GDG DevFest
TAIPEI 2017

學習如何學習 Gradle

qrtt1

我是誰 .java

EXTREMX

技能:

It's qrtt1

文組的工程師

臺南師範學院

喜歡教育心理學

GDG DevFest TAIPEI 2017

摸著良心問問

這是第幾次發願學好 Gradle

野外求生訓練，北方在哪兒？

總是在 build script 裡迷路嗎？

接下來的時間，我們將將補補概念知識

GDG DevFest TAIPEI 2017

我看了官網 User Guide

Gradle Build Tool 4.3.1

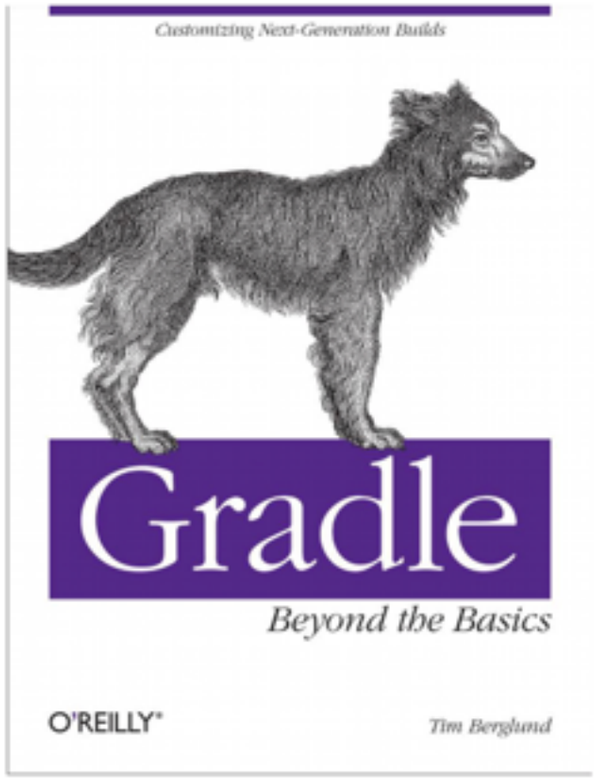
I. About Gradle

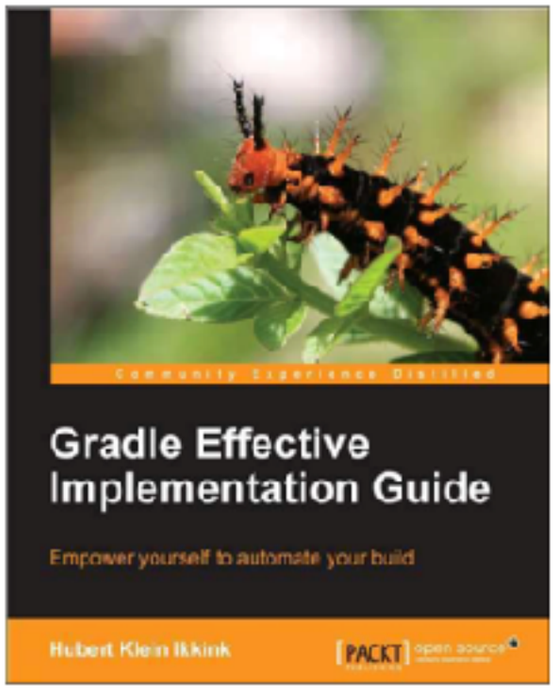
II. Working with existing builds

III. Writing Gradle build scripts

GDG DevFest TAIPEI 2017

我看了些書..., But





http://bit.ly/2mZqyGe

http://amzn.to/2AXaM0Z

GDG DevFest TAIPEI 2017

真的上戰場時

- 如何指定 java 相容版本
- 怎麼新增 1 個 task 在某個 task 後
- 如何在 compile 後把檔案複製到 ...
- 如何替 apk 的檔名加上 revision

最後我只能 ...

- 上 stack overflow 找個相似的片段
- 貼回 build.gradle 看看合不合用
- 修修改改，看起來符和需求就好

對比一下 Java 學習

當我還是個 Java 新手時，我在 google 上找到可以用的片段。
貼回 IDE 上，修修改改，覺得能動了！就立馬 commit 交差。

對比一下 Gradle 學習

當我還是個 Gradle 新手時，我在 google 上找到可以用的片段。
貼回 build script上，修修改改，覺得能動了！就立馬 commit 交差。

你根本不懂自己在寫什麼!!!

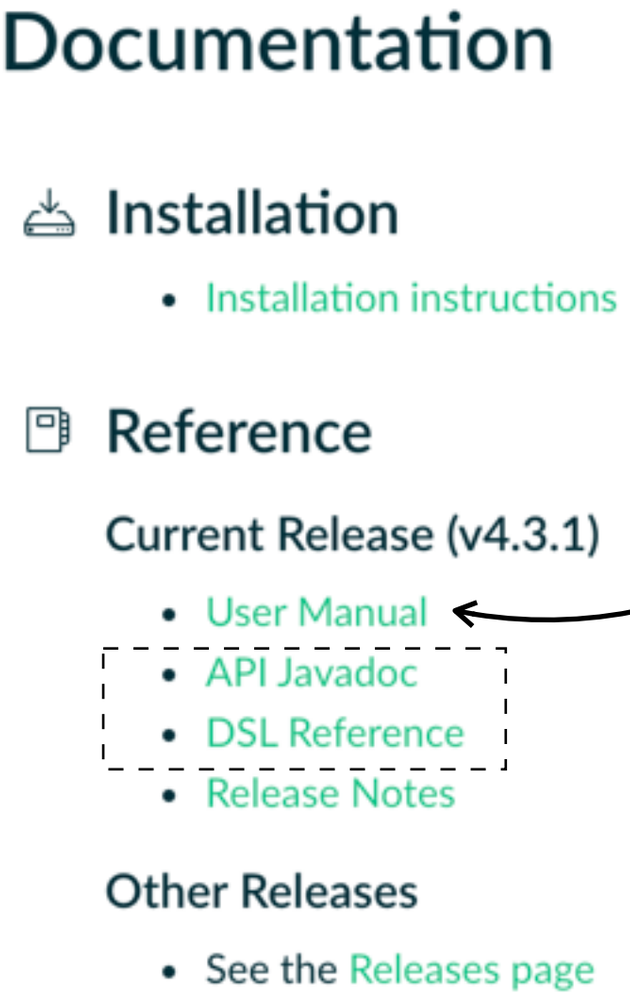
讓它去吧

TIPS: following steps

現在，一起學看看...

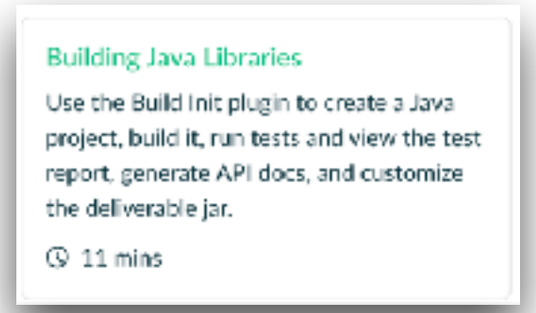
使用官方文件

https://gradle.org/docs/



你也是一開始陣亡在這嗎？

現在有簡易版的呦!



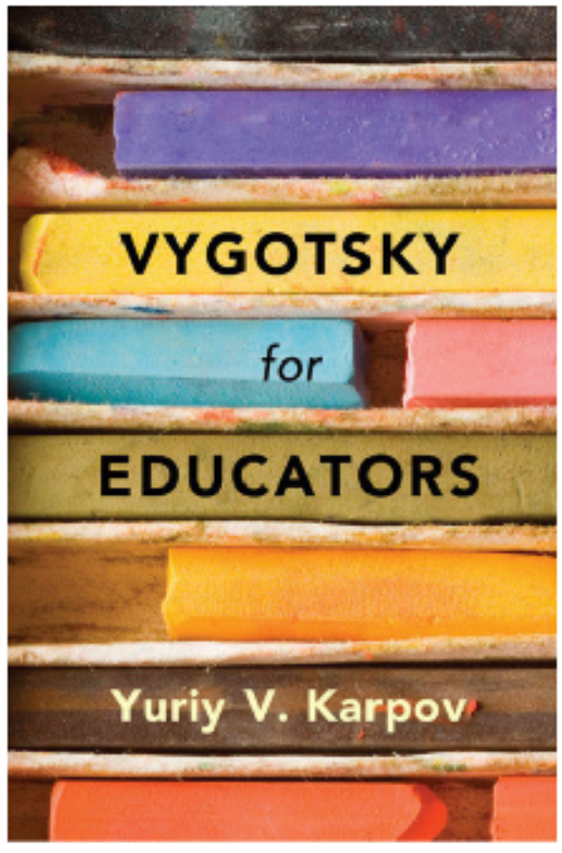
先冷靜一下

這一定有什麼問題，讓我一直學不會

知識的類型

教育心理學

課程設計的書



http://amzn.to/2izCr0W

事實 Factual

概念 Conceptual

程序 Procedural

後設認知 Metacognitive

知識的類型

事實 Factual 它很早就存在了，可能沒有人知道它為何存在。
它的存在不需要特別的理由，甚至沒有什麼原因可以解釋。你只能記住它。

- `apply plugin: 'java'` 引用 Java Plugin
- 開發 Java 專案需引用 Java Plugin
- 開發 Android 專案需引用 Android Plugin
- `task copyDocs(type: Copy)`

這麼寫建立得出 copy task

概念 Conceptual
程序 Procedural
後設認知 Metacognitive

知識的類型

概念 Conceptual 可以用來解釋、描述事實的知識。

from 與 into 是來至 closure delegate 給 Copy type

```
task copyDocs(type: Copy) {
    from 'src/main/doc'
    into 'build/target/doc'
}
```

事實 Factual

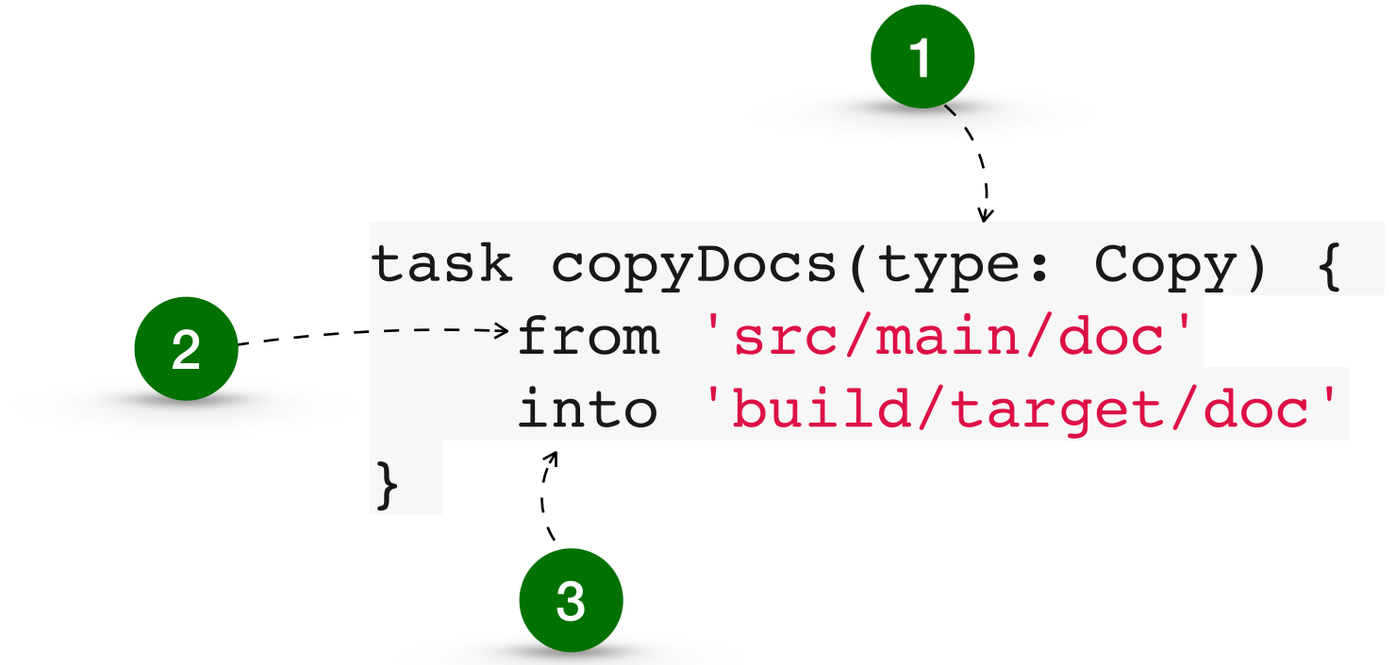
程序 Procedural

後設認知 Metacognitive

知識的類型

程序 Procedural

使用知識的方法，結合現實的情境練習，用它來面對未遭遇過的問題。



事實 Factual

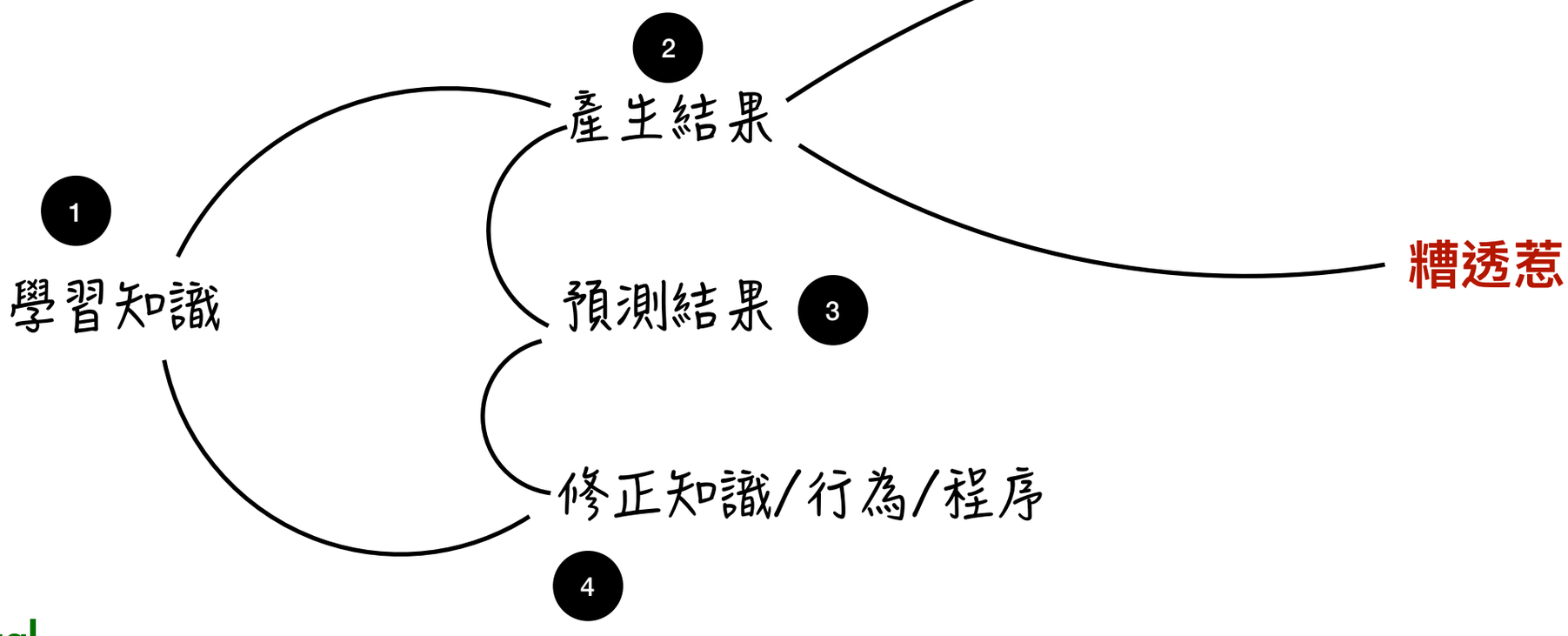
概念 Conceptual

後設認知 Metacognitive

知識的類型

後設認知 Metacognitive

自我調整 Self-regulation



事實 Factual

概念 Conceptual

程序 Procedural

GDG DevFest TAIPEI 2017

困境

Gradle Build Tool 4.3.1

I. About Gradle

1. Introduction
2. Overview

II. Working with

3. Installing Gradle
4. Using the Gradle C
5. The Gradle Console
6. The Gradle Wrapper
7. The Gradle Daemon
8. Dependency Management Basics
9. Introduction to multi-project builds
10. Continuous build
11. Composite builds
12. The Build Environment
13. Troubleshooting
14. Embedding Gradle using the Tooling API
15. Build Cache

III. Writing Gradle build scripts

16. Build Script Basics
17. Build Init Plugin

23. Wrapper Plugin

31. Comparing Builds
32. Publishing artifacts
33. The Maven Plugin

若這些作為『事實、程序』知識

得花多久時間記憶它!?

事實 Factual

概念 Conceptual

程序 Procedural

後設認知 Metacognitive

如何學得快?

- 正確地分類知識，需要大量『記憶』的部分用小抄取代

  其實抄久了就記住了，記不住就不常用，繼續留在小抄內吧。

- 連結過去相似的經驗，產生學習遷移

  話目就是舉一反三

- 連結過去的背景知識，抵消大量新『事實』知識的數量。

  知識的成長速度沒有想像中的快，大多只是變形與觀念的轉變

引用新概念消減事實過載

用我們比較熟悉的 .java 來看 Gradle

- Gradle 是 Groovy DSL
- Groovy 大致與 Java 一樣（除了部分的 syntax sugar）
- Groovy DSL 都有一個負責喬事情的物件

BasicScript

API Javadoc 與 DSL Reference

安全 | https://docs.gradle.org/4.3.1/dsl/

Build script blocks

`allprojects { }`
`artifacts { }`
`buildscript { }`
`configurations { }`
`dependencies { }`
`repositories { }`
`sourceSets { }`
`subprojects { }`
`publishing { }`

Core types

`Project`
`Task`
`Gradle`
`Settings`
`IncludedBuild`
`Script`
`JavaToolChain`
`SourceSet`
`SourceSetOutput`
`SourceDirectorySet`
`IncrementalTaskInputs`

Reference

Current Release (v4.3.1)

- User Manual
- API Javadoc
- DSL Reference
- Release Notes

Other Releases

- See the Releases page

有試著用過它嗎？

補充資料：【認識 Gradle】（4）看懂 Gradle Script
http://www.codedata.com.tw/java/understanding-gradle-4-gradle-script/

TIPS: println scriptTarget

喬事人 BasicScript

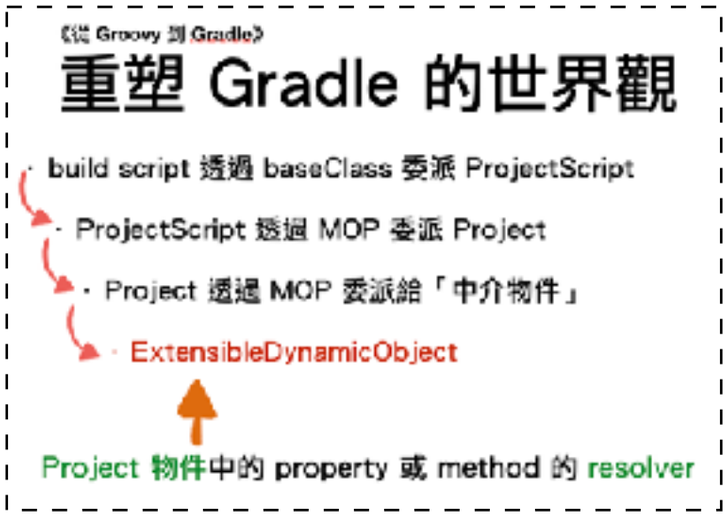
https://github.com/gradle/gradle/blob/v4.1.0-milestone-1/subprojects/core/src/main/java/org/gradle/groovy/scripts/BasicScript.java

```
public abstract class BasicScript extends org.gradle.groovy.scripts.Script implements
    private StandardOutputCapture standardOutputCapture;
    private Object target;
    private ScriptDynamicObject dynamicObject = new ScriptDynamicObject(this);

    public void init(Object target, ServiceRegistry services) {
        standardOutputCapture = services.get(StandardOutputCapture.class);
        setScriptTarget(target);
    }

    public Object getScriptTarget() {
        return target;
    }

    private void setScr
        this.target = t
        this.dynamicObj
    }

    public StandardOutputCapture getStandardOutputCapture() {
        return standardOutputCapture;
    }
```

所有的動作都透過 **MetaObjectProtocol** 呼叫 **target**

當你對一個物件呼叫 method 或 access field 時，
當被使用對象不存在的替代執行路徑

今天只要學會查 Javadoc 就成功一半了！



說人話！!!

TIPS: find Project core-type in the document

請把 BuildScript 當作 Project 物件

所有的 method call / property access 都先當是對 Project 物件來做事。

https://docs.gradle.org/4.3.1/dsl/org.gradle.api.Project.html

TIPS: println delegate

如果操作的對象不是 Project 該怎麼辦？

先認識一下必要的 Groovy 語法

In Gradle 新
Groovy 常用語法

optional typing 懶得寫 type
就用 def (其實就是 Object)

new HashMap()

```
def map = [:]
```

看到 [] 就是容器，
看到 : 就是 Map

```
def colors = [
    red: '#FF0000',
    green: '#00FF00',
    blue: '#0000FF']
```

http://www.groovy-lang.org/syntax.html

問它一下 delegate 是誰

Groovy DSL Features

- Closure 支援 delegate 機制

{} closure 將實作 delegate 給 Copy

```
task copyDocs(type: Copy) {
    from 'src/main/doc'
    into 'build/target/doc'
}
```

https://docs.gradle.org/current/javadoc/org/gradle/api/tasks/Copy.html

https://docs.gradle.org/4.3.1/dsl/org.gradle.api.tasks.Copy.html

野外求生技能摘要

- 認識 MetaObjectProtocol 後，知道 Project 通常是直接作用對象

概念　　事實

- 認識 Closure 特性後，練習透過 delegate 找出負責的物件，就查到教學文件沒跟你介紹過的 method

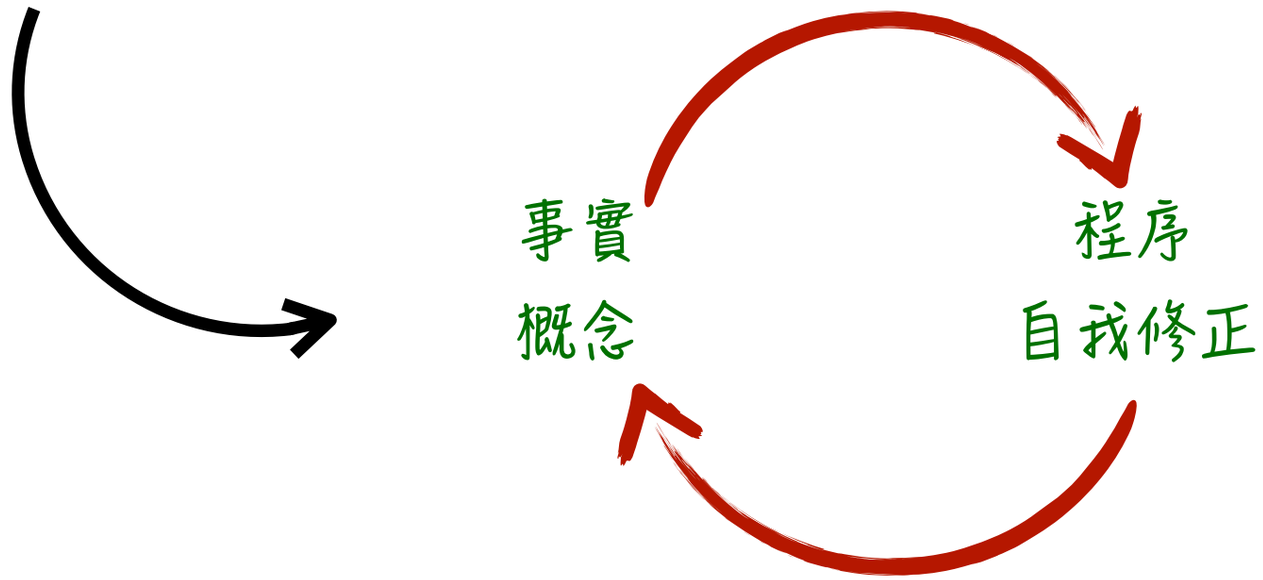
概念

事實

對比一下 Gradle 學習

當我還是個 Gradle 新手時，我在 google 上找到可以用的片段。
貼回 build script上，修修改改，覺得能動了！就立馬 commit 交差。

你根本不懂自己在寫什麼!!!



後續學習建議

- 《由 Groovy 到 Gradle》 概念

  https://github.com/qrtt1/JCConf2015_from_groovy_to_gradle

- Gradle RuleSource Plugin 筆記 概念 事實

  https://github.com/qrtt1/Notes/blob/master/20170312_gradle.rule.source.plugin.md

- MultipleFlavor 實作思路 概念 程序

  https://github.com/qrtt1/Notes/blob/master/20170312_android.multi.flavors.md

你沒有學會 Gradle，但學會了如何學習 Gradle

Q & A

不可以問太具體的，會不會做 OOO 的功能。因為我也不知道呦！