zeliang_python_road Documentation

Release 1.0

Zeliang YAO

Mar 02, 2019

Contents

Conclusion

1. Installation & Test

We take about two weeks to create and improve the documentation for both business users and developers, also test the installation of software used in environment **local** and **VDI**

1.1 Update Documentation

>**Updated Files**

- Installation_kit–DEV_USER–v4.docx
- Installation_kit–BUSINESS_USER–v4.docx
- Screenshots for docs
- Location: ***V:RESGBSCDASH-PREPGDOadminhow-to guidesDRAFTS***

>Modification Log****

- Modification.xlsx
- Location: ***V:RESGBSCDASH-PREPGDOadminhow-to guidesDRAFTS***

1.2 Software Installed

<table border="1" class="docutils"> <thead> <tr> <th>Software</th> <th>Business User</th> <th>Develop User</th> <th>Current Version</th> </tr> </thead> <tbody> <tr> <td>Anaconda (Python)</td> <td></td> <td></td> <td>Anaconda2-5.3.0-Windows-x86_64</td> </tr> <tr> <td>Json Editor</td> <td></td> <td></td> <td>-</td> </tr> <tr> <td>Sublime Text</td> <td></td> <td></td> <td>3176 portable version</td> </tr> <tr> <td>Atom</td> <td></td> <td></td> <td>1.34.0</td> </tr> <tr> <td>Win Merge</td> <td></td> <td></td> <td>2.16.0 portable version</td> </tr> <tr> <td>Node JS</td> <td></td> <td></td> <td>8.9.3</td> </tr> <tr> <td>STRAWBERRY PERL</td> <td></td> <td></td> <td>5.28.1.1</td> </tr> <tr> <td>Git</td> <td></td> <td></td> <td>2.20.1</td> </tr> <tr> <td>Git Desktop</td> <td></td> <td></td> <td>-</td> </tr> <tr> <td>GitHub SG</td> <td></td> <td></td> <td>-</td> </tr> </tbody> </table>

2. Front -End (SharePoint) Successfully import API by using [JavaScript Object Model (JSOM)](https://docs.microsoft.com/en-us/sharepoint/dev/sp-add-ins/sharepoint-net-server-csom-jsom-and-rest-api-index) of SharePoint 2013 , including the following features : - Create folders / files on SP - Move files on SP - Delete folders / files on SP

2.1 Functions Created

> Export user groups ==(Create Files on SP)==

- Create the user groups file directly on SharePoint

> Version Control ==(Promises, async/await)== - Roll-back or Roll-out the current version of TBQRT by clicking on a button - See the conception [here](https://spok.fr.world.socgen/team/Dashboards/TBQRT/admin/rollout_rollback.PNG)

2.2 JS files

In order to solve the problems, we created and modified some JavaScript files :

<table border="1" class="docutils"> <thead> <tr> <th>Name</th> <th>Usage</th> </tr> </thead> <tbody> <tr> <td>sharepoint_io_utils.js</td> <td>Include basic operations on SP</td> </tr> <tr> <td>sharepoint_io_utils_promise.js</td> <td>Another choice for async / await</td> </tr> <tr> <td>export_user_groups.js</td> <td>Import a function to export user groups on SP</td> </tr> <tr> <td>TBQRT_admin.js</td> <td>Add all necessary functions for main page TBQRT_admin.html</td> </tr> </tbody> </table>

Some code example of JavaScript :

1.Copy files on SP

‘‘‘javascript let sp_copyFile_promise = function(sourcefilepath, targetfilepath, resultpanel, successMessage, errorMessage) {

return new Promise(function(resolve, reject) { var clientContext; var oWebsite; var sourcefileUrl; var targetfileUrl; var sourcefile;

clientContext = new SP.ClientContext.get_current(); oWebsite = clientContext.get_web();

clientContext.load(oWebsite); // oSourcefilepath = oWebsite.getFolderByServerRelativeUrl(sourceFolder); // context.load(oSourceFolder, ‘Files’); // oSourcefile = oWebsite.getFileByServerRelativeUrl(sourceFolder);

clientContext.executeQueryAsync(function () { sourcefileUrl = oWebsite.get_serverRelativeUrl() + ‘/’ + sourcefilepath; targetfileUrl = oWebsite.get_serverRelativeUrl() + ‘/’ + targetfilepath; sourcefile = oWebsite.getFileByServerRelativeUrl(sourcefileUrl); sourcefile.copyTo(targetfileUrl, SP.MoveOperations.overwrite);

clientContext.executeQueryAsync(Function.createDelegate(this, successHandler), Function.createDelegate(this, errorHandler)

);

}, errorHandler);

function successHandler() {

if (successMessage === undefined) { successMessage = ‘ The file has been copied from ‘ + sourcefileUrl + ‘ to ‘ + targetfileUrl + ‘ ’;

} resultpanel.innerHTML += successMessage; resolve();

}

function errorHandler() {

if (errorMessage === undefined) { errorMessage = ‘ Request failed: ‘ + arguments[1].get_message() + ‘ ’;

} resultpanel.innerHTML += errorMessage; reject(new Error(errorMessage));

}

});

}

‘‘‘ 2. async / await function for version change

> Some code

‘‘‘javascript async function rollOutOrRollBack(

first_conf_path, second_conf_path, third_conf_path, first_data_path, second_data_path, third_data_path, buttonId, mainSuccessMessage, confErrorMessage, dataErrorMessage) {

try { await sp_readFile_promise(first_conf_path, _g_logPanel, ‘’, confErrorMessage); // Or be more precise : let checkFile = sp_readFile_promise(first_conf_path, _g_logPanel, ‘’, confErrorMessage); await sp_readFile_promise(first_data_path, _g_logPanel, ‘’, dataErrorMessage); // Or be more precise : let checkFile = sp_readFile_promise(first_data_path, _g_logPanel, ‘’, dataErrorMessage);

await sp_copyFile_promise(second_conf_path, third_conf_path, _g_logPanel, ‘’, undefined); await sp_copyFile_promise(first_conf_path, second_conf_path, _g_logPanel, ‘’, undefined); await sp_deleteFile_promise(first_conf_path, _g_logPanel, ‘’, undefined);

await sp_copyFile_promise(second_data_path, third_data_path, _g_logPanel, ‘’, undefined); await sp_copyFile_promise(first_data_path, second_data_path, _g_logPanel, ‘’, undefined); await sp_deleteFile_promise(first_data_path, _g_logPanel, mainSuccessMessage, undefined); // other code // await awaitCallback(‘success’, buttonId); update_button_color(buttonId, ‘success’);

} catch (err) { // other code // await awaitCallback(‘fail’, buttonId); update_button_color(buttonId, ‘danger’);

}

CHAPTER 1

}

3. Python (Back-End)

3.1 Evolution for TB_RCT_CERTIF

> Evolutions required from client

We finished some evaluations for ***RCT_CERTIFICATION*** and ***TBQRT*** Dashboard required from the client Guillaume Peignot :

- Calcul et présentation des résultat selon la nouvelle organisation du Groupe (BU & SU)
- Calcul et présentation des résultats avec ou sans les ELR (en filtrant sur le champ « code ELR ») de RCT
- Pour le flag prudentiel ne prendre en compte que les tiers avec la valeur 15 pour la CTR pour le calcul de l'indicateur

> Creation of new API for filter RCT_PM

In order to realize the function for filter, we created a new Python package:

- Location: **V:RESGBSCDASH-PREPGDOhomolprocesslibsrct**
- filters.py
- decorators.py

> Some functions in Package

<table border="1" class="docutils"> <thead> <tr> <th>Name</th> <th>Example Usage</th> </tr> </thead> <tbody> <tr> <td>filter_list</td> <td>filter on CTR=15 or STLR='open' ,return a Data-frame</td> </tr> <tr> <td>filter_ifEmpty</td> <td>filter CODE_STP is empty or not, return a Data-frame</td> </tr> <tr> <td>filter_all_result</td> <td>apply all filters above, return a Data-frame</td> </tr> <tr> <td>@timer</td> <td>count how much time cost for a function</td> </tr> </tbody> </table>

3.2 Process Notify_Recipients

> Context Reminder

Our tools run on two different platforms/environments: 1. Local environment (code in Python, data stored on shared drive V:or SharePoint): it is the main running "engine". 2. SharePoint environment: this is necessary to run

SharePoint-specific code, not available within the local environment; this environment is used only when the feature cannot be developed within the local environment. 3 technical platforms are available for SharePoint-specific code:

- REST APIs, which require a specific setup to handle HTTP requests (in particular error codes, etc.).
- .NET client object model, with C# code, which require also a specific setup.
- **JavaScript client object model (JSOM), which allows for an easy integration: it only requires to inject JavaScript code int**
  - This is the selected technical platform.
- Note: there is no Python interface for SharePoint.

> Problem

- We would like to launch from the main Python "running engine" an *asynchronous* process (SharePoint-specific code) running on SharePoint environment: retrieving current user Access Control Lists.
- **We need a way for:**
  1. the main Python "running engine" to communicate to SharePoint: launch a SharePoint process.
  2. the SharePoint process to "communicate" to the main Python "running engine" when it completes.

> Solution

1. "Communication" from local environment (Python) to SharePoint - We open the general "admin console" webpage from the local environment (Python). - We "simulate" the push of a button on launching the webpage through the use of query strings.

2. "Communication" from SharePoint to local environment (Python) Our solution is to "block" the Python process in local environment until the SharePoint *asynchronous* process completes. - before we run the SharePoint process, we move the user ACL we got last time to the __ARCHIVE__ folder, - the Python process reads the user ACL (whenever it is available), - when the Python process reads the ACL ("CONCERNED_RECIPIENTS_FILE"), we can be sure it's always the latest version (which was just produced), because the previous one was archived just before. ‘‘‘python while not os.path.exists(CONCERNED_RECIPIENTS_FILE):

    pass

concerned_current_recipients_TO_list = codecs.open(CONCERNED_RECIPIENTS_FILE, 'r', encoding='utf-8').read()

‘‘‘

3.3 Check Data consistency & Integrity

In order to check the data consistency , we implemented a model, here is a test I did :

I checked the data consistency between current month and last month for ==SUMMARY_SUPER_COMPACT_EN==

> Data-frames

- prev_df : SUMMARY_SUPER_COMPACT_EN for last month
- current_df : SUMMARY_SUPER_COMPACT_EN for this month
- delta_df : result after merge

> Keys

‘‘‘python merge_keys = [

    'REPORTING PCRU DOMAIN', 'RISK DOMAIN', 'OPEN_STATUS_EN', 'EAD SIGN', 'ELR STATUS', 'CTR CATEGORY', 'DATA_CODE__Lib_EN', 'STATUS_CODE__Cat_EN_1'

]

val_keys = [‘TPs Count’, ‘EAD’]

final_cols = merge_keys + val_keys

‘‘‘ **> How**

`python delta_df = prev_df[final_cols].merge(current_df[final_cols], how='outer', on=merge_keys, suffixes=('_prev', '_now')) `

> formula

‘‘‘python delta_df[‘Exist in Previous’] = 1 delta_df[‘Exist in Current’] = 1

delta_df.loc[delta_df[‘EAD_prev’].isnull(), ‘Exist in Previous’] = 0 new_lines = delta_df.loc[delta_df[‘Exist in Previous’] == 0]

delta_df.loc[delta_df[‘EAD_now’].isnull(),’Exist in Current’] = 0 removed_lines = delta_df.loc[delta_df[‘Exist in Current’] == 0]

‘‘‘ ***prev_df.shape[0] + new_lines.shape[0] = current_df.shape[0] + removed_lines.shape[0]***

3.4 Code Profile

The most common tool for code profiling is the default extension in Pycharm , click [here](https://www.jetbrains.com/help/pycharm/profiler.html) to get more details

CHAPTER 2

Indices and tables

- genindex
- modindex
- search