|  |  |  |  |
| --- | --- | --- | --- |
| **Subject :** | QPlay Backend Platform | **Latest Ver :** | 1.4.4 |
| **Latest Release Date :** | 2018-08-28 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Change History** | | | | |
| Date | Author | Version | Type | Comment |
| 2018-05-23 | Darren.K.Ti | 1.0 | Function  DB | 功能修改記錄 : 4.1  1. QPlay 後台新增功能 [ 公司管理 ]  2. QPlay Login 流程修改  3. DB `qplay` 新增 Table  4. QPlay Backend Login - Remember Me |
| 2018-06-19 | Darren.K.Ti | 1.1 | Function  DB | 功能修改記錄 : 4.2  1. QPlay 後台新增功能 [ 消費券管理 ]  2. DB `qplay` 新增 QPay 相關 Table  3. QPlay API 新增   1. QPay 相關 API 2. 更新 QAccount Password   4. APP QPlay   1. 整合 QPay 2. 新增功能 [ 更新 QAccount Password ]   5. QPay Web 查詢頁面 |
| 2018-06-25 | Darren.K.Ti | 1.2 | Function DB  API | 功能修改記錄 : 4.3  1. QPlay 後台新增功能 [ Function Management ]  2. DB `qplay` 新增 Function Manage 相關 Table  3. QPlay API 新增 Function Manage 相關 API |
| 2018-07-06 | Darren.K.Ti | 1.3 | Function  API | 功能修改記錄 : 4.4  1. Plugin QLogin 頁面 支援 QAccount  2. QPlay API login 支援 QAccount  3. QPlay API register 支援 QAccount  3. APP QPlay 使用 Function Manage |
| 2018-07-16 | Darren.K.Ti | 1.3.1 | Function  DB  API | 功能修改記錄 : 4.3  1. Function Detail 的權限管理, 要處理 company\_label  2. 新增設定 QAccount 的權限等級 |
| 2018-07-20 | Darren.K.Ti | 1.4 | Function  DB  API | 功能修改記錄 : 4.2  1. 完成 2018-06-25 尚未完成的項目 |
| 2018-07-26 | Darren.K.Ti | 1.4.1 | Function  API | 功能修改記錄 : 4.2  1. QPay Web API : 修正執行流程  2. 依據功能，個別評估時程 |
| 2018-07-31 | Darren.K.Ti | 1.5 | Function  DB | 功能修改記錄 : 4.5  1. QPlay 後台新增功能 [預約推播]  2. DB `qplay` 新增 預約推播 相關 Table  3. QPlay API - getMessageList 要過濾預約推播 |
| 2018-08-02 | Darren.K.Ti | 1.5.1 | Function | 功能修改記錄 : 4.5  3. QPlay API - getMessageList 要過濾預約推播   * 修正處理邏輯 |
| 2018-08-07 | Darren.K.Ti | 1.5.2 |  | 功能修改記錄 : 4.5 - 修正文字錯誤  4.5.12 程式碼修改記錄  qplay/ app/ Http/ Controllers/ pushController.php   * pushBatchService () - return view “push/ **push\_batch\_service**” * pushBatchServiceData () - return view “push/ **push\_batch\_service\_data**” |
| 2018-08-16 | Darren.K.Ti | 1.4.2 | Function | 功能修改記錄 : 4.3  4.3.12 程式碼修改記錄 -  1. 3.d. 修改文字 'FUNCTION\_VARIABLE\_NAME'  2. 2.h.ii.4.  Function Name 修正為可以編輯修改  Variable Name 修正為不可編輯修改 |
| 2018-08-16 | Darren.K.Ti | 1.4.3 | API | 功能修改記錄 : 4.4  1. QPlay API login 支援 QAccount  2. QPlay API register 支援 QAccount  3. QPlay API logout 支援 QAccount  4.4.12 程式碼修改記錄 qplayApi/  g. app/ Repositories/ UserRepository.php -  funciton QAccountLogin ()  修改程式執行邏輯，QAccount 的  帳號需要比對 `qp\_user`.emp\_no，另外，  不需要驗證 `qp\_user`.ad\_flag = N  Verify.php / CommonUtil.php 新增 function，透過 emp\_no  取得 User 資料 |
| 2018-08-24 | Darren.K.Ti | 1.6 | API | 功能修改記錄 : 4.6  1. QPlay API register 紀錄 ad\_flag 在 qp\_session  2. QPlay API login 紀錄 ad\_flag 在 qp\_session  3. QPlay API sendPushToken 在 JPush  註冊 QAccount Tag |
| 2018-08-28 | Darren.K.Ti | 1.4.4 | Funciton API | 功能修改記錄 : 4.3  1. 4.3.3 對應文件   * API 文件更新為 Qplay\_API\_1.3.9\_20180828   2. 4.3.4 修改項目   * 檔案不加入修改   + qpalyApi/ qplayController.php * 新增檔案   + qplayApi/ AppRepository.php   + qplayApi/ appFunctionController.php * 檔案名稱異動   + qplayApi/ FunctionUserRepository.php => UserFunctionRepository.php   + qplayApi/ FunctionRoleRepository.php => RoleFunctionRepository.php   3. 4.3.12 程式碼修改記錄   * qplayApi/ CommonUtil.php   + 取消新增     - function getUserEnableInfoByUserID ()   + 加入新增     - function getUserRole ()     - function ADFlag () * qplayApi/ FunctionRepository.php   + function getFunctionByApp 改為 function getFunctionListByAppId * qplayApi/ UserFunctionRepository.php   + function getFunctionRoleData 改為 function getAlloeRole * qplayApi/ RoleFunctionRepository.php   + function getFunctionUserData 改為 function getAlloeUser * qplayApi/ FunctionService.php   + function getFunctionList 改為 function getFunctionListByAppId |

# 目錄

[**目錄**](#_fvukxvw609q9) **3**

[**1.0 系統設計 (System Design)**](#_17qb5z9rsywe) **4**

[**2.0 資料庫設計 (Database Design)**](#_svpws9pqxbdm) **5**

[**3.0 Implementation**](#_l3yah7s6jx9k) **5**

[**4.0 功能修改記錄**](#_y9sxdk33wuqt) **6**

[4.3 [ QPlay 後台新增功能 [ Function Management ]，DB `qplay` 新增 Function Manage 相關 Table，QPlay API 新增 Function Manage 相關 API ]](#_xfzx9vmshw0k) 6

# 

# 1.0 系統設計 (System Design)

**1.1 系統架構圖 (System Framework Diagram)**

* QPlay Backend Platform 整體的系統架構圖，待補上

**1.2 系統流程圖 (System Flow)**

* QChat API 基於 laravel 5.2 框架進行開發，並使用 Repository + Service 模式之程式架構。
* 使用 index.php 作為進入點，route 為路由分配，midware 作為中介層，進行 request 和 response 的驗證，以及寫入 API Log。



**1.3 類別圖 (Package / Class Diagram)**

* QPlay Backend Platform 整體的類別圖，待補上

# 2.0 資料庫設計 (Database Design)

請參考資料庫設計文件 : QPlay\_DB\_Schema.docx

2.1 資料庫物件清單 (Database Object List)

* 每個修改項目 4.X 各自描述

2.2 實體關聯圖 (Entity-Relation Chart)

* 每個修改項目 4.X 各自描述

2.3 資料表結構 (Table Layout)

* 每個修改項目 4.X 各自描述

# 3.0 Implementation

3.1 公用元件 (Public Component)

* 每個修改項目 4.X 各自描述

3.2 共用介面 (Public Interface)

* 每個修改項目 4.X 各自描述

# 4.0 功能修改記錄

## 4.3 [ QPlay 後台新增功能 [ Function Management ]，DB `qplay` 新增 Function Manage 相關 Table，QPlay API 新增 Function Manage 相關 API ]

4.3.1 需求說明

* **QPlay 後台新增功能 [ Function Management ]**，用來管理各個 APP 內部頁面功能的使用權限:
  + 產生以下操作功能
    - Function List
    - Function Add
      * 綁定 APP。
    - Function Edit
      * 設定啟用 / 停用 / 刪除，使用權限。
  + 每個 Function 綁定一個 APP，依據 [QAccount] 或 [企業角色] 或 [使用者] 來設定使用權限。
    - 權限的比重 :
      * QAccount 和 公司兩者之間的比重大小順序，依照不同情境設定。
        + [QAccount] < [企業角色] < [使用者]
        + [企業角色] < [QAccount] < [使用者]
        + [QAccount] < [公司]
        + [公司] < [QAccount]
    - ex:
      * 重設 QAccount 密碼的權限
        + [QAccount] < [公司]

QAccount : false

[公司 Shop] : true

* + - * + 消費券店家，屬於 QAccount 的成員，無法重設 QAccount 密碼，但是，同時屬於 [公司 Shop]，所以最後決定，可以重設 QAccount 密碼。
        + 沒有 AD 的員工，屬於 QAccount 的成員，最後決定，無法重設 QAccount 密碼。
      * News 的使用權限
        + [公司] < [QAccount]

[公司 Qisda] : true

QAccount : false

* + - * + Qisda DL 員工，屬於 QAccount 的成員，所以最後決定，

沒有 News 的使用權限。

* Qisda IDL 員工，不屬於 QAccount 的成員，所以最後決定，

可以使用 News。

* **DB `qplay` 新增 Function Manage 相關 Table**。
* **QPlay API 新增 Function Manage 相關 API** :
  + getFunctionList
    - 備註 :
      * 必須考慮到，之後會透過此 API 取得 APP List Data。
      * 以下項目，目前暫時不考慮 :
        + APP 分類。
        + APP 排序。
  + getFunctionDetail
    - 備註 :
      * 必須考慮到，之後會透過此 API 取得 APP Detail Data。

4.3.2 時程評估

* [ QPlay 後台新增功能 (Function Management) ]
  + 後台選單參數設定 / 多國語系設定
    - 預估時程 : ? ~ ? (4 hr)
    - 實際時程 :
  + DB Table 新增
    - 預估時程 : ? ~ ? (2 hr)
    - 實際時程 :
  + Function List
    - 預估時程 : ? ~ ? (6 hr)
    - 實際時程 :
  + Function Add
    - 預估時程 : ? ~ ? (6 hr)
    - 實際時程 :
  + Function Edit
    - 預估時程 : ? ~ ? (2 Day)
    - 實際時程 :
* [ QPlay API 新增對應的 API ]
  + API getFunctionList
    - 預估時程 : ? ~ ? (3 Day)
    - 實際時程 :
  + API getFunctionDetail
    - 預估時程 : ? ~ ? (2 Day)
    - 實際時程 :

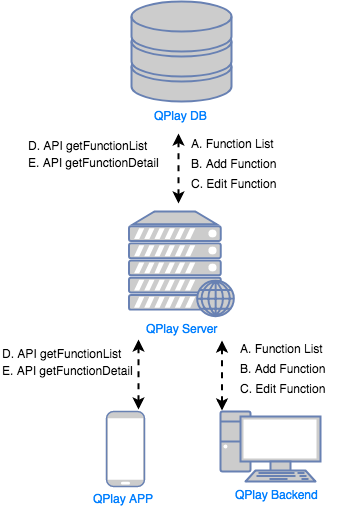
4.3.3 對應文件

* QPlay\_DB\_Schema\_3.10\_20180817.docx
* QPlay\_API\_1.3.9\_20180828.docx

4.3.4 修改項目

* DB `**qplay**`
  + 新增 Table
    - `**qp\_function**`
    - `**qp\_role\_function**`
    - `**qp\_user\_function**`
* qplayApi/ qplayApi/
  + 修改
    - app/ Http/ **routes.php**
    - ~~app/ Http/ Controllers/~~ **~~qplayController.php~~**
    - app/ lib/ **CommonUtil.php**
  + 新增
    - app/ Http/ Controller/ **appFunctionController.php**
    - app/ Model/
      * **QP\_Function.php**
      * **QP\_Role\_Function.php**
      * **QP\_User\_Function.php**
    - app/ Repositoreies/
      * **FunctionRepository.php**
      * **~~FunctionRoleRepository.php~~ RoleFunctionRepository.php**
      * **~~FunctionUserRepository.php~~ UserFunctionRepository.php**
    - app/ Services/ **FunctionService.php**
* qplay/
  + 修改
    - app/ Http/ **routes.php**
    - app/ lib/ **ResultCode.php**
    - resources/ lang
      * en/ **messages.php**
      * zh-cn/ **messages.php**
      * zh-tw/ **messages.php**
  + 新增
    - app/ Model/
      * **QP\_Function.php**
      * **QP\_Role\_Function.php**
      * **QP\_User\_Function.php**
    - app/ Repositories/
      * **FunctionRepository.php**
      * **FunctionRoleRepository.php**
      * **FunctionUserRepository.php**
    - app/ Http/ Controllers/ **FunctionController.php**
    - app/ Services/ **FunctionService.php**
    - resources/ views/ **function\_maintain**/
      * **function\_list.blade.php**
      * **function\_detail.blade.php**

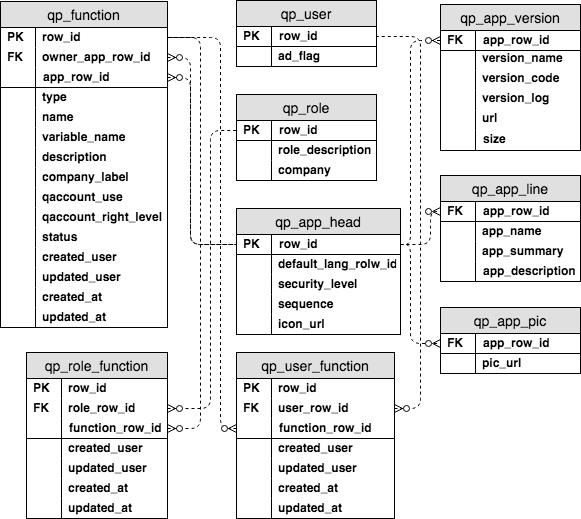
4.3.5 功能架構圖



4.3.6 資料庫物件清單

|  |  |  |  |
| --- | --- | --- | --- |
| **DB Type** | **Type** | **Object Name** | **Description** |
| MySQL | Table | qp\_function | 各個 APP 的頁面功能，可以依據 [企業角色] 或是 [使用者]，限制使用的權限。 |
| MySQL | Table | qp\_role\_function | 依據 [企業角色] 設定可以使用哪些 Function。 |
| MySQL | Table | qp\_user\_function | 依據 [使用者] 設定可以使用哪些 Function。 |
| MySQL | Table | qp\_user | 取得 User 的 row\_id 和 ad\_flag (判定是否為  QAccount)。 |
| MySQL | Table | qp\_app\_head | 取得 APP 的 row\_id。 |
| MySQL | Table | qp\_role | 取得 Role 的 row\_id。 |
| MySQL | Table | qp\_app\_line | 取得 APP 的語系資料。 |
| MySQL | Table | qp\_app\_version | 取得 APP 的版本資料。 |
| MySQL | Table | qp\_app\_pic | 取得 APP 的截圖資料。 |
| MySQL | Table | qp\_language | 取得 QPlay 的語系。 |

4.3.7 實體關聯圖



4.3.8 資料表結構

* 參考文件 : QPlay\_DB\_Schema\_3.10\_20180817.docx

4.3.9 公用元件

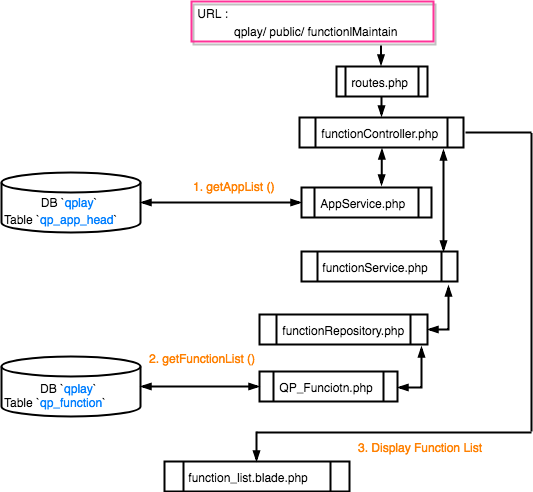
* 內部模組

4.3.10 共用介面

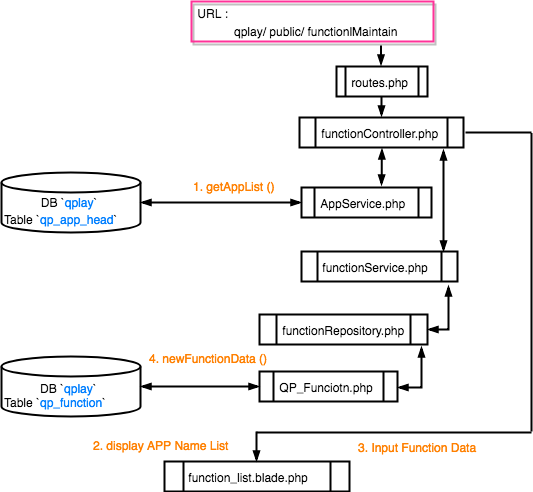
* 請參考 API 文件 : QPlay\_API\_1.3.9\_20180828.docx

4.3.11 資料流程圖 Data Flow Diagram

* QPlay 後台新增功能 [Function Manage] - Function List



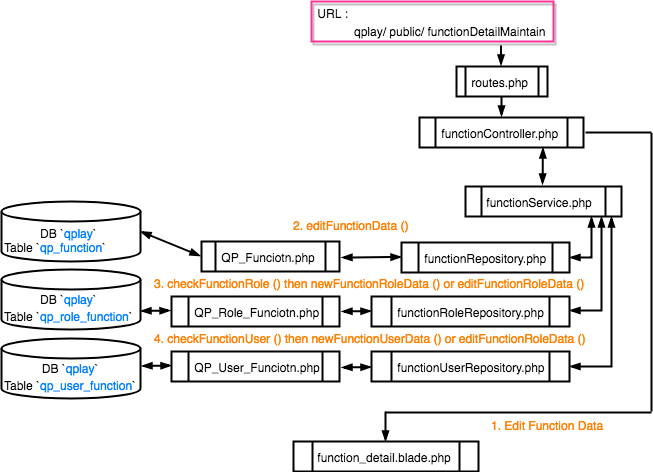
* QPlay 後台新增功能 [Function Manage] - Function Add



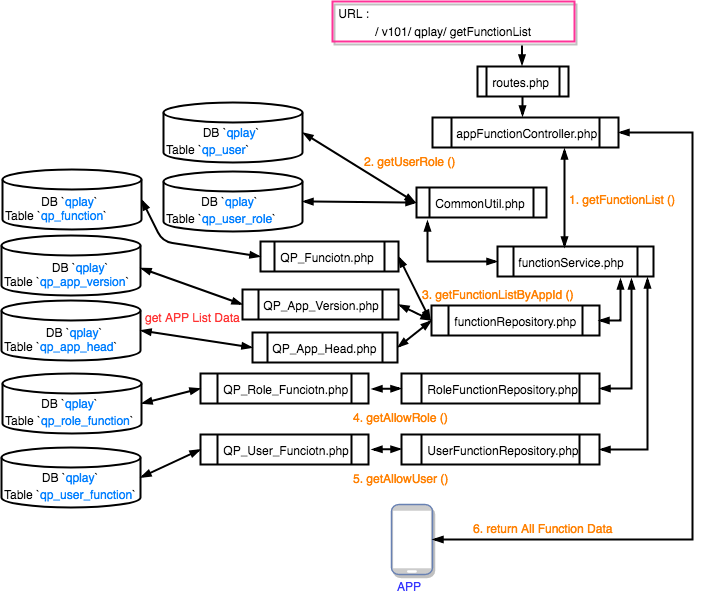
* QPlay 後台新增功能 [Function Manage] - Function Detail Data



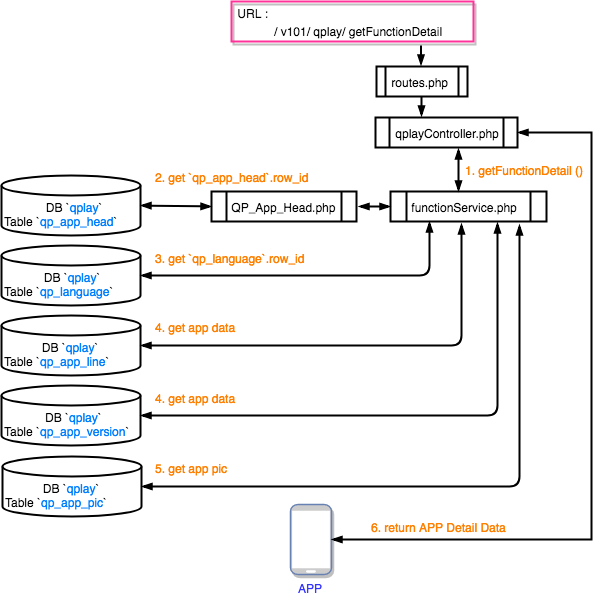
* QPlay 後台新增功能 [Function Manage] - Function Edit



* API getFunctionList



* API getFunctionDetail



4.3.12 程式碼修改記錄

1. qplayApi/ qplayApi/
   1. app/ Http/ **routes.php**
      1. 新增 URL Request Handle : getFunctionList，經由 ~~qplayController.php~~

appFunctionController.php 處理，取得 User 在此 APP 的所有 Function List。

* + 1. 新增 URL Request Handle : getFunctionDetail，經由 ~~qplayController.php~~

appFunctionController.php 處理，取得 User 在此 APP ，單一個 Function 的詳細資料。

* 1. app/ Http/ Controllers/ **~~qplayController.php~~ appFunctionController.php**
     1. 依據 URL Request，決定如何使用 FunctionService.php。
     2. 定義 getFunctionList ()。
     3. 定義 getFunctionDetail ()。
  2. app/ lib/ **CommonUtil.php**
     1. ~~比照 qplay/ app/ lib/ CommonUtil.php :~~
        1. ~~新增 getUserEnableInfoByUserID ()~~
           1. ~~取得此 User 目前屬於哪些 企業角色 Role。~~
           2. ~~不需要取得以下資料 :~~

~~`qp\_user\_group`~~

~~`qp\_group\_menu`~~

~~`qp\_user\_menu`~~

* + 1. 新增 function getUserRole ()
       1. 取得此 User 目前屬於哪些企業角色 Role
    2. 新增 function ADFlag ()
       1. 依據此 User 這次登入的 UUID，找出這次連線的 ad\_flag
  1. app/ Model/
     1. **QP\_Function.php**
        1. 定義 DB `qplay` Table `qp\_function`。
     2. **QP\_Role\_Function.php**
        1. 定義 DB `qplay` Table `qp\_role\_function`。
     3. **QP\_User\_Function.php**
        1. 定義 DB `qplay` Table `qp\_user\_function`。
  2. app/ Repositoreies/
     1. **FunctionRepository.php**
        1. 定義 ~~getFunctionByApp ()~~ getFunctionListByAppId ()
           1. 依據 API 傳入的 APP Key 找到對應的 `qp\_app\_head`.row\_id，

在 `qp\_function` 找到相對應的 Function 列表。

* + - * 1. 依據 API 傳入的 UUID，找到 `qp\_user`.ad\_flag，判定此 User 是否屬於 QAccount 用戶，決定此 User 是否

套用 `qp\_function`.qaccount\_use 這個資料欄位的設定值。

* + - * 1. ex:

Function [ News ] 的 `qp\_function`.qaccount\_use = N，如果是一般用戶，因為 ad\_flag = N，所以還是有權限可以使用 [ News ] ; 如果是 QAccount 用戶，因為 `qp\_user`.ad\_flag = Y，所以此 User 必須套用 qaccount\_user = N，沒有權限可以使用 [ News ]。

* + 1. **~~FunctionRoleRepository.php~~ RoleFunctionRepository.php**
       1. 定義 ~~getFunctionRoleData ()~~ getAlowRole ()
          1. 依據 ~~getUserEnableInfoByUserID ()~~ getUserRole () 的結果，透過

role\_row\_id 以及 funciton\_row\_id 在 `qp\_role\_function` 取得資料。

* + 1. **~~FunctionUserRepository.php~~ UserFunctionRepository.php**
       1. 定義 ~~getFunctionUserData ()~~ getAllowUser ()
          1. 依據 User 的 `qp\_user`.row\_id 在 `qp\_user\_function` 取得資料。
  1. app/ Services/ **FunctionService.php**
     1. 定義 ~~getFunctionList ()~~ getFunctionListByAppId ()。
        1. 依據權限的權重，由小至大，進行下列流程處理資料，並且回傳 :
           1. FunctionRepository.php 的 ~~getFunctionByApp ()~~ getFunctionListByAppId () 取得此 APP 的

全部 Function，依據 `qp\_function`.qaccount\_right\_level 決定權重大小

* `qp\_function`.qaccount\_right\_level = 1
  + QAccount < [ 公司 ]
* `qp\_function`.qaccount\_right\_level = 2
  + [ 公司 ] < QAccount
    - * 1. 依據 APP 端傳入的 login\_type = AD or QAccount，決定此 User 是否為 QAccount 的使用者。
        2. 如果 QAccount < [ 公司 ]。
        3. FunctionRepository.php 的 ~~getFunctionByApp ()~~ getFunctionListByAppId () 取得此 APP 的

全部 Function，假設共有 3 個，預設全部權限皆為 N，放入陣列，

陣列的 key = `qp\_function`.variable\_name，

Array[ FunA : {right : N, type : APP}, FunB : {right : N, type : FUN}, FunC : {right : N, type : FUN}]。

* + - * 1. 檢查 User 是否為 QAccount 用戶，依據此 Function 的

`qp\_function`.qaccount\_use 決定使用權限。

* + - * 1. **~~FunctionRoleRepository.php~~ RoleFunctionRepository.php** 的 ~~getFunctionRoleData ()~~ getAlowRole () 取得

Role 和 Function 之間的使用權限設定值，如果 User 所屬的 Role 可以使用 FunB，則陣列被更新為

Array[ FunA : {right : N, type : APP}, FunB : {right : Y, type : FUN}, FunC : {right : N, type : FUN}]。

* + - * 1. **~~FunctionUserRepository.php~~ UserFunctionRepository.php** 的 ~~getFunctionUserData ()~~ getAllowUser () 取得

User 和 Function 之間的使用權限設定值，如果 User 被設定可以使用 FunC，則陣列被更新為

Array[ FunA : {right : N, type : APP}, FunB : {right : Y, type : FUN}, FunC : {right : Y, type : FUN}]。

* + - * 1. 最後，將處理過後的陣列資料，回傳至 APP 端，APP 端依據

variable\_name 的 right = Y / N，決定相對應的功能，是否可以使用。

* + - * 1. 承上，如果 [ 公司 ] < QAccount，則修改上面的檢查順序。
      1. 承上，如果要考慮加入 APP List :
         1. 執行 SQL 的時候，DB Table `qp\_function` 需要

LEFT JOIN DB Table `qp\_app\_head`，取得

`qp\_app\_head`.package\_name 和 `qp\_app\_head`.icon\_url。

* + - * 1. 此時的陣列資料會是下列格式 :

Array[ FunA : {right : N, type : APP, packageName : apprrs, icon : icon\_512.png}, FunB : {right : Y, type : FUN}, FunC : {right : Y, type : FUN}]。

* + - * 1. 承上，APP QPlay 可以在 APP List，依據 APP Name 和 Icon 顯示 APP 資料。
    1. 定義 getFunctionDetail ()。
       1. getFunctionList () 會告知 APP 端各個 Function 的 Type = FUN / APP，如果是 APP， 則透過此 API getFunctionDetail ()，取得 APP 的詳細資料。
       2. 依據 APP 端傳入的資料 packageName，透過 `qp\_app\_head` 找到對應

的 row\_id。

* + - 1. 使用 `qp\_app\_head`.row\_id 在 `qp\_app\_head`，`qp\_app\_line`，

`qp\_language`，`qp\_app\_version`，`qp\_app\_pic` 找到相對應的資料，整合之後回傳給 APP 端。

* + - 1. 承上，APP 的下載路徑，必須依據 `qp\_app\_line`.external\_app 的值，決定是否需要使用 FilePath::getApkDownloadUrl () 做處理，可以參考 qplayController.php。
      2. 承上，`qp\_app\_line` 和 `qp\_app\_pic` 的處理邏輯 :
         1. APP 端 Call API 時會在 URL 帶入 lang 參數，透過此參數

取得 `qp\_language`.row\_id 。

* + - * 1. 處理 `qp\_app\_head` 的時候，可以取得 default\_lang\_row\_id。
        2. 將 `qp\_app\_line` 和 `qp\_app\_pic` 中屬於此 APP 的資料都撈出來。
        3. 比對是否有 API 帶入的 lang 語系，回傳的資料中，只需要包含此語係。
        4. 如果 API 帶入的 lang 語系沒有資料存在，則使用 default\_lang\_row\_id 的資料做回傳。
        5. 總之，每次只會回傳一種語系的資料。
      1. 承上，由於之前 getAppList () 是直接在 qplayController.php 執行 SQL 撈取

`qp\_app\_head`，`qp\_language`，`qp\_app\_line`，`qp\_app\_version`，`qp\_app\_pic` 這些 Table 的資料，沒有依據這些 Table 額外新增

相關的 Repository 跟 Model，這次同樣依照此邏輯處理。

1. qplay/
   1. app/ Http/ **routes.php**
      1. 新增 URL Request Handle : functionMaintain，經由 functionController.php 處理

[ Function Manage ] 的 Function List 和 Function Add。

* + 1. 新增 URL Request Handle : functionDetailMaintain，經由 functionController.php 處理

[ Function Manage ] 的 Function Data Display 和 Function Edit。

* 1. app/ lib/ **ResultCode.php**
     1. 針對 [ Function Manage ] 的資料維護結果，新增 Result code。
        1. Function Variable Name 已經存在此 APP。
  2. resources/ lang
     1. en/ **messages.php**
     2. zh-cn/ **messages.php**
     3. zh-tw/ **messages.php**
        1. 三種語系，依據 UI 需求，增加對應的文字變數。
  3. app/ Model/
     1. **QP\_Function.php**
        1. 定義 DB `qplay` Table `qp\_function`。
     2. **QP\_Role\_Function.php**
        1. 定義 DB `qplay` Table `qp\_role\_function`。
     3. **QP\_User\_Function.php**
        1. 定義 DB `qplay` Table `qp\_user\_function`。
  4. app/ Repositories/
     1. **FunctionRepository.php**
        1. 定義 getFunctionList ()
           1. 回傳 Function 列表。
        2. 定義 getFunctionDatal ()
           1. 回傳 Function 詳細資料。
        3. 定義 newFunctionData ()
           1. 新增 Function 資料。
        4. 定義 editFunctionData ()
           1. 編輯 Function 資料。
     2. **FunctionRoleRepository.php**
        1. 定義 getFunctionRoleData ()
           1. 回傳 Role - Function 的使用權限。
        2. 定義 checkFunctionRole ()
           1. 檢查 Role - Function 是否已經存在。
        3. 定義 newFunctionRoleData ()
           1. 新增 Role - Function 使用權限。
        4. 定義 editFunctionRoleData ()
           1. 編輯 Role - Function 的使用權限。
     3. **FunctionUserRepository.php**
        1. 定義 getFunctionUserData ()
           1. 回傳 User - Function 的使用權限。
        2. 定義 checkFunctionUser ()
           1. 檢查 User - Function 是否已經存在。
        3. 定義 newFunctionUserData ()
           1. 新增 User - Function 的使用權限。
        4. 定義 editFunctionUserData ()
           1. 編輯 User - Function 的使用權限。
  5. app/ Http/ Controllers/ **FunctionController.php**
     1. 依據 URL Request，決定如何使用 FunctionService.php。
     2. 定義 functionMaintain ()。
     3. 定義 functionDetailMaintain ()。
     4. 定義 getFunctionList ()。
     5. 定義 newFunctionData ()。
     6. 定義 editFunctionData ()。
  6. app/ Services/ **FunctionService.php**
     1. 定義 getFunctionList () :
        1. 透過 FunctionRepository.php 取得 Function 資料 做列表呈現。
     2. 定義 newFunctionData () :
        1. 透過以下檔案，新增 Function 資料 :
           1. FunctionRepository.php
           2. AppService.php

使用 getAppList() 取得 APP List，每個 Function 綁定單一個 APP Key。

* + 1. 定義 editFunctionData () :
       1. 透過以下檔案，編輯 Function 資料 :
          1. FunctionRepository.php

顯示 / 編輯 Function 基本資料。

* + - * 1. FunctionRoleRepository.php

顯示 / 編輯 Role - Function 的使用權限。

如果 Role - Function 已經存在，~~改為變更 Status~~ Delete 資料，重新 Insert 資料。

* + - * 1. FunctionUserRepository.php

顯示 / 編輯 User - Function 的使用權限。

如果 User - Function 已經存在，~~改為變更 Status~~ Delete 資料，重新 Insert 資料。

* + - * 1. AppService.php

使用 getAppList() 取得 APP List，每個 Function 綁定單一個 APP Key。

* + - * 1. CommonUtil.php

使用 getAllCompanyRoleList() 取得 Role List，設定每個 Role - Function 的使用權限。

* 1. resources/ views/ **function\_maintain**/
     1. **function\_list.blade.php**
        1. Function 資料列表 :
           1. 使用 Bootstrap Table 產生資料列表。
           2. 顯示 APP Key / `qp\_function`.name / `qp\_function`.type

/ `qp\_function`.variable\_name / `qp\_function`.status 。

* + - 1. Function 資料新增 :
         1. 使用 Dialog UI 進行資料的新增。

透過下拉選單，選擇此項目是屬於哪個 APP Key 底下的項目。

選擇類型 Function / Appliction。

如果是 Application，另外出現下拉選單，必須選擇此

項目是對應到哪一個 APP。

填入 Name / Variable Name / Description。

* + - * 1. DB Table `qp\_function`.qaccount\_use 預設為 N。
    1. **function\_detail.blade.php**
       1. 顯示 Function 詳細資料，進行編輯。
       2. `qp\_funciton`.company\_label 的資料處理，比照 APP 管理的邏輯，參考以下程式檔案 :
          1. qpaly/ resources/ views/ app\_maintain/ app\_detail/ info.blade.php
          2. qpaly/ resources/ views/ app\_maintain/ app\_detail/ main.blade.php
          3. qpaly/ app/ Http/ Controllers/ AppMaintainController.php
       3. UI 由上至下排列，依序設定下列項目:
          1. 下拉選單 - QAccount 權限等級

option.value = 1 , option.text = QAccount < 公司

option.value = 2 , option.text = 公司 < QAccount

* + - * 1. 下拉選單 - 用戶設定

option.value = 1 , option.text = 依公司選擇

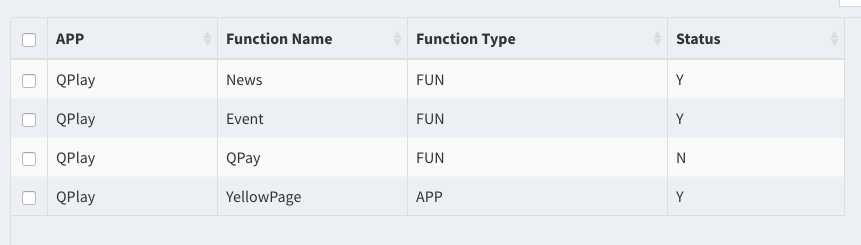
option.value = 2 , option.text = 依企業角色選擇

* + - * 1. 下拉選單 - QAccount 的使用權限。
        2. 企業角色 Role 的使用權限，透過 checkbox 設定 (當用戶設定 = 2)。
        3. 單一使用者的使用權限，透過 checkbox 設定 (當用戶設定 = 2)。
      1. 修正資料操作邏輯
         1. 功能名稱 - Function Name : 可以編輯修改
         2. 功能 ID - Function ID : 不可以編輯修改

1. QPlay Backend Platform
   1. [ 系統管理 ] > [ 選單維護 ] > [ 新增 ]
      1. 菜單名 : FUNCTION\_MAINTAIN
      2. 英文名 : Function Maintain
      3. 簡體中文名 : 功能管理
      4. 繁體中文名 : 功能管理
   2. [ 系統管理 ] > [ 選單維護 ] > 點擊 ( **功能管理** ) > [ 新增 子選單 ]
      1. 菜單名 : FUNCTION\_MAINTAIN
      2. 鏈結 : functionMaintain
      3. 英文名 : Function Maintain
      4. 簡體中文名 : 功能管理
      5. 繁體中文名 : 功能管理
   3. [ 系統管理 ] > [ 群組維護 ]
      1. Administrator : 預設有最高權限，可以使用全部群組。
   4. 上述動作完成之後，必須修改以下檔案，以 (菜單名) 為變數名稱，建立相對應的資料
      1. qplay/ resources/ lang/ en/ **messages.php**
         1. 'FUNCTION\_VARIABLE\_NAME' => 'Function ID'
      2. qplay/ resources/ lang/ zh-cn/ **messages.php**
         1. 'FUNCTION\_VARIABLE\_NAME' => '功能 ID'
      3. qplay/ resources/ lang/ zh-tw/ **messages.php**
         1. 'FUNCTION\_VARIABLE\_NAME' => '功能 ID'
      4. 新增
         1. 'TITLE\_FUNCTION\_MAINTAIN' => Function Maintain'
   5. 最後，要看 User 的操作權限設定選項中的 [功能管理] 是否被勾起。
      1. 備註 : 就算是 Administrator 群組，個人的操作權限不一定跟群組一樣。

4.3.13 UI 示意圖

* Function Manage - Function List



* Function Manage - Function Add



* Function Manage - Function Edit



