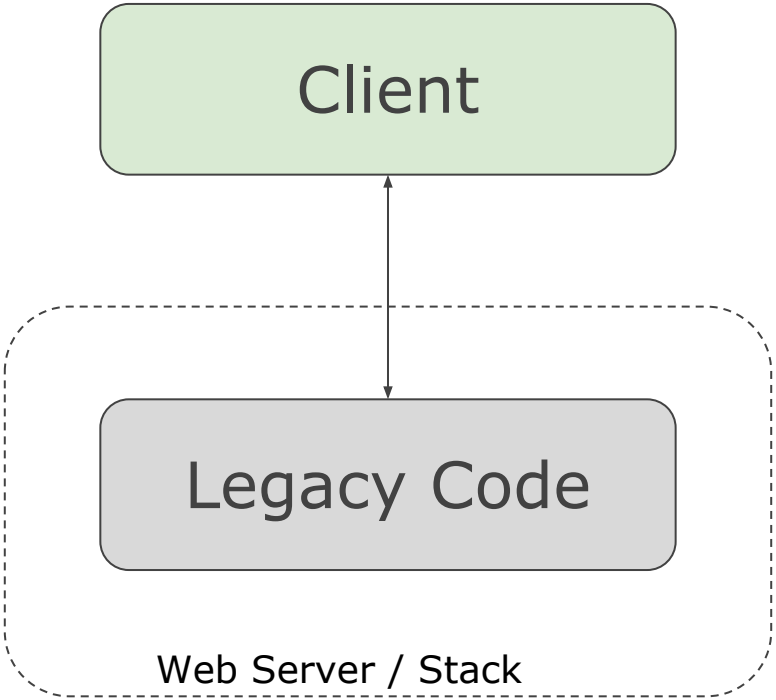
- Lumen
- Phalcon (Micro)
- ...

Practices

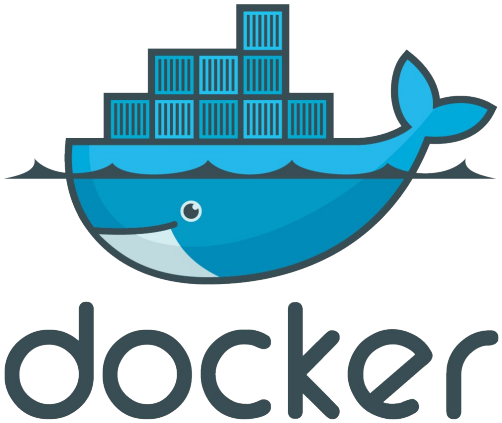
Process



#1 Preparation


Client
Legacy Code
Web Server / Stack

Dependency Services



Database Migration & Seeding

Phinx

Propel

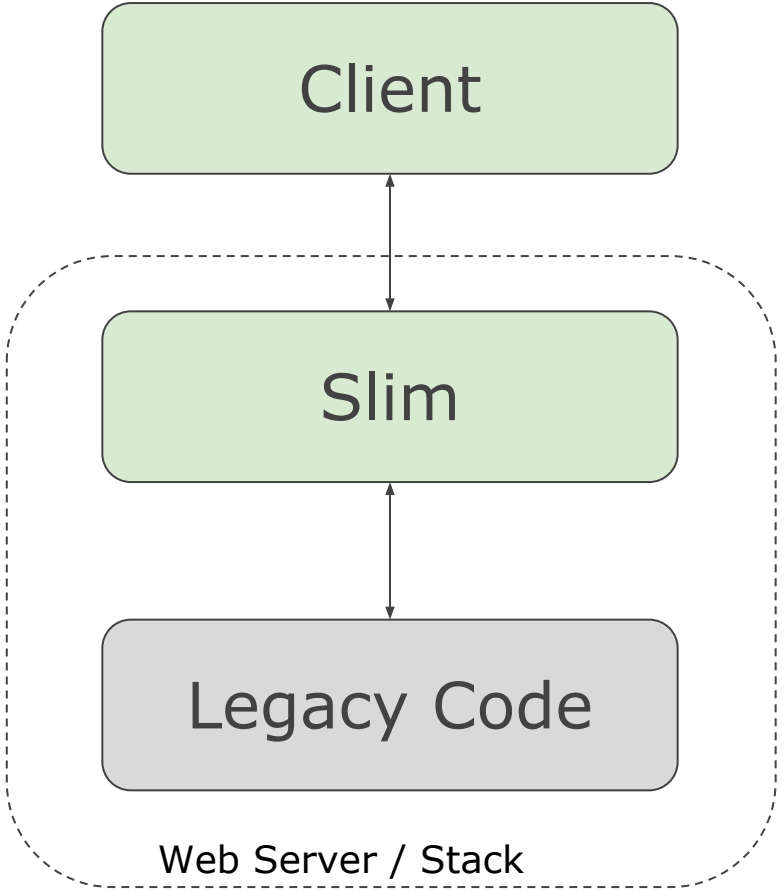
Doctrine

Router Setting

nginx.conf

apache.conf
.htaccess

#2 Composer


Client
Slim
Legacy Code
Web Server / Stack

Initial Composer & Slim

```
$ cd /path/to/project/
$ composer init
$ composer require slim/slim
```

/path/to/project/public/index.php

```php
<?php
use \Psr\Http\Message\ServerRequestInterface as Request;
use \Psr\Http\Message\ResponseInterface as Response;

require 'vendor/autoload.php';

$app = new \Slim\App;
$app->get('/hello/{name}', function (Request $request, Response $response) {
    $name = $request->getAttribute('name');
    $response->getBody()->write("Hello, $name");

    return $response;
});
$app->run();
```

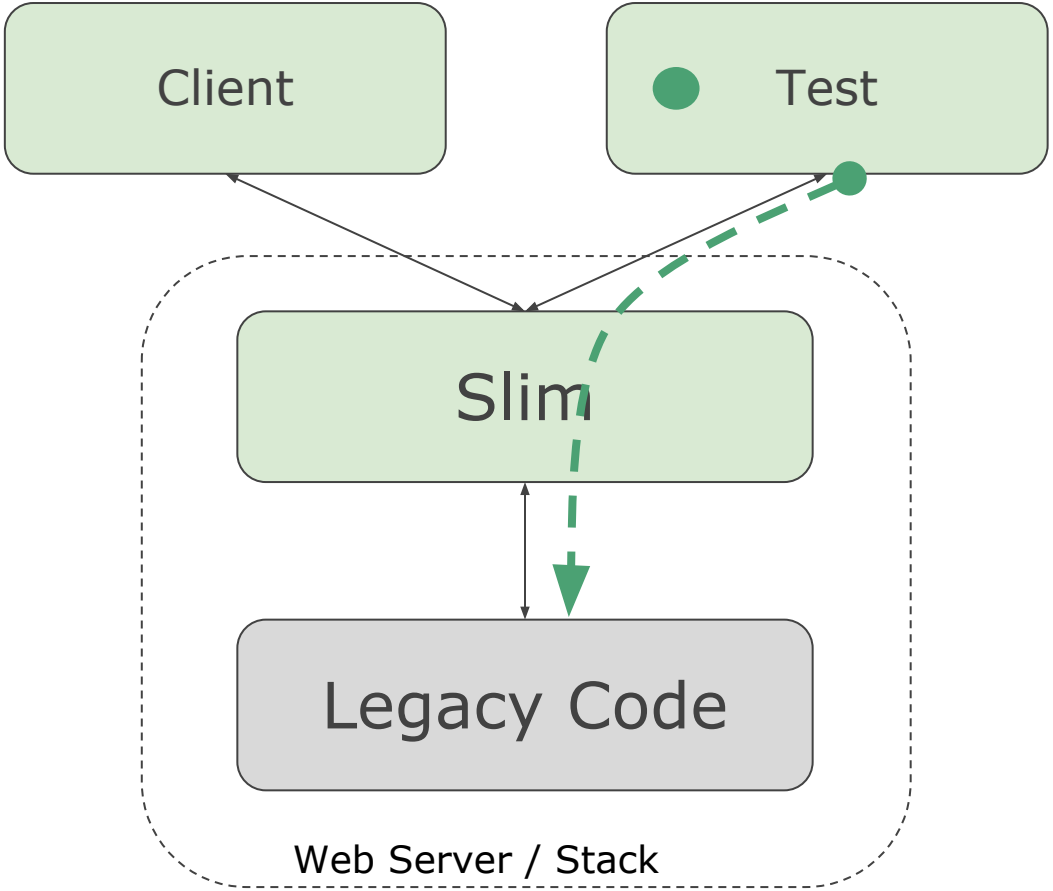
注意:本原始碼 啟動方法可參考 Slim 官方網站說明, 請視專案實際佈署方法調整相關參數。

```php
// http://localhost/admin.php
$app->any('/admin.php', function (Request $request, Response $response) {
   require __DIR__ . '/../admin.php';
});


// http://localhost/
$app->any('/{all:.*}', function (Request $request, Response $response) {
   require __DIR__ . '/../index.php';
});
```

ZEND FRAMEWORK

```php
// All route pattern
$app->any('/{all:.*}', function (Request $request, Response $response) {

    // Run Zend Framework application
    $application = new Zend_Application(
        APPLICATION_ENV,
        APPLICATION_PATH . '/configs/application.ini'
    );
    $application->bootstrap()->run();
});
```

#3 Testing


Client
Test
Slim
Legacy Code
Web Server / Stack

Codeception

```php
$I = new AcceptanceTester($scenario);
$I->wantTo('test admin page and test login');

$I->amOnPage('/admin');
$I->see('Password');
$I->fillField('password', '0000');
$I->click('登入');
$I->see('Logout');
```

+

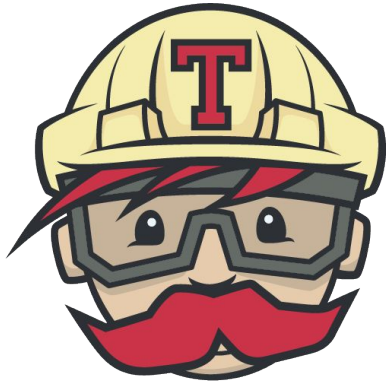
Codeception

Selenium

CI Server

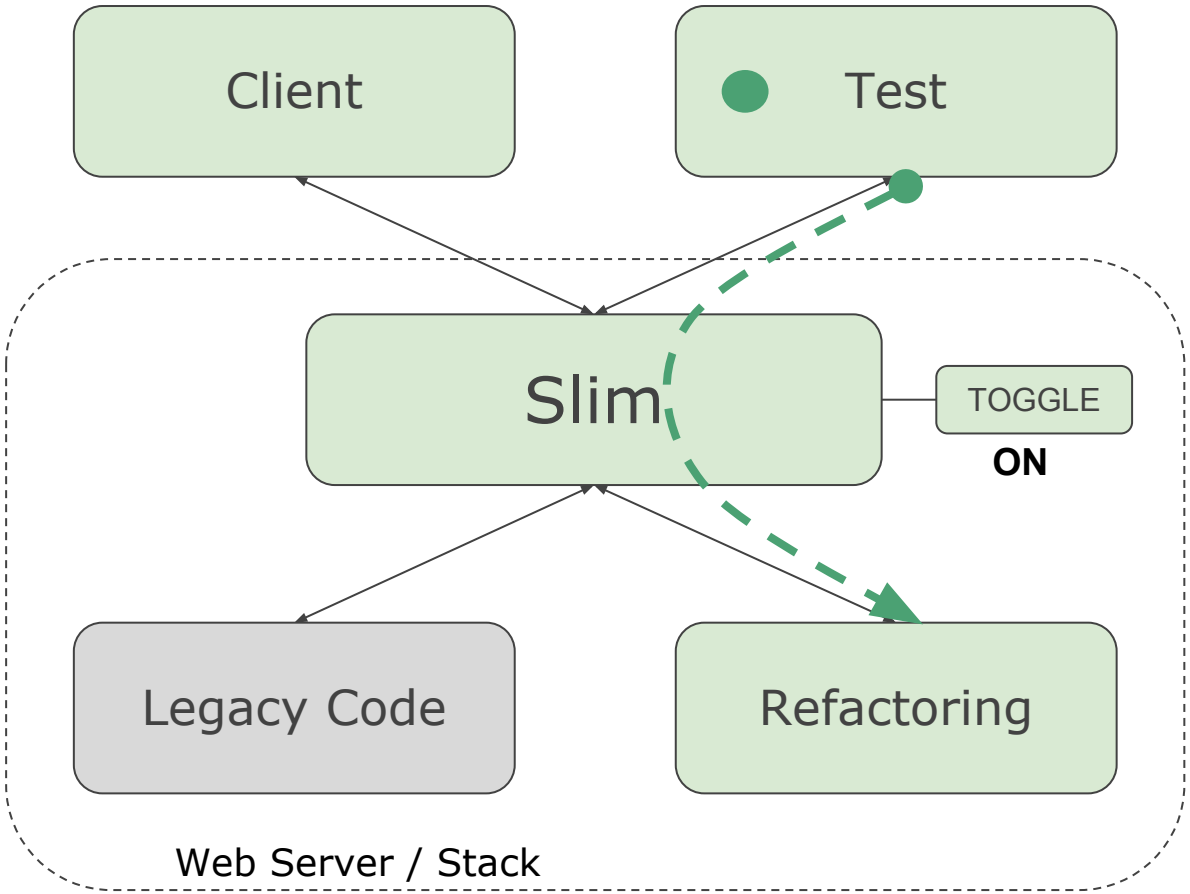
GitLabCI

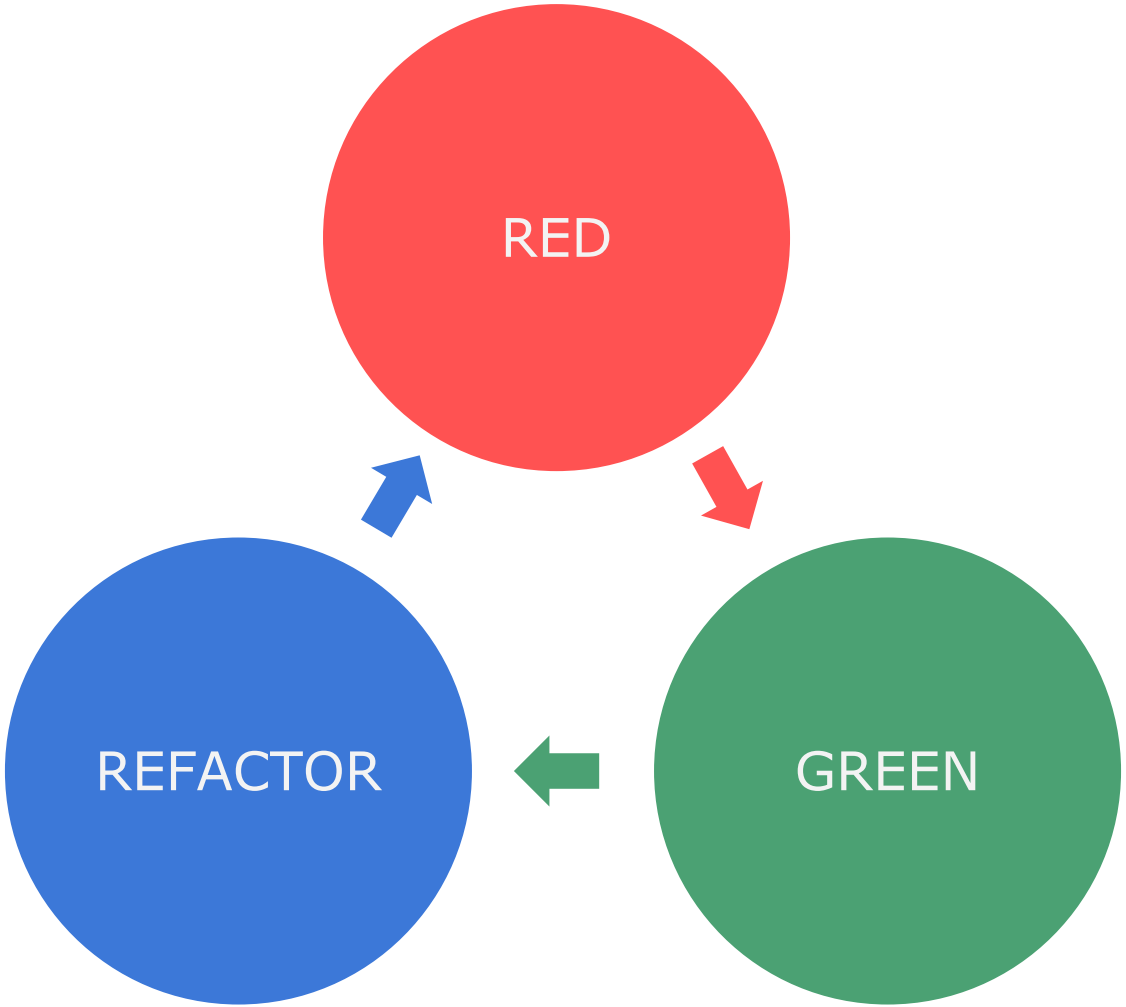
CircleCI



TravisCI

#4 Refactoring


Client
Test
Slim
TOGGLE
ON
Legacy Code
Refactoring
Web Server / Stack


RED
GREEN
REFACTOR

```php
// File: application/modules/api/controllers/V1Controller.php
public function indexAction() {
    $data = [ /* data */ ];
    $this->getResponse()->setBody(json_encode($data));
}
```

```php
// File: application/modules/api/controllers/V1Controller.php
public function indexAction() {
    $data = [ /* data */ ];
    $this->getResponse()->setBody(json_encode($data));
}

// ------------------------- 傳說中的分隔線 -------------------------
// File: index.php
$app->get('/api/v1', function (Request $request, Response $response) {
   $data = [ /* data */ ];

   $body = $response->getBody();
   $body->write(json_encode($data));
   return $response->withHeader('Content-type', 'application/json');
});
```

Router Rule

Slim

```
// All route pattern
$app->any('/{all:.*}', function (Request $request, Response $response) {
    // Legacy code
});
```

Slim

```php
$app->get('/api/v1', function (Request $request, Response $response) {
    // Refactoring code
});

// ------------------------- 我隔故我在 -------------------------

// All route pattern
$app->any('/{all:.*}', function (Request $request, Response $response) {
    // Legacy code
});
```

Feature Toggle

```php
$app->get('/api/v1', function (Request $request, Response $response) {
    // ...
});
```

```
~~$app->get('/api/v1', function (Request $request, Response $response) {~~
~~    // ...~~
~~});~~

// Development is true
// Production is false
$toggle = false;
define('ENABLE_REFACTORING', $toggle);

ENABLE_REFACTORING AND $app->get('/api/v1', function (Request $request, Response
$response) {
    // ...
});
```

```
$app->get('/api/v1', function (Request $request, Response $response) {
    // ...
});

// Development is true
// Production is false
$toggle = false;
define('ENABLE_REFACTORING', $toggle);

ENABLE_REFACTORING AND $app->get('/api/v1', function (Request $request, Response
$response) {
    // ...
});
```

```
// print "True Line"
true AND print('True Line');

// Nothing happened
false AND print('False Line');
```

```
$app->get('/api/v1', function (Request $request, Response $response) {
    // ...
});
```

```
// Development is true
// Production is false
$toggle = false;
define('ENABLE_REFACTORING', $toggle);
```

```
ENABLE_REFACTORING AND $app->get('/api/v1', function (Request $request, Response
$response) {
    // ...
});
```

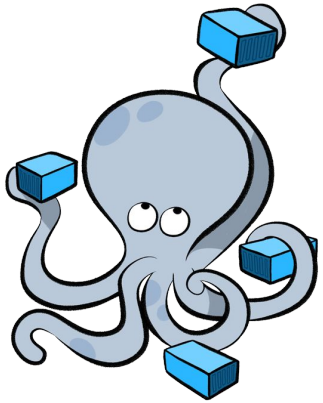
Environment Variable

```
$app->get('/api/v1', function (Request $request, Response $response) {
    // ...
});
```

```
// Production is 0
// Development is 1
$toggle = (boolean) getenv('ENABLE_REFACTORING');
define('ENABLE_REFACTORING', $toggle);
```

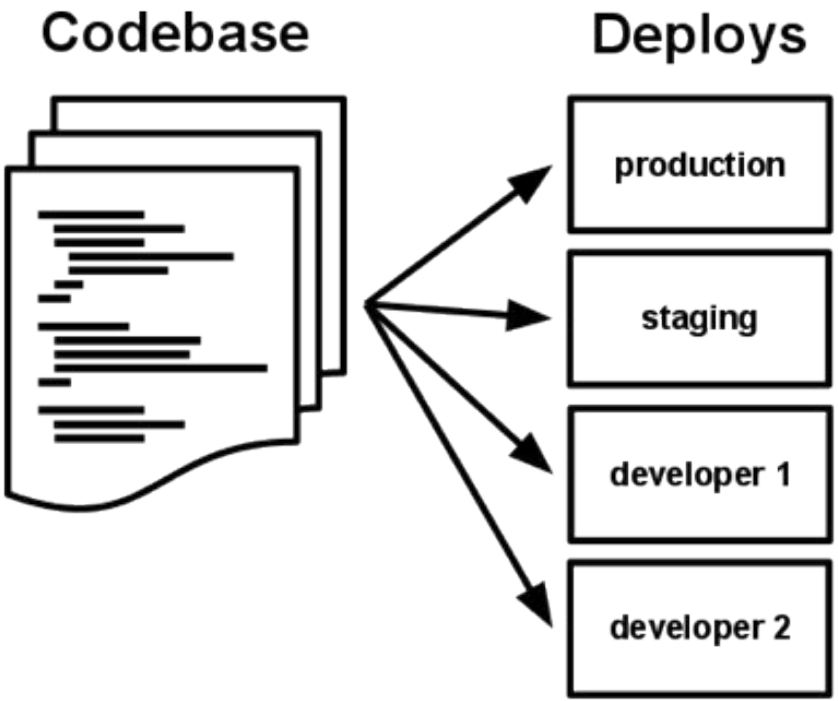
```
ENABLE_REFACTORING AND $app->get('/api/v1', function (Request $request, Response
$response) {
    // ...
});
```

Using Env. Variable



```yaml
// docker-compose.yml
web:
 image: php:7.0-apache
 ports:
   - 8080:80
 volumes:
   - .:/var/www/html
 environment:
   ENABLE_REFACTORING: 0
```

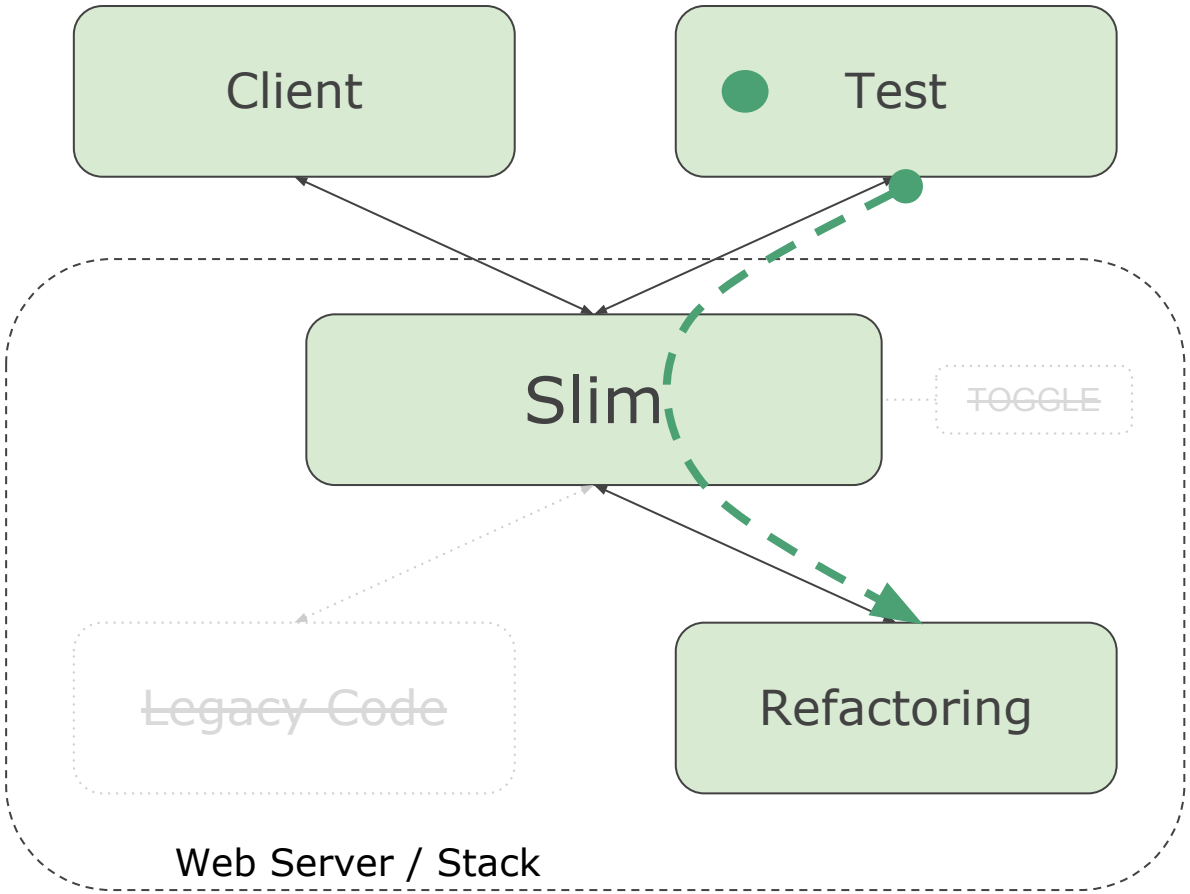
12 Factors

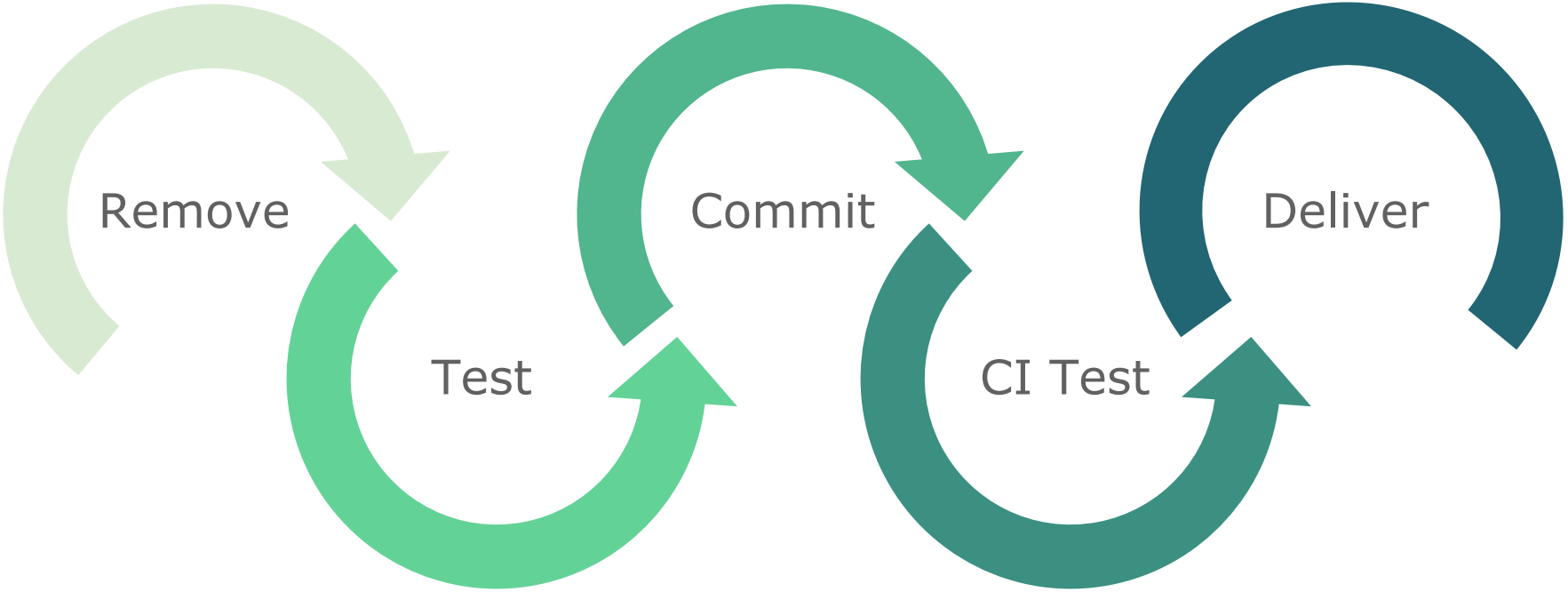

Codebase
Deploys
production
staging
developer 1
developer 2

Deploys Review

- Slim + Legacy Code 執行環境 (production)
- Test + Slim + Legacy Code 執行環境 (development)
- CI server 執行環境 (testing)

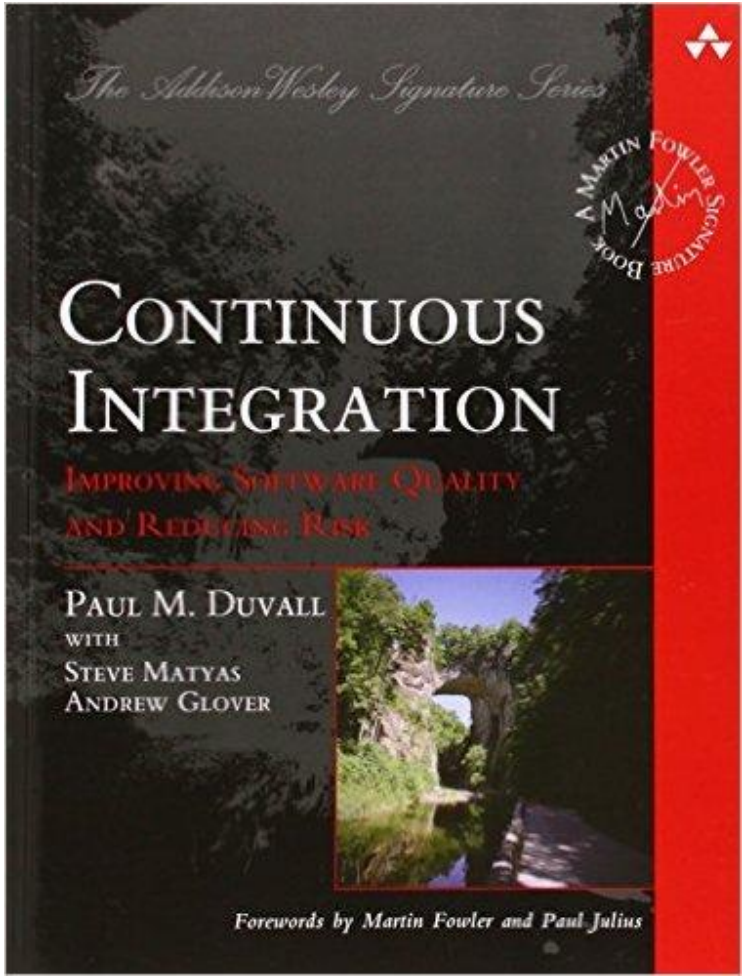
#5 Removing


Client
Test
Slim
~~TOGGLE~~
~~Legacy Code~~
Refactoring
Web Server / Stack


Remove
Test
Commit
CI Test
Deliver

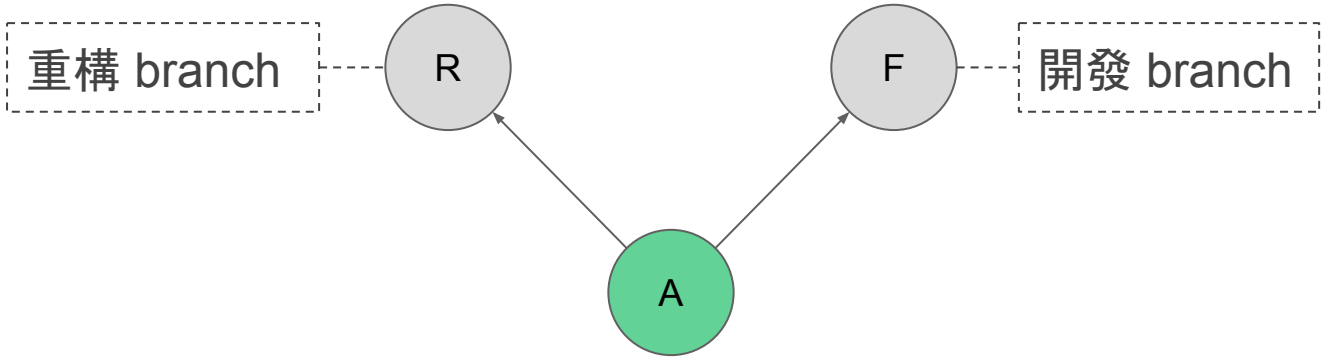
等等等等

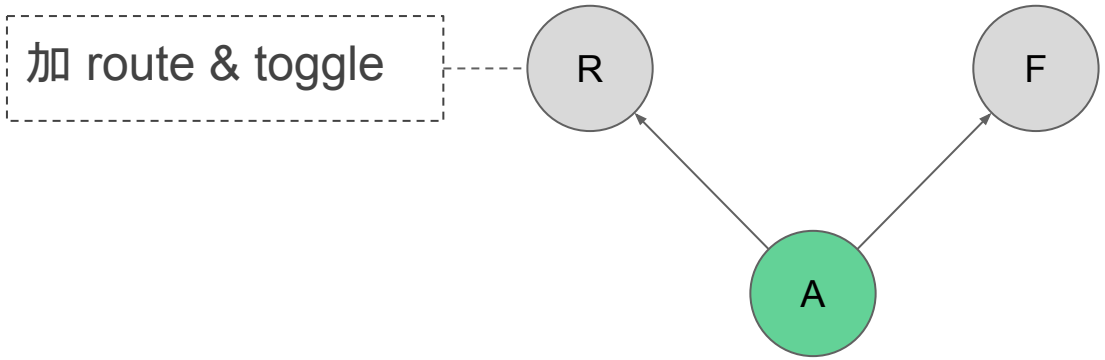
講那麼多，啊是要怎麼同時開發與重構

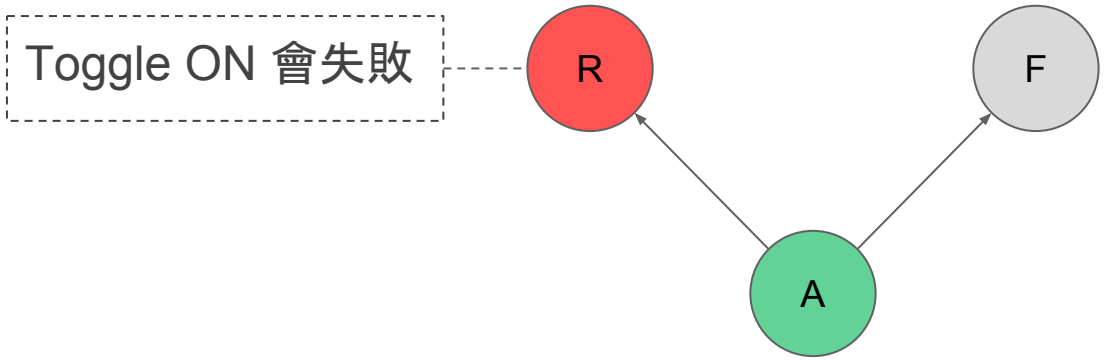

The Addison-Wesley Signature Series
A Martin Fowler Signature Book
Continuous Integration
Improving Software Quality and Reducing Risk
Paul M. Duvall
with
Steve Matyas
Andrew Glover
Forewords by Martin Fowler and Paul Julius

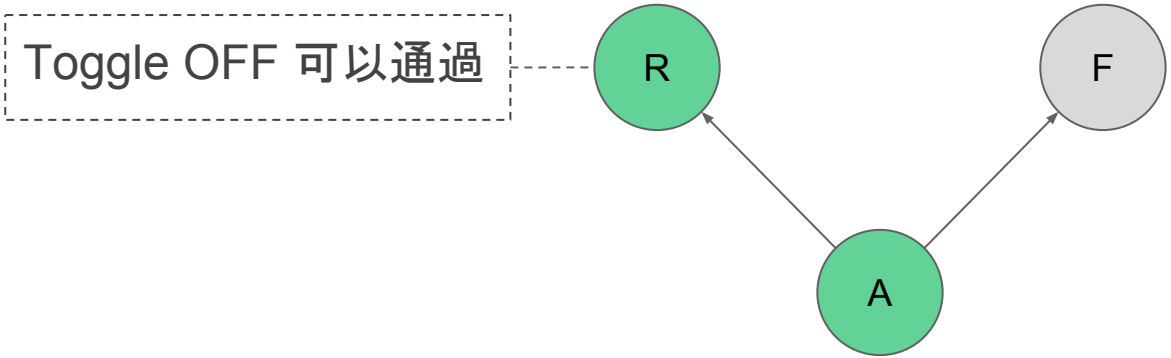
完成 #2 整合 Slim + #3 測試後

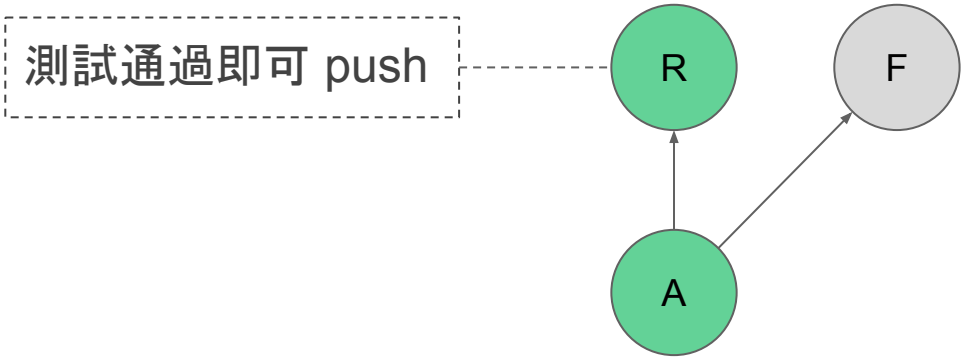
A

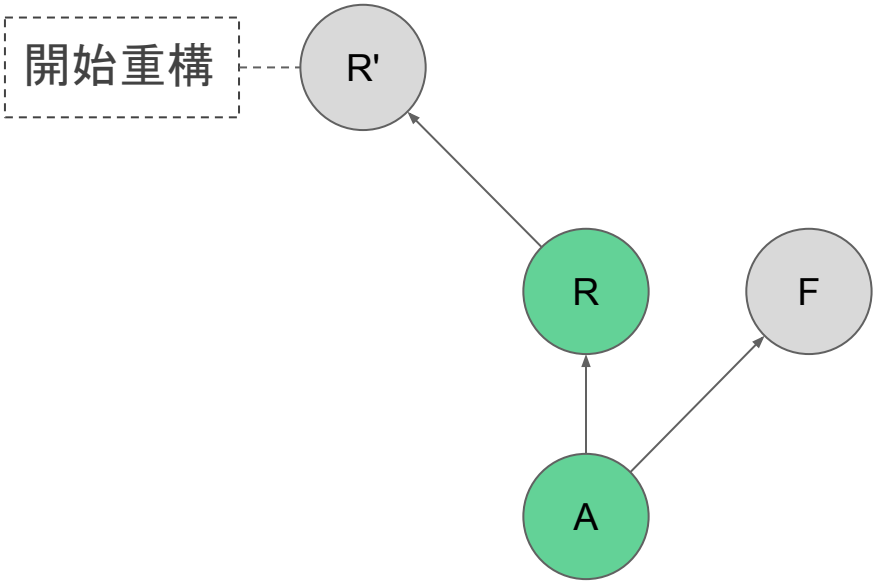

重構 branch
R
F
開發 branch
A

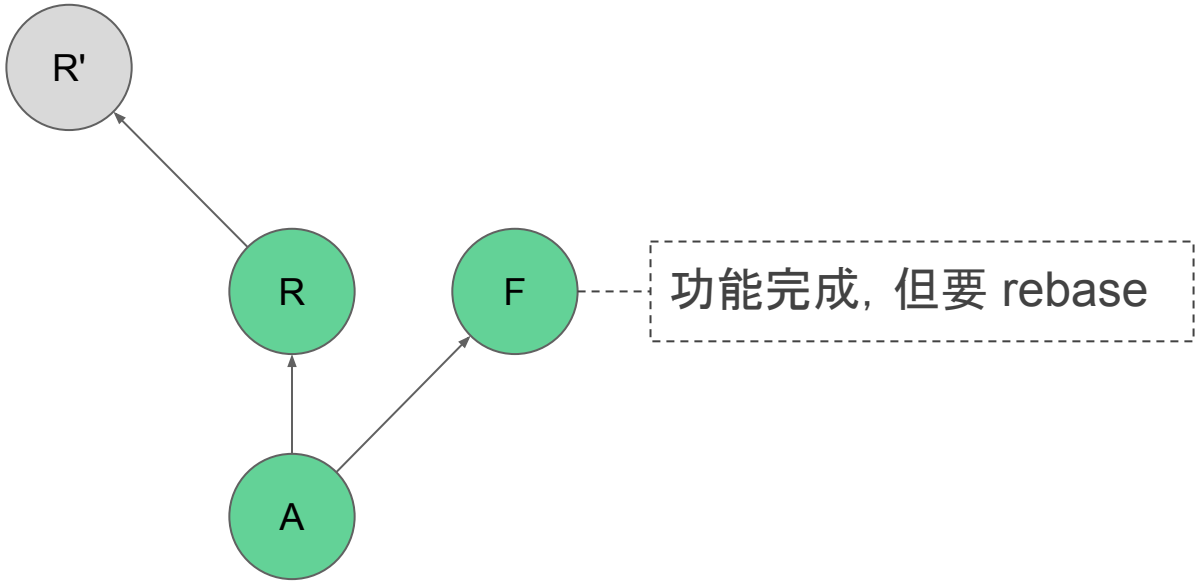

加 route & toggle
R
F
A

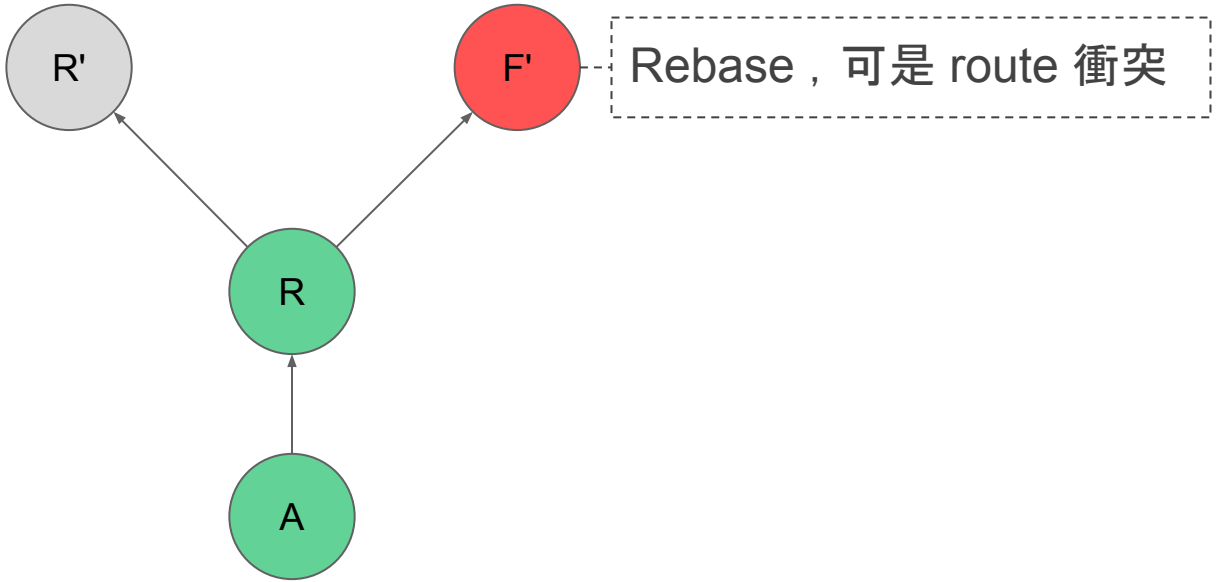

Toggle ON 會失敗
R
F
A

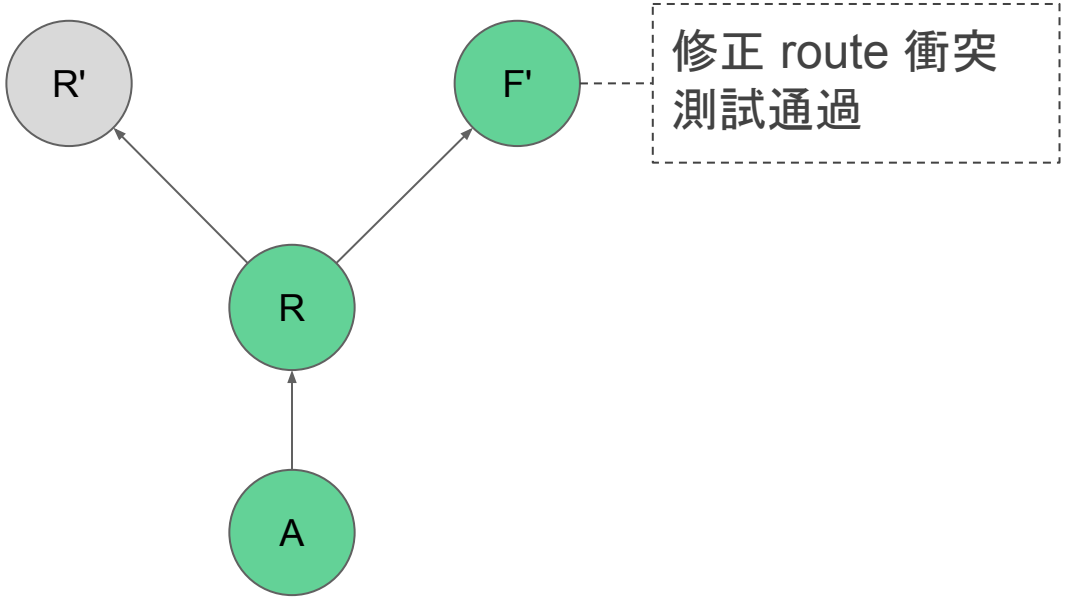

Toggle OFF 可以通過
R
F
A

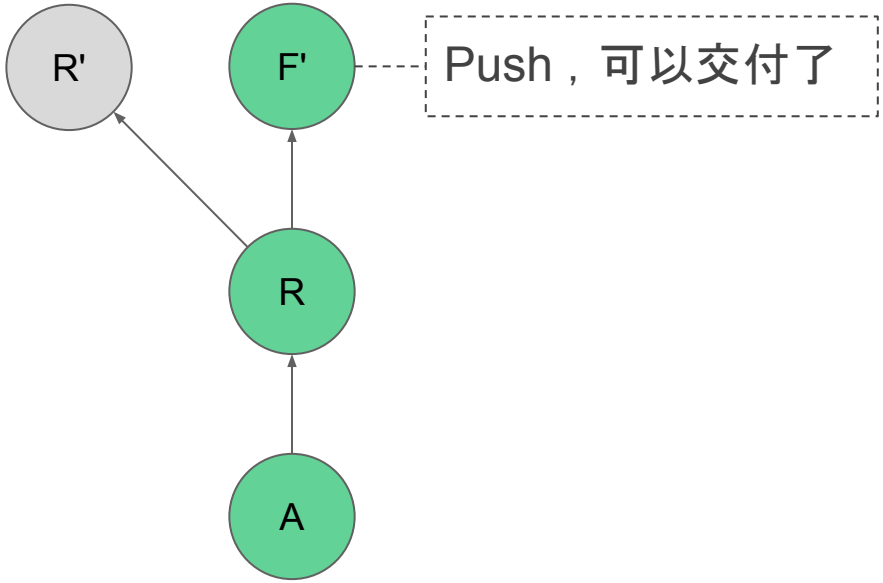

測試通過即可 push
R
F
A


開始重構
R'
R
F
A

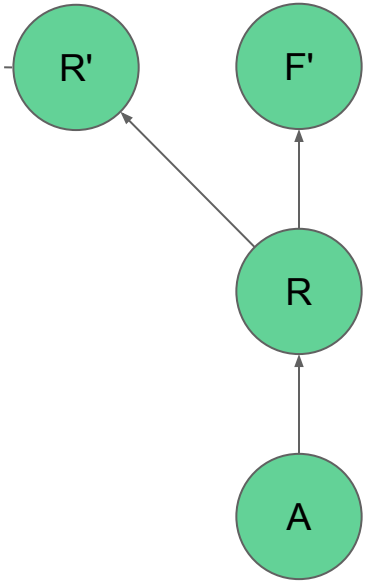

R'
R
F
A
功能完成，但要 rebase


R'
F'
Rebase，可是 route 衝突
R
A

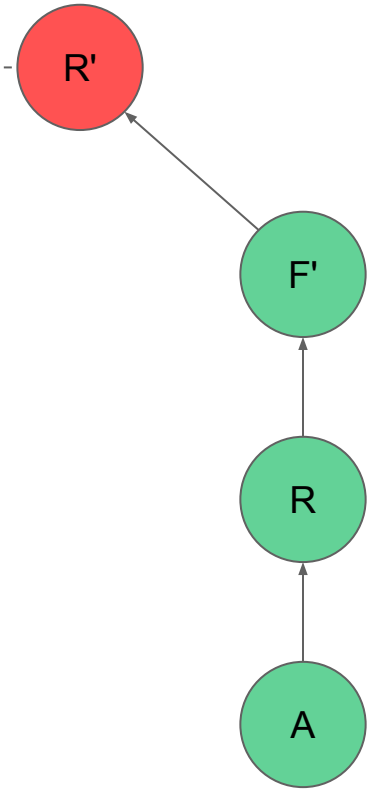

R'
F'
修正 route 衝突
測試通過
R
A


R'
F'
Push，可以交付了
R
A

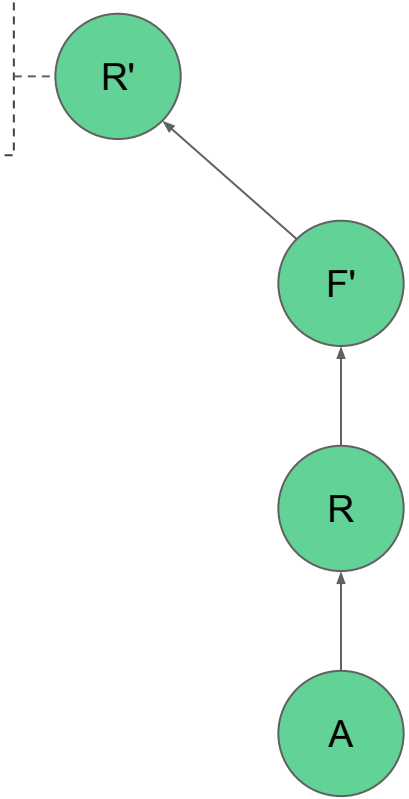
重構完成了
Toggle ON 通過

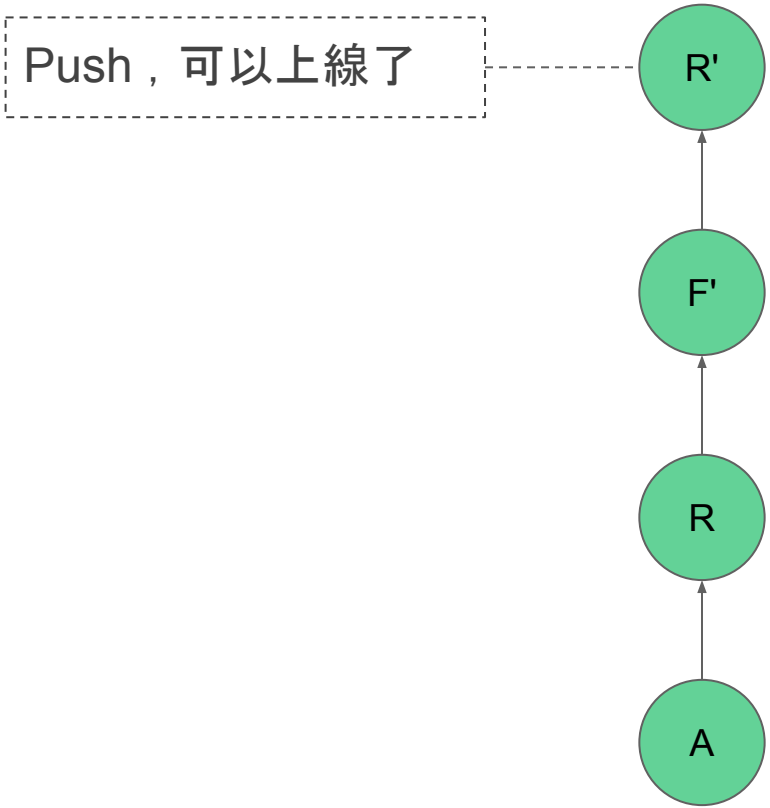


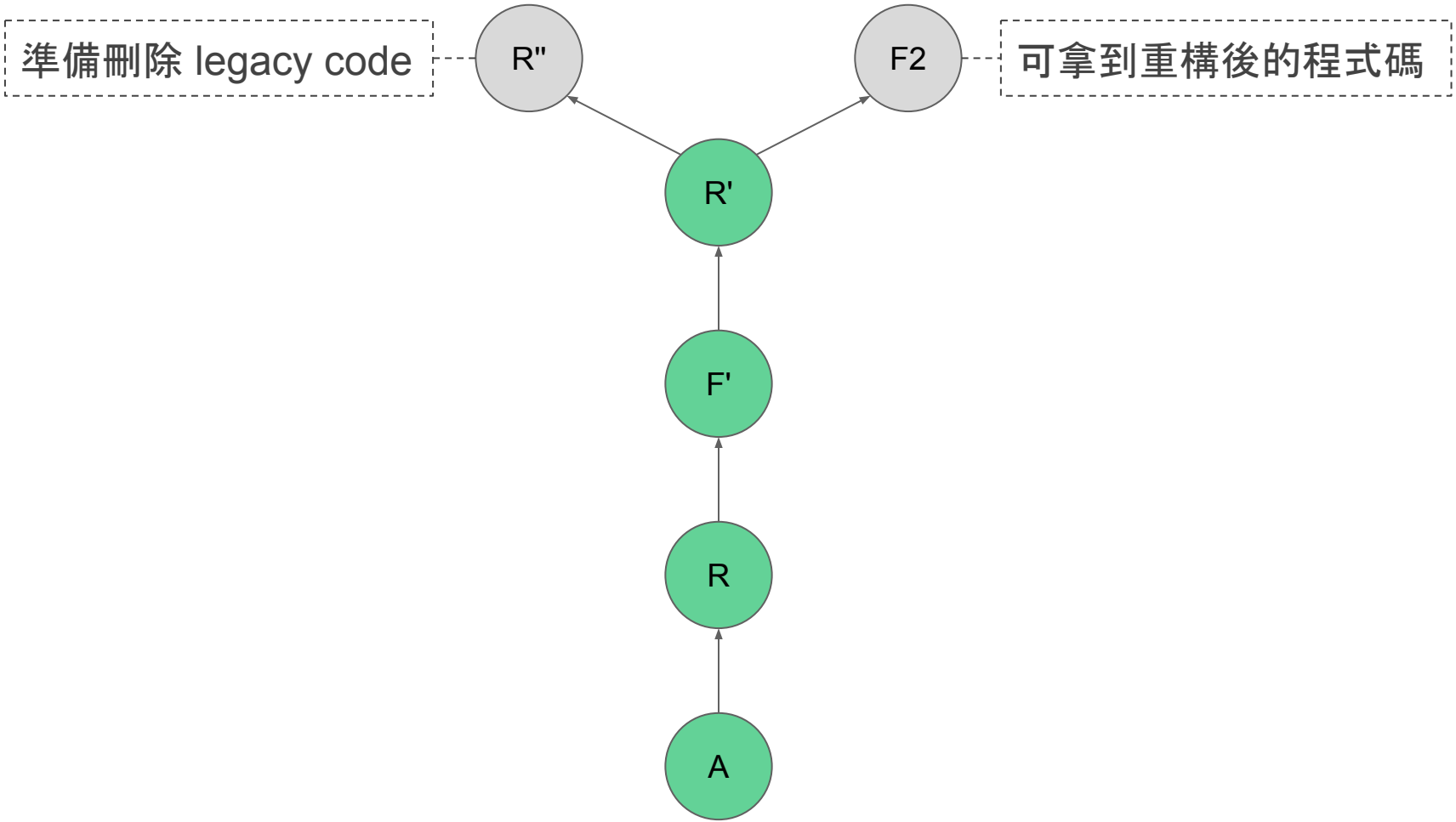
Rebase 沒衝突，可是測試不通過，開發功能時，剛好修改了 API



修改重構程式呼叫 API
的方法，測試通過




Push，可以上線了
R'
F'
R
A


準備刪除 legacy code
R''
F2
可拿到重構後的程式碼
R'
F'
R
A

CI 要領

預防勝於治療

及早發現，及早治療

Conclusion

Advantages

- 導入後，可以開始使用 Composer
- 導入後，可以開始寫自動化測試，與實踐 CI
- 導入後，問題會越來越少，風險也會越來越好控管

Disadvantages

Disadvantages

- 環境建置很困難

Disadvantages

- 環境建置很困難
  - 這是應該要做的任務

Disadvantages

- 環境建置很困難
  - 這是應該要做的任務
- 開發、維運與業務團隊配合困難

Disadvantages

- 環境建置很困難
  - 這是應該要做的任務
- 開發、維運與業務團隊配合困難
  - 公司團隊就是要一起合作，不然要幹嘛

Disadvantages

- 環境建置很困難
  - 這是應該要做的任務
- 開發、維運與業務團隊配合困難
  - 公司團隊就是要一起合作，不然要幹嘛
- 效能會因為多一層 proxy 而受到影響

Disadvantages

- 環境建置很困難
  - 這是應該要做的任務
- 開發、維運與業務團隊配合困難
  - 公司團隊就是要一起合作，不然要幹嘛
- 效能會因為多一層 proxy 而受到影響
  - 可考慮採用 Phalcon 或其他適合的解決方案

Q & A

快快樂樂 重構

平平安安 上線

Thanks!