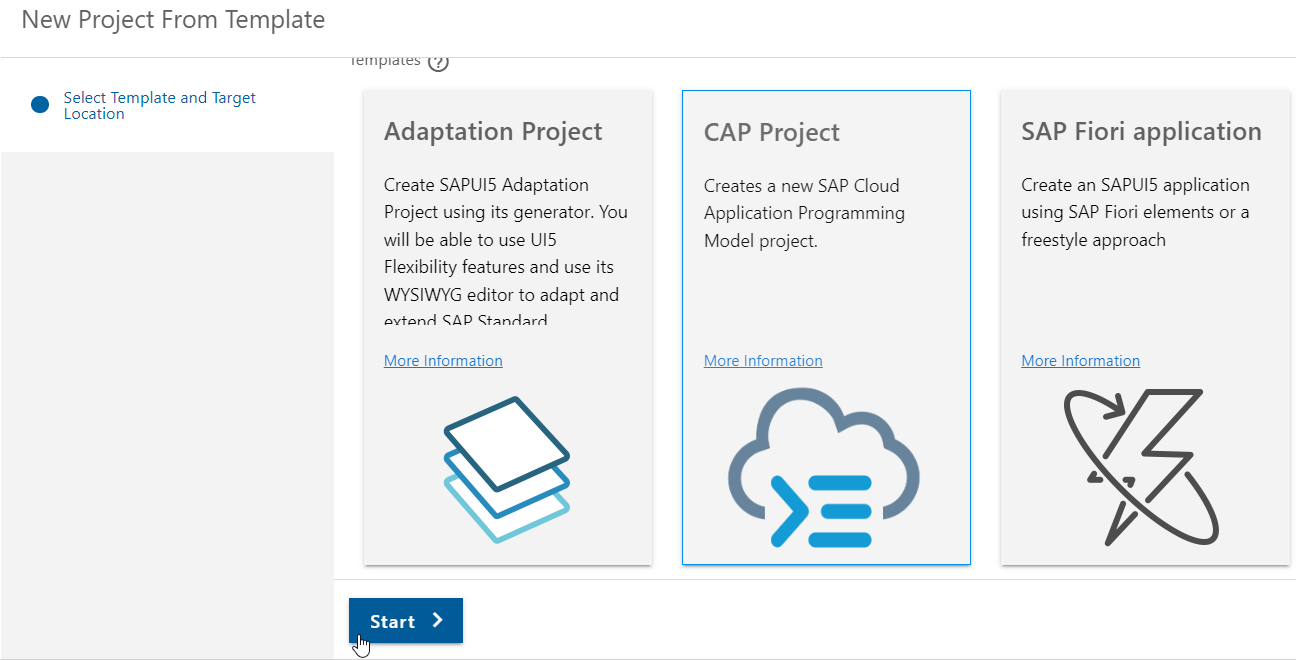
**ML2200’s BTP 開發CAP/UI5 學習手冊**

**目錄**

1. **BTP CAP 開發**
2. **BTP UI5 開發**
3. **打包與配置BTP**
4. **GitLab 使用**
5. Create CAP

使用New Project From Template 創建CAP Project



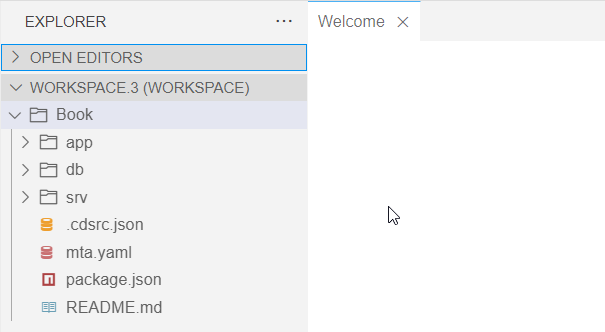
2

1

填入CAP Project Details 之後選擇Finish, BAS 自動生成Project（select your runtime选JAVA）

一張含有 文字 的圖片

自動產生的描述



1. Create data-model.cds in folder db（默认情况有这个文件，改成我们需要的namespace和entity即可）

一張含有 文字 的圖片

自動產生的描述

1. create cat-service.cds in folder srv（using为data-model的namespace名字）

一張含有 文字 的圖片

自動產生的描述

1. 定義您要的schema name 在mta.yaml 檔案裡的resource 自行添增config : schema: (如下圖)

一張含有 文字 的圖片

自動產生的描述

1. 開發階段可以先使用csv file 來預覽(放到db-data文件夹底下)，csv file naming rune 為

data-model 裡的namespace-entity (ex: big\_class-zccs\_big\_class.csv)

1. Open terminal
   1. Type cds watch, then 開啟pre-view web
   2. 如果這時候系統出現ERROR for dependencies 可以輸入

npm install -g @sap/cds

-g 後面放缺少的package (我練習時出現缺少"@sap/cds" "express" "hdb" "sqlite3" 或版本不符)

一張含有 文字 的圖片

自動產生的描述

* 1. 可以看到Odata (資料來自db/data/big\_class-zccs\_big\_class.csv)

一張含有 文字 的圖片

自動產生的描述

也可以透過[Fiori preview](https://port36641-workspaces-ws-xx66z.us10.trial.applicationstudio.cloud.sap/$fiori-preview/view_zccscap_big_class/zccs_big_class" \l "preview-app" \o "Preview in Fiori elements) 來檢視odata

一張含有 文字 的圖片

自動產生的描述

1. Deploy Cloud Foundry
   1. 打包CAP

透過CAP terminal 輸入mbt build

* 1. 跑完打包流程BAS 會自動建立一個目錄mta\_archives, 並且產生檔案\*.mtar(本範例產生zccscap\_big\_class\_1.0.0.mtar)

一張含有 文字 的圖片

自動產生的描述

* 1. Login Cloud Foundry
     1. 可以使用指令登入

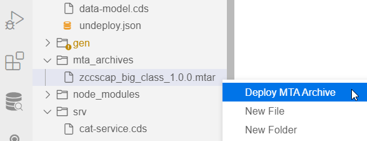
cf api https://api.cf.us10.hana.ondemand.com

cf login

* + 1. 或使用find command 登入(輸入帳號、密碼)
  1. Deploy
     1. 使用terminal 輸入 cf deploy \*.mtar (mtar檔案路徑+檔名)

cf deploy mta\_archives/zccscap\_big\_class\_1.0.0.mtar

* + 1. 或透過BAS UI 在mta\_archives目錄底下選擇剛剛打包好的mtar 檔按右鍵選擇Deploy



1. 設定container
   1. Deploy 之後可以在HANA DB 看到系統幫忙建立的table (Schema 沒有指定會由系統自行編)

一張含有 桌 的圖片

自動產生的描述

* 1. 但是如果要直接open data 系統會出現錯誤無法query

一張含有 文字 的圖片

自動產生的描述

* 1. 此時需要設定container 把剛剛建好的database 加到explorer

一張含有 文字 的圖片

自動產生的描述

* 1. Database type 要選HDI Container, 然後選這次Create DB

一張含有 文字 的圖片

自動產生的描述

* 1. 這時候再回去open Data 就可以直接瀏覽新建的table 內容了

一張含有 桌 的圖片

自動產生的描述

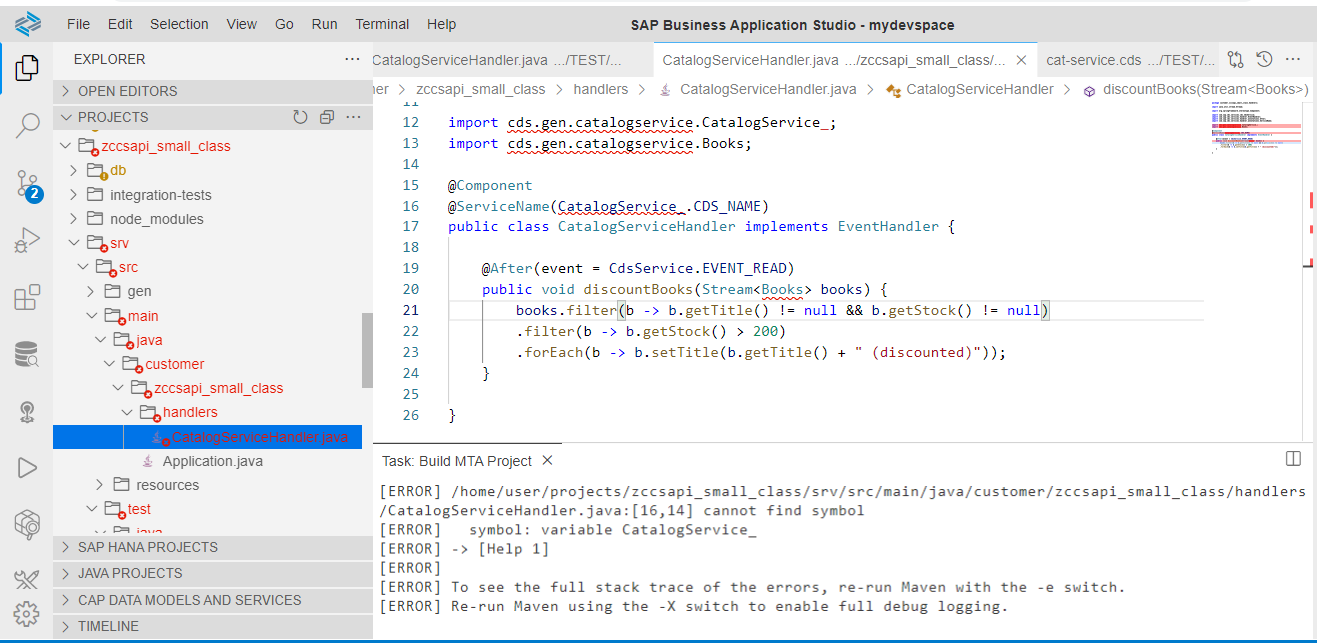
* 1. 透過瀏覽器輸入SRV 提供的網址，可以看到DB 內的資料，表示這個CAP 連結ODATA service 功能已經ready in BTP (如果瀏覽器沒有Parsed 功能會出現下面結果)

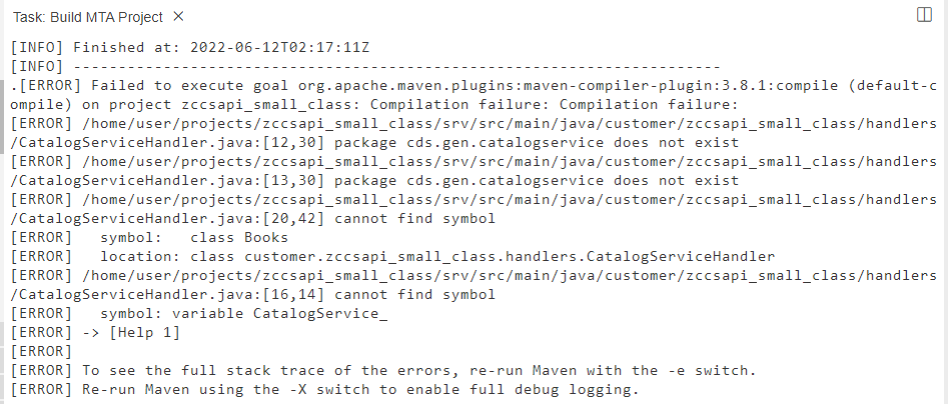
一張含有 文字 的圖片

自動產生的描述

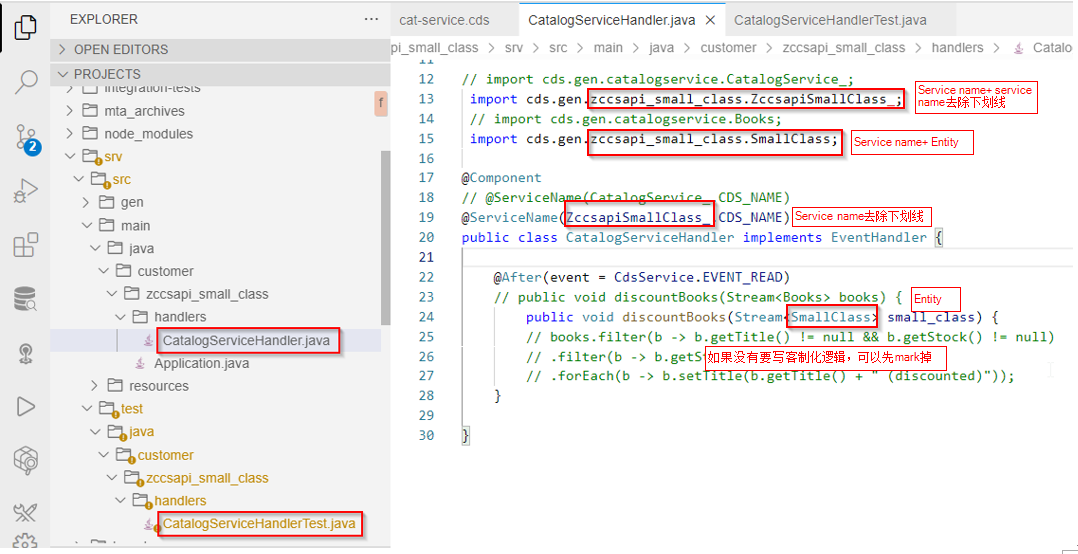
接下來就要透過UI5 介面提供user 操作介面了。

參考網址：[https://18c571abtrial-trail-aneo7gd3-dev-zccscap-big-class-srv.cfapps.us10.hana.ondemand.com](https://18c571abtrial-trail-aneo7gd3-dev-zccscap-big-class-srv.cfapps.us10.hana.ondemand.com/)

* 1. 按以上step做完后，build project会报错（可能是选择JAVA的原因，不确定Node.js会不会报错）。



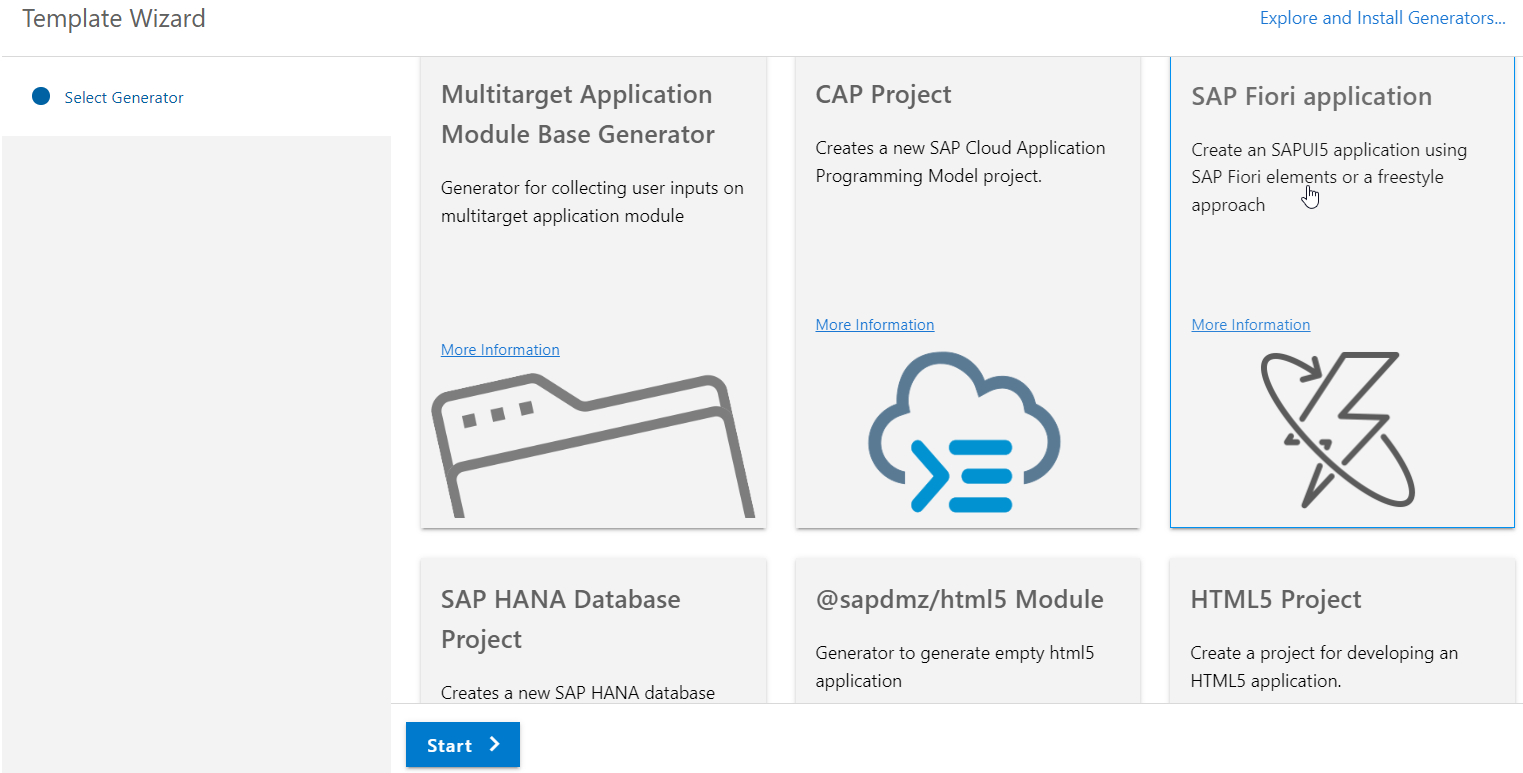
原因是我们是选择“Basic Sample Files”创建project，所以系统会默认创建Books table，同时会创建一些sample 代码在service handler的JAVA文件。需要做相应的修改。



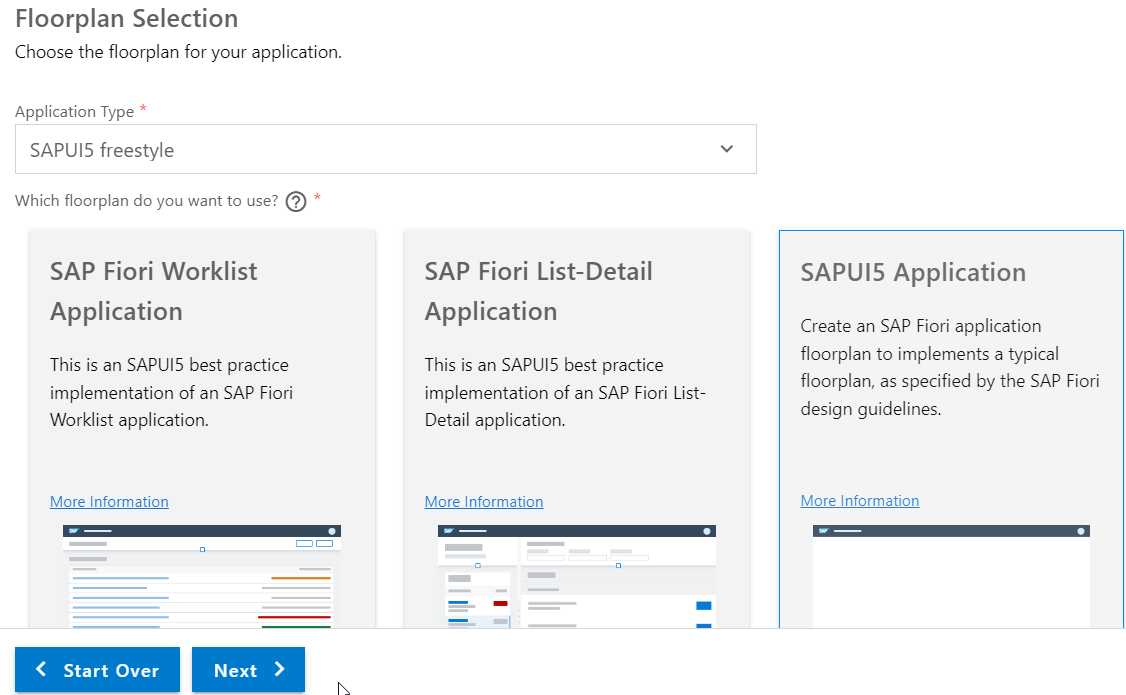


1. Create UI5

透過BAS template SAP Fiori application建立初始project (步驟可以參考下面步驟)



剛開始建議使用SAPUI5 freestyle + SAPUI5 Application(空白底稿), 多加熟悉程式語法



不用選擇Data Source , 我們可以在程式碼內自建datasource

一張含有 文字 的圖片

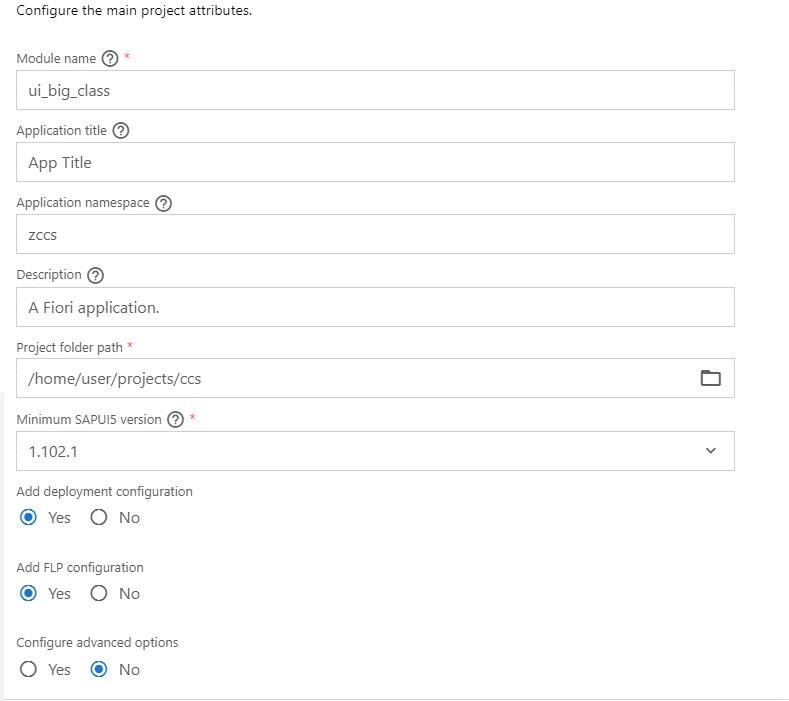
自動產生的描述

輸入project View name(通常會放上function name 此範例為CCS 的BigClass)

一張含有 文字 的圖片

自動產生的描述

輸入Project attributes.



先輸入namespace : zccs (此範例以CCS 2.0 project 為例)

接著輸入Module name : ui\_big\_class (project naming rule zccsui\_big\_class 這邊取後半段ui\_big\_class)

Project path 選擇在project/ccs 底下

選項勾選

Add deployment configuration 🡪 可以自動產生mta.yaml 檔案提供打包/deploy 參數配置

Add FLP configuration 🡪 可以自動產生Fiori Launchpad rout 參數配置

其他Description/Title 可以自行定義

維護deployment configuration 參數設定

一張含有 文字 的圖片

自動產生的描述

因為我們BTP 背後使用的是Cloud Foundry 最後這些project 都要deploy 到Cloud Foundry

所以

Target 選擇Cloud Foundry

Destination 配置先選擇 None

維護Fiori Launchpad 設定

一張含有 文字 的圖片

自動產生的描述

這三個欄位除了Action 一定要用Display 之外，其他object/title 還沒有命名規則。

如果在create project by template 的時候，忘了勾選Add FLP config 那就不會有這一頁設定。

不用擔心在project 建立之後也可以在webapp/manifest.json 檔案的dataSources 下面加上crossNavigation 程式片段(記得修改project name)

"crossNavigation": {

            "inbounds": {

                "zccs-uibigclass-inbound": {

                    "signature": {

                        "parameters": {},

                        "additionalParameters": "allowed"

                    },

                    "semanticObject": "BigClassSemantic",

                    "action": "Display",

                    "title": "{{flpTitle}}",

                    "subTitle": "{{flpSubtitle}}",

                    "icon": ""

                }

            }

        }

檔案維護:

**Xs-app.json**

新增source for Odata service, 內容可以參考對應CAP 服務的endpoints(如下範例為/odata/v4/MasterDataService/ )，destination 就放Odata service name (MasterDataService)

{

        "source": "^/odata/v4/MasterDataService/(.\*)$",

        "target": "/odata/v4/MasterDataService/$1",

        "authenticationType": "none",

        "destination": "MasterDataService"

    },

一張含有 文字 的圖片

自動產生的描述

另外一個sample 為/odata/v4/ApproveDataService/

{

        "source": "^/odata/v4/ApproveDataService/(.\*)$",

        "target": "/odata/v4/ApproveDataService/$1",

        "authenticationType": "none",

        "destination": "ApproveDataService"

    },

一張含有 文字 的圖片

自動產生的描述

**Modify Ui5.yaml**

新增path in backend

- path: /odata/v4/MasterDataService

            url: <https://scpgatw-wistronprework-sdev-zccsapi-financial-data-srv.cfapps.eu10.hana.ondemand.com>

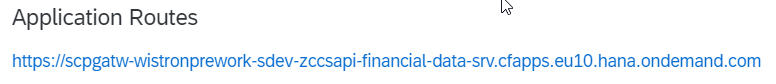
Path 來源參考OData endpoints data service

一張含有 文字 的圖片

自動產生的描述

Path

url 來源參考這個UI5要引用的CAP(zccsapi\_financial\_data-srv)在[BTP Cockpit](https://account.hana.ondemand.com/cockpit) CAP Application Routes



url

另外一個sample

backend:

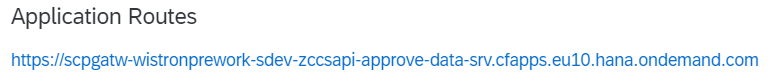
          - path: /odata/v4/ApproveDataService

            url: https://scpgatw-wistronprework-sdev-zccsapi-approve-data-srv.cfapps.eu10.hana.ondemand.com

一張含有 文字 的圖片

自動產生的描述

Path



url

**Modify manifest.json**

1. 新增DataService in dataSources

MasterDataService (參考Ui5.yaml path 服務名稱)

"MasterDataService": {

                "uri": "/odata/v4/MasterDataService/",

                "type": "OData",

                "settings": {

                    "odataVersion": "4.0"

                }

            }

"MasterDataService": {

                "uri": "/odata/v4/MasterDataService/",

                "type": "OData",

                "settings": {

                    "odataVersion": "4.0"

                }

            }

另一個UI5 sample : ApproveDataService (參考Ui5.yaml path 服務名稱)

Uri : 取OData service name(/odata/v4/ApproveDataService)

"ApproveDataService": {

                "uri": "/odata/v4/ApproveDataService/",

                "type": "OData",

                "settings": {

                    "odataVersion": "4.0"

                }

}

"ApproveDataService": {

                "uri": "/odata/v4/ApproveDataService/",

                "type": "OData",

                "settings": {

                    "odataVersion": "4.0"

                }

            }

1. 新增ServiceModel in sap.ui5 底下的models, 定義service model(取 DataService名稱+Model)

datasource 參考Xs-app.json 宣告的destination(MasterDataService)

"MasterDataServiceModel": {

"MasterDataServiceModel": {

                "dataSource": "MasterDataService",

                "preload": true,

                "settings": {

                  "synchronizationMode": "None",

                  "operationMode": "Server",

                  "autoExpandSelect": true,

                  "earlyRequests": true

                }

},

                "dataSource": "MasterDataService",

                "preload": true,

                "settings": {

                  "synchronizationMode": "None",

                  "operationMode": "Server",

                  "autoExpandSelect": true,

                  "earlyRequests": true

                }

              },

另一個UI5 sample : ApproveDataServiceModel

datasource 參考Xs-app.json 宣告的destination(ApproveDataService)

" ApproveDataServiceModel": {

" ApproveDataServiceModel": {

                "dataSource": " ApproveDataService",

                "preload": true,

                "settings": {

                  "synchronizationMode": "None",

                  "operationMode": "Server",

                  "autoExpandSelect": true,

                  "earlyRequests": true

                }

              },

                "dataSource": " ApproveDataService",

                "preload": true,

                "settings": {

                  "synchronizationMode": "None",

                  "operationMode": "Server",

                  "autoExpandSelect": true,

                  "earlyRequests": true

                }

              },

**Modify ApproveAmountLimit.view.xml**

(本文件指出需要修改或注意的部分，詳細內容可以參考Gitlab 上其他UI5 程式碼)

修改 webapp/view 底下的ApproveAmountLimit.view.xml

1. 在MVC:view 增加定義會使用到的library(本sample 使用ui 與layout.form)

xmlns:core="sap.ui.core"

xmlns:f="sap.ui.layout.form"

1. 展開content 內容
   1. 定義Table
      1. 宣告items 來源 Path 放(servicemodel >/ Entity)

Path 'ApprovDataServiceModel>/ApproveAmountLimits',

Parameters $$updateGroupId : 'ApproveAmountLimitsGroup'()

* 1. 定義 headerToolbar
     1. 因為headerToolbar 有放一個search 欄位所以宣告<SearchField>

id="idSearchFieldLevel" (這邊以要搜尋欄位命名例如Level)

placeholder="{i18n>searchFieldPlaceholderLevel}"

* + 1. 宣告與使用Button(功能按鈕)

New/Delete/Refresh 🡨 如沒特別需求直接COPY 使用

* 1. 定義columns(欄位)
     1. 針對Entity 每個欄位定義

<Column id="idDocument\_type">

             <Text

                 id="idTextDocument\_type"

                 text="{i18n>Document\_type}"

             />

         </Column>

* 1. 定義items(Entity records)
     1. 會呈現多筆records 所以宣告<ColumnListItem id="idColumnListItem">
     2. 針對每個欄位宣告從DataSourceModel對應的欄位名稱

Value 放DataServiceModel > 對應entity欄位(小寫)

<cells>

               <Input

                   value="{ApproveDataServiceModel>document\_type}"

                   liveChange=".onInputChange"/>

         </cells>

1. 宣告頁腳<footer>
   1. 這邊定義新增、修改、刪除record 之後要進行SAVE 還是Cancel🡨 沒有特別需求就直接COPY使用

**Modify ApproveAmountLimit.controller.js**

修改 webapp 底下的ApproveAmountLimit.controller.js

1. 在sap.ui.define 宣告會使用到的library(本sample 使用下列)

"sap/m/MessageToast",

    "sap/m/MessageBox",

    "sap/ui/model/Filter",

    "sap/ui/model/FilterOperator",

    "sap/ui/model/json/JSONModel"

1. 在function 宣告會使用到的MessageToast, MessageBox, Filter, FilterOperator, JSONModel
2. 宣告每一個觸發的控制項(如新增、刪除、修改) 🡨 沒特別需求可以直接COPY 使用

但要特別注意DataServiceModel 需跟前面定義的匹配

**Modify i18n.properties in webapp/i18n 目錄**

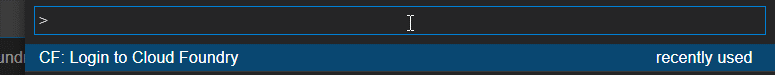
這個檔案可以針對UI5 上的各個按鈕/Title 進而翻譯多國語言

(如果有仔細分類不同語言的話系統會依照瀏覽器的語系自動呼叫對應的語系)

Deploy

1. 登入cloud

View/Find Command



prework : https://api.cf.eu10.hana.ondemand.com

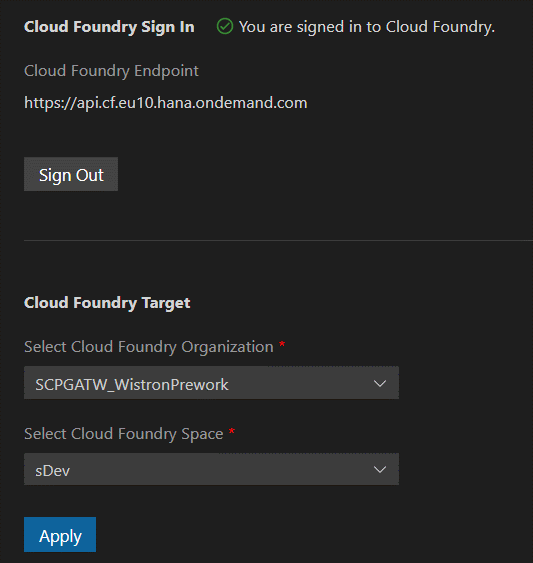
一張含有 文字, 螢幕擷取畫面, 監視器, 螢幕 的圖片

自動產生的描述

-輸入帳號

-輸入密碼

點選Sign in , 成功後會出現下面畫面

  
-選擇登入的Organization

prework : SCPGATW\_WistronPrework

-選擇登入的Space

prework : sDev

1. 打包CAP

方式一：在終端機輸入指令mbt build

方式二：使用UI 選項

一張含有 文字 的圖片

自動產生的描述

右鍵點選mta.yaml 🡪出現選單🡪 選擇Build MTA Project🡪系統便開始打包MTA

1. Deploy

方式一：在終端機輸入指令cf deploy \*.mtar

方式二：使用UI 選項

在剛剛mta打包後產生一個新目錄mta\_archives裡面的zccs-uiproductiontypes\_0.0.1.mtar 按右鍵🡪選擇Deploy MTA Archive，系統便開始Deploy

一張含有 文字 的圖片

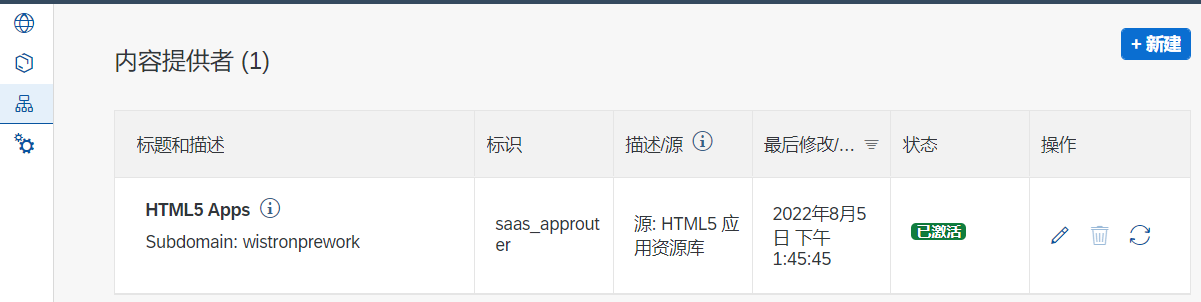
自動產生的描述

配置站台APP

登入帳點管理器，點選左側menu – 内容提供者



在”内容提供者” 點選更新內容，因為剛deploy 上的UI5並不會自動update，選擇更新後剛剛Deploy 的APP 才會出現在HTML5 Apps 清單



點選內容管理器  並選擇內容瀏覽器

一張含有 文字 的圖片

自動產生的描述

搜尋剛剛deploy 的APP，勾選項目後按下

一張含有 文字 的圖片

自動產生的描述

這時候再回到我的內容頁簽就可以看到剛剛加進來的APP

一張含有 文字 的圖片

自動產生的描述

再來我們必須要把這個APP assign 權限，在pre-work 階段我們全部先放到every 這個角色



進到角色 Everyone 先選擇編輯，再打開分配面板按鈕

一張含有 文字 的圖片

自動產生的描述

這時候可以看到目前已經有assign 給Everyone 的APP 在右側面板, 但這時候找不到我們剛剛上船的APP，請先直接按放大鏡(或輸入上傳的APP name 再按放大鏡)

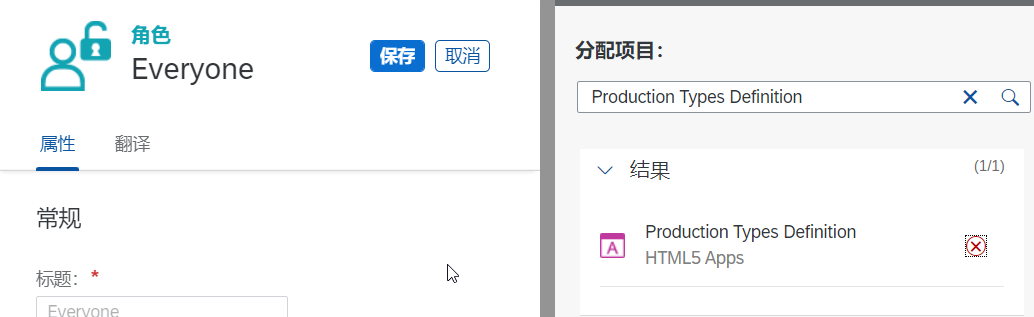
一張含有 文字 的圖片

自動產生的描述

搜尋後可以看到剛剛Deploy 的APP 出現在結果，並且APP 旁邊有個 **+** 號



按下 + 表示把這個APP assign 給這個角色使用，符號也會變成紅色，按下保存



這樣這個角色就可以使用新的APP ，同樣的動作我們再重複群組CCS 設定。

一張含有 文字 的圖片

自動產生的描述

這樣這個APP(Production Types Definition) 就會歸類到 Launchpad CCS 群組底下

一張含有 桌 的圖片

自動產生的描述

按下這個APP 就可以呼叫(執行)，我們也可以透過上傳後驗證城市是否正確。

一張含有 文字 的圖片

自動產生的描述