IBM Cloud 用戶實作研習營

使用Web app上傳檔案至Cloud Object Storage並保密其金鑰

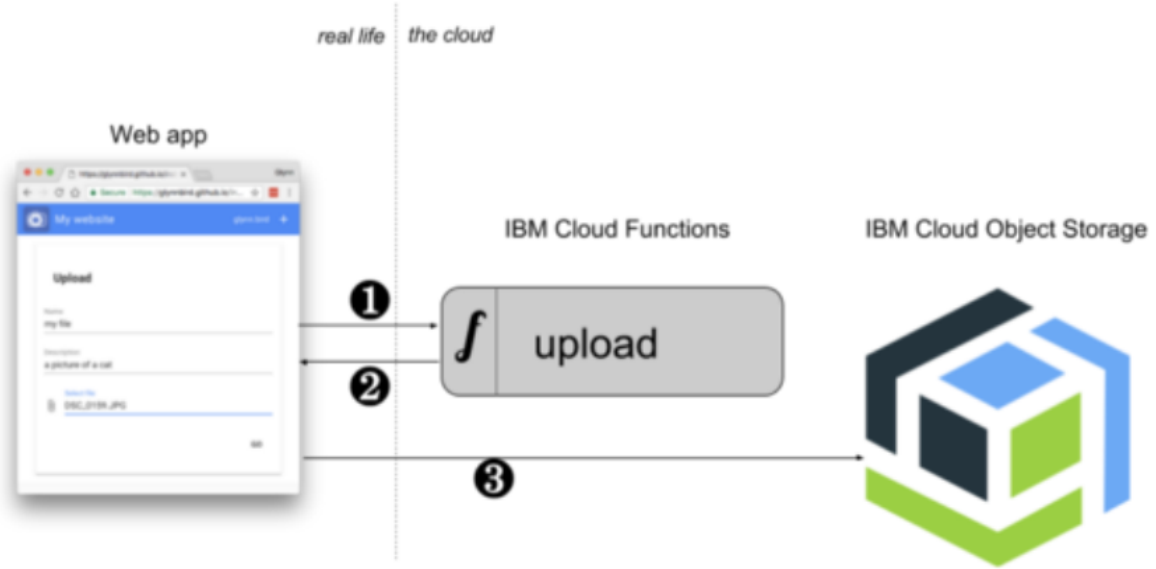
2019/8/7

如何使用Web app上傳檔案至Cloud Object Storage並保密其金鑰

IBM Cloud提供用戶應用程式上傳無大小限制的資料至IBM Cloud Object Storage，而應用程式要將檔案上傳至IBM Cloud Object Storage必須擁有相關的私密金鑰。
若我們不希望應用程式可以存取IBM Cloud Object Storage私密金鑰，又希望同時能讓應用程式上傳檔案至IBM Cloud Object Storage，我們可以使用以下方式解決：

- 透過IBM Cloud Functions預先取得使用IBM Cloud Object Storage的憑證，來避免Web app存取Cloud Object Storage的私密金鑰
- 使用pre-signed URL，讓Web app直接將檔案上傳至IBM Cloud Object Storage，不須透過IBM Cloud Functions上傳

流程：



1. Web app使用API Call向IBM Cloud Functions發出上傳檔案的請求
2. IBM Cloud Functions 產生有時效性且僅能上傳單一物件的pre-signed URL，回傳給Web app
3. Web app利用可利用IBM Cloud Functions 回傳的pre-signed URL直接將檔案上傳至IBM Cloud Object Storage

使用IBM Cloud Functions上傳檔案

建立CORS

Step1：建立cors.json檔案

執行以下指令，在終端機切換到執行目錄C:\ CloudFunction

cd ./ CloudFunction

在該目錄下建立cors.json檔)建立cors.json檔案，在該檔案內貼上以下結構 (不需修改)

```
{
   "CORSRules": [
     {
       "AllowedOrigins": ["*"],
       "AllowedHeaders": ["*"],
       "AllowedMethods": ["GET","PUT"],
       "MaxAgeSeconds": 3000
     }
   ]
 }
```

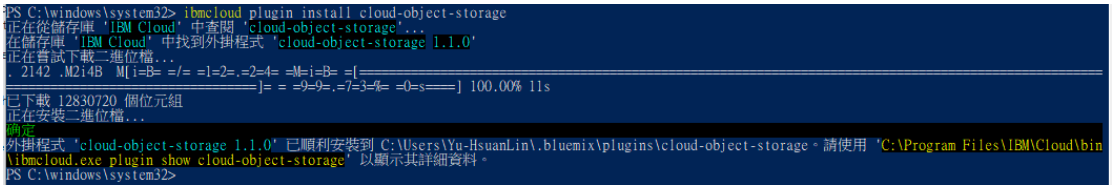
如下圖：

Step2：寫入CORS組態至服務 (這裡使用ibmcloud CLI，欲使用AWS CLI可參考Appendix)

- 執行以下指令

```
ibmcloud plugin install cloud-object-storage
```

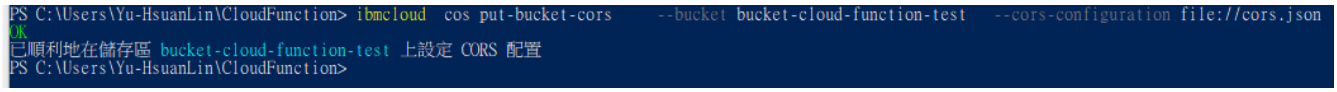
如下圖：



- 執行以下指令

```
ibmcloud  cos put-bucket-cors  --bucket <bucket名稱>  --cors-configuration file://<json檔名稱>
```

如下圖：



建立產生pre-signed URL的Node.js檔案

Step1：建立Node.js檔案 ，將檔名設為 upload.js

建立Node.js檔案，並貼上以下內容

```
// built-in Node.js crypto library
const crypto = require('crypto');

// generates a random, 20 character UUID
const uuid = function() {
  const chars =
'abcdefghijklmnopqrstuvwxyz0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ'
  var bits = []
  for(var i = 0; i < 20; i++) {
    bits.push(chars[Math.floor(Math.random()*chars.length)])
  }
  return bits.join('')
}
// serverless function that generates a new Object Storage pre-
signed URL that allows
// a file to be uploaded to Object Storage without knowing the
secret key.
const main = function(msg) {

  // pre-sign the request
  var extension = msg.content_type.split('/')[1]
  var expires = '' + Math.floor(((new Date()).getTime()/1000) +
60)
  var path = '/' + msg.bucket + '/' + uuid() + '.' + extension
  var hmac = crypto.createHmac('sha1', msg.secret_key);
  hmac.update('PUT\n' + '\n' + msg.content_type +  '\n' +
expires + '\n' + path)
  var signature = hmac.digest('base64')

  // generate URL
  var params = 'AWSAccessKeyId=' +
encodeURIComponent(msg.access_key) + '&Signature=' +
encodeURIComponent(signature) + '&Expires=' +
encodeURIComponent(expires)
  var request_url = msg.endpoint + path + '?' + params
  return { url: request_url }
}
exports.main = main
```

部屬IBM Cloud Functions

Step1：安裝 Cloud Functions 外掛程式

在終端機輸入以下指令：

`ibmcloud plugin install cloud-functions`

Step2：設定目標名稱空間

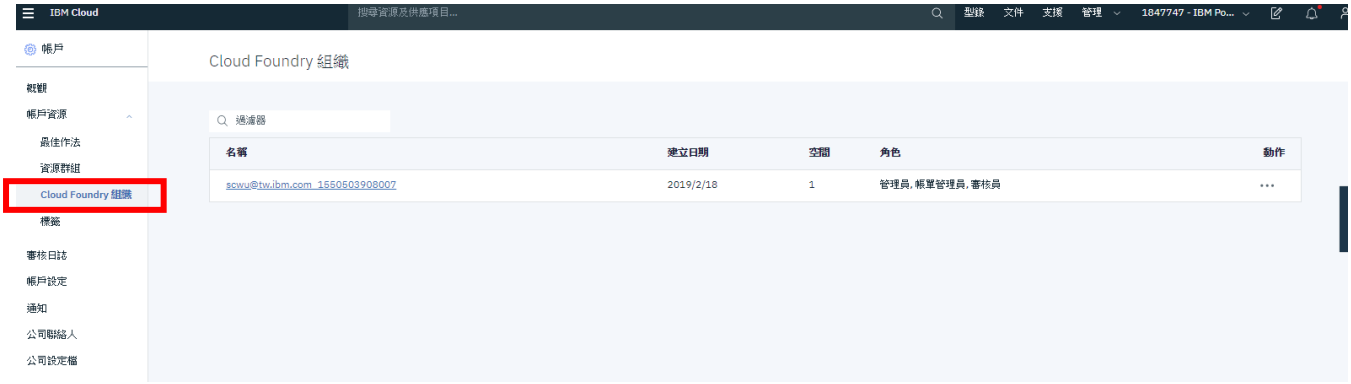
在終端機輸入以下指令，<org>及<space>需替換為自己的：

`ibmcloud target -o <org> -s <space>`

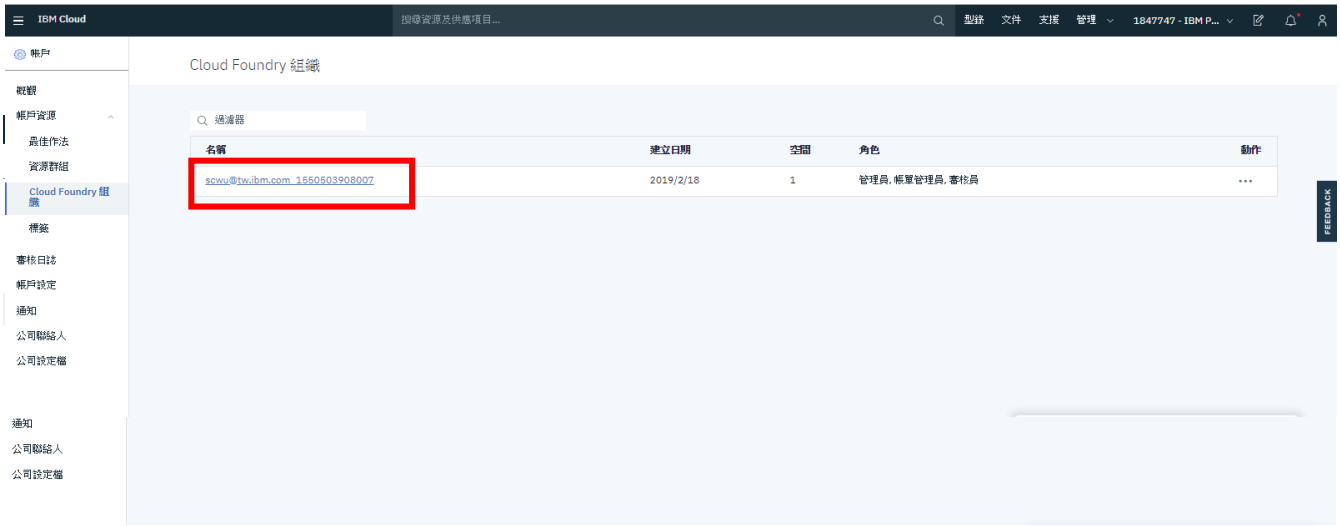
- 如何查詢<org>即 <space>的值

進入到portal首頁，從右上角 管理 中選擇 帳戶

點選 Cloud Foundry 組織



以此處為例，scwu@tw.ibm.com_1550503908007 即為要填入的<org>值



以此處為例，執行代碼如下：

```
ibmcloud target -o scwu@tw.ibm.com_1550503908007 -s dev
```

Step3：將upload.js部署至IBM Cloud Functions

執行以下代碼：

```
bx wsk action update <動作名稱> upload.js  --web true --param access_key '<access key值>'
--param secret_key '<secret key值>' --param bucket '<bucket名稱>'
--param endpoint '<public 端點位置>'
```

- 其中<動作名稱>可以設為如：upload- wk2201908000
  (wk2201908000須改為自己的序號，如wk2201908001、wk2201908002)

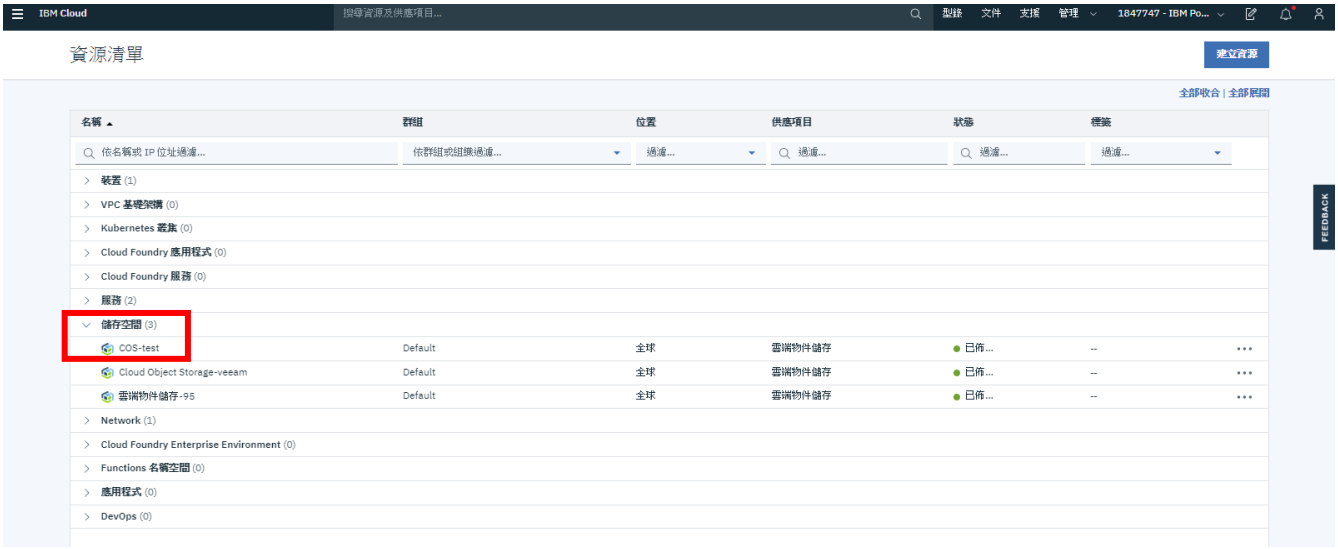
  所建立的<動作名稱>可在portal首頁測選單/Function

  點選動作來查看

- 如何查看access key值、secret_key值

從portal首頁側邊選單選擇 資源清單

在儲存空間選擇之前所建立的Cloud Object Storage



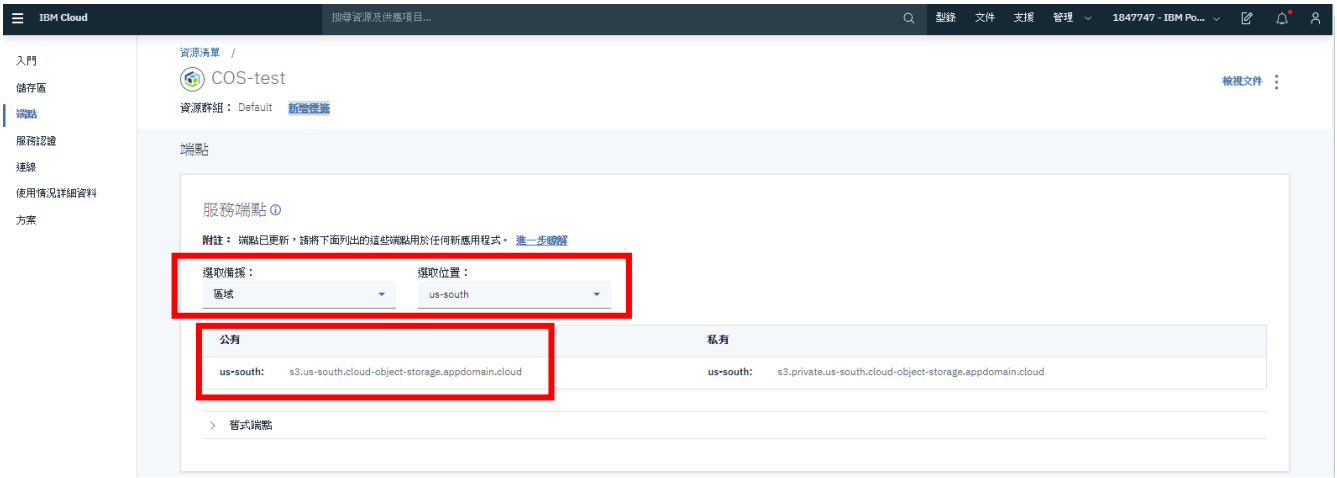
在側邊選單選擇 服務認證

點選 檢視認證

其中 access_key_id 的值即為 access key的值、secret_access_key的值即為secret_key的值

- 如何查看 public 端點位置

同樣在剛剛的畫面(Cloud Object Storage管理介面)，側邊選單選擇端點，
點選後，在選擇備援部分選擇區域，選取位置選擇us-south，
複製 公有 (public)的位置 為public端點位置 ，
以此處為例即為 s3.us-south.cloud-object-storage.appdomain.cloud



Step4：產生action URL

在終端機執行以下指令，其中<動作名稱>為剛剛將upload.js部署至IBM Cloud Functions時所使用的<動作名稱>

```
bx wsk action get <動作名稱> --url
```

複製所產生的URL，等一下會用到，

以下圖為例，複製https://us-south.functions.cloud.ibm.com/api/v1/web/scwu%40tw.ibm.com_1550503908007_dev/default/upload-wk2201908000

```
PS C:\Users\Yu-HsuanLin\CloudFunction> bx wsk action get upload-wk2201908000 --url
ok: got action upload-wk2201908000
https://us-south.functions.cloud.ibm.com/api/v1/web/scwu%40tw.ibm.com_1550503908007_dev/default/upload-wk2201908000
PS C:\Users\Yu-HsuanLin\CloudFunction>
```

從網頁前端開啟

Step1：建立html檔

- 建立html檔，檔名設為 `upload.html` ，並在檔案內貼上以下程式碼

```
<!DOCTYPE
html>
          <html lang="en">
          <head>
            <style type="text/css">
              body {
                font-family: "Helvetica Neue",Helvetica,Arial,sans-serif
              }
              #drop_zone {
                width:50%;
                height:200px;
                padding:20px;
                border: 2px dashed red
              }
              .ondrag {
                border: 2px solid red !important
              }
              #status {
                width:50%;
                background-color:black;
                color: #00ff00;
                min-height:200px;
                font-size:22px;
                padding:20px;
                border: 2px solid black
              }
            </style>
          </head>
          <body>
              <div id="drop_zone" ondrop="drop_handler(event);" ondragover="onDragOver(event)"
          ondragleave="onDragLeave(event)">
                <strong>Drop Zone ...</strong>
              </div>
              <div id="status"></div>
              <script src="https://code.jquery.com/jquery-3.3.1.min.js"></script>
              <script>
```

```
        // given a content-type, get a presigned URL to allow a file of that type to be
uploaded
        var getPresignedURL = function(content_type, callback) {
          $.ajax({
            type: 'POST',
            url: 'YOUR_SERVERLESS_URL',
            data: JSON.stringify({content_type: content_type}),
            contentType: 'application/json; charset=utf-8',
            dataType: 'json',
            error: function(e) {
              console.log('ajax error', e);
              callback(true, null);
            },
            success: function(d) {
              console.log('ajax success', d)
              callback(null, d)
            }
          });
        }
        // make an HTTP PUT, writing the file's contents to object storage
        var upload = function(f, content_type, url, callback) {
          var reader = new FileReader();
          reader.onload = function(event) {
            $.ajax({
              type: 'PUT',
              url: url,
              data: reader.result,
              contentType: content_type,
              processData: false,
              error: function(e) {
                callback(e, null);
              },
              success: function(d) {
                callback(null, d)
              }
            });
          }
          reader.readAsArrayBuffer(f)
        }
        var onDragOver = function(ev) {
          ev.preventDefault();
          $('#drop_zone').addClass('ondrag')
        }
        var onDragLeave = function(ev) {
          ev.preventDefault();
```

```
        $('#drop_zone').removeClass('ondrag')
      }
      // called when a file is dropped in the drop zone
      var drop_handler = function (ev) {
        ev.preventDefault();
        $('#drop_zone').removeClass('ondrag')
        var f = ev.dataTransfer.files[0];
        $('#status').html('')
        $('#status').append('Preparing to upload file<br>')
        $('#status').append('* filename: ' + f.name + '<br>')
        $('#status').append('* content-type: ' + f.type + '<br>')
        $('#status').append('* content-length: ' + f.size + '<br>')
        $('#status').append('Getting pre-signed upload URL from IBM Cloud Functions<br>')
        getPresignedURL(f.type, function(err, presigned) {
          if (err) {
            return console.log('error - could not get URL')
          }
          var url = presigned.url
          $('#status').append('Uploading to IBM Cloud Object Storage<br>')
          upload(f, f.type, presigned.url, function(err, data) {
            if (err) {
              $('#status').append('Upload error<br>')
            } else {
              $('#status').append('Upload Complete<br>')
            }
          })
        });
      }
    </script>

  </body>
  </html>
```

- 將程式碼內的 YOUR_SERVERLESS_URL 改為剛剛複製的那串URL，

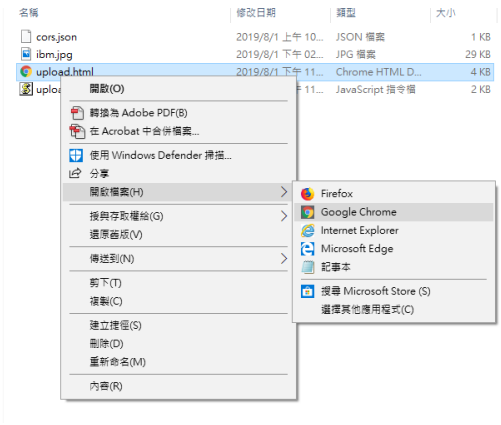
並在URL後面加上.json?content_type=image/jpg

以此處為例，即將YOUR_SERVERLESS_URL 改為

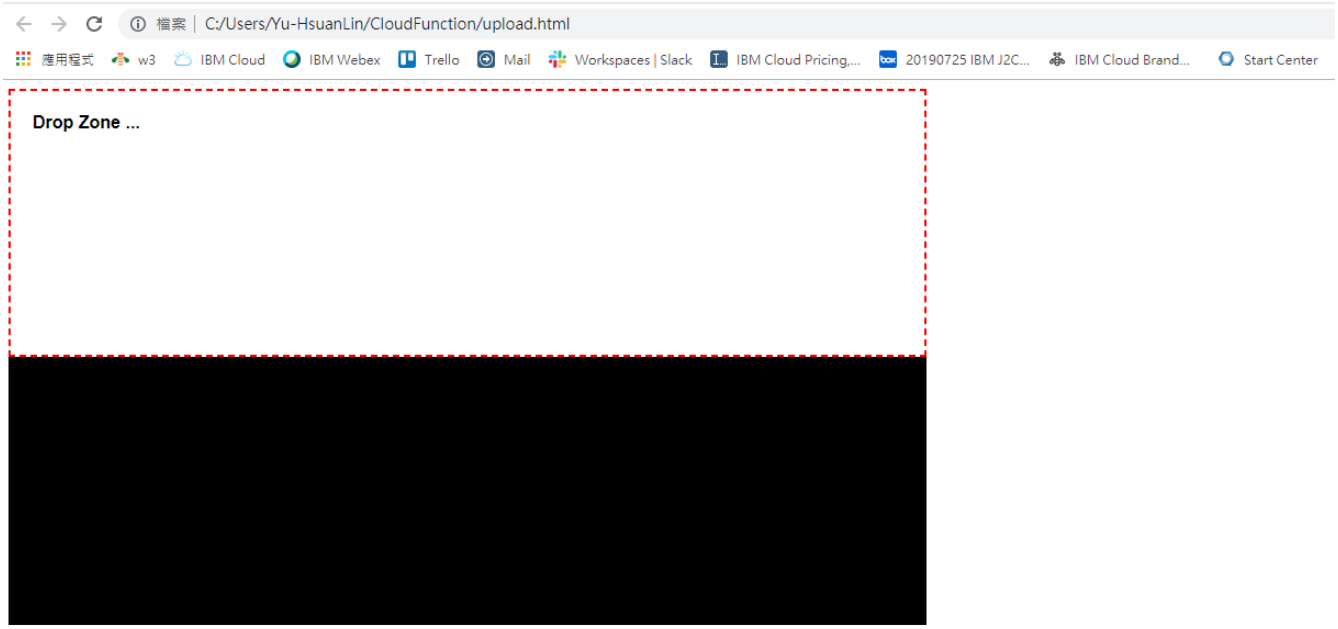
https://us-south.functions.cloud.ibm.com/api/v1/web/scwu%40tw.ibm.com_1550503908007_dev/default/upload-wk2201908000.json?content_type=image/jpg

進行測試

Step1：在瀏覽器開啟upload.html檔

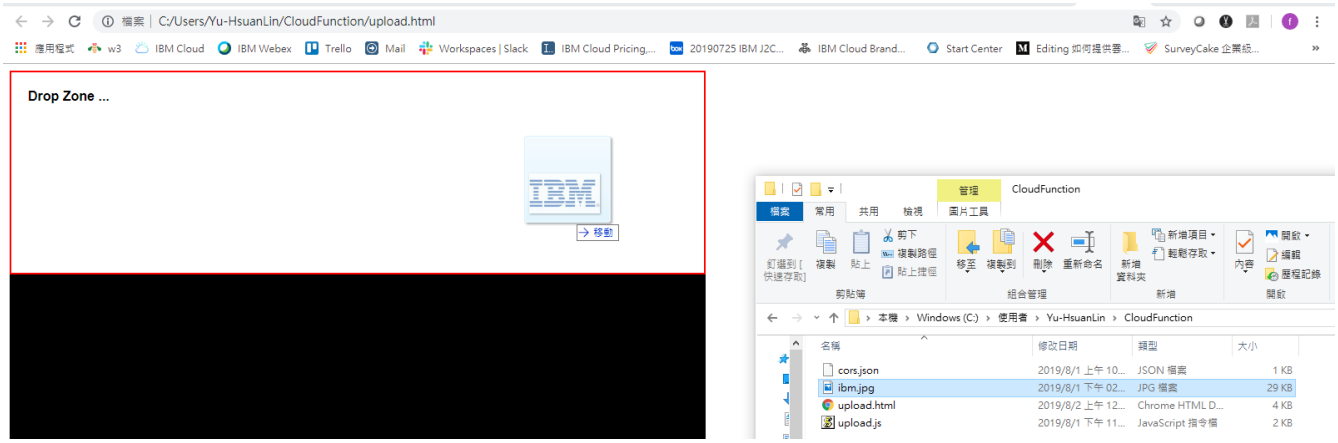


網頁開啟後會像下圖



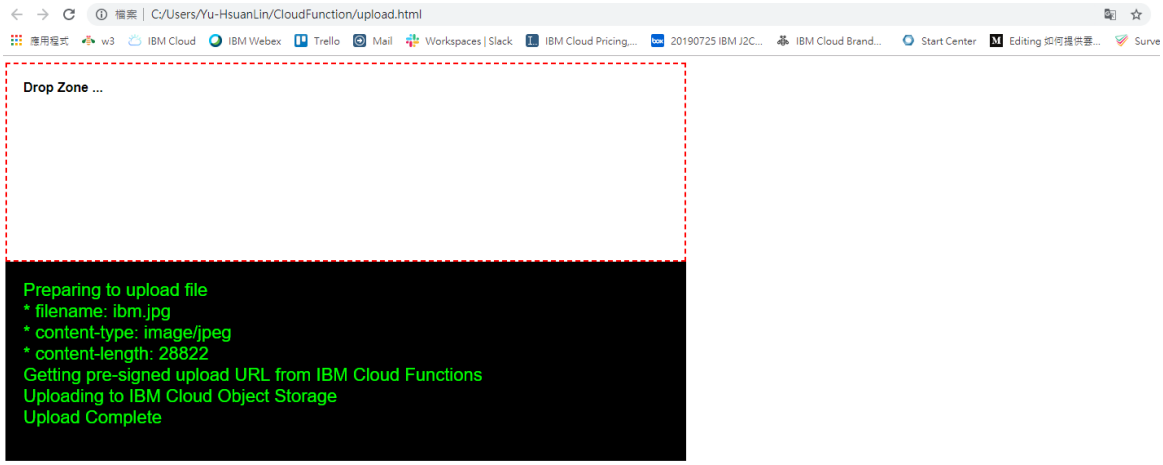
Step2：上傳檔案

將jpg圖片檔拉至網頁紅色框框內



可以看到當拉jpg圖檔置網頁後，

1. 網頁會傳送要求pre-signed的URL的請求給IBM Cloud Functions，
2. IBM Cloud Functions 會回傳pre-signed的URL，
3. 網頁接著利用這個pre-signed的URL直接將檔案上傳是IBM Cloud Object Storage



進入到ibm.cloud.com我們所建立的bucket內即可看到剛剛所上傳的jpg檔了

Appendix

使用AWS CLI 寫入CORS組態至服務

Step1：

Windows作業系統版本：至https://aws.amazon.com/tw/cli/下載安裝檔

Mac / Linux作業系統版本 /已安裝pip：使用指令 `pip install awscli` 進行安裝

Step2：

在終端機輸入以下指令

`aws configure`

指令輸入後，會要求須依序填入

AWS Access Key ID [None]: {Access Key ID}

AWS Secret Access Key [None]: {Secret Access Key}

Default region name [None]: {Provisioning Code}

Default output format [None]: json

- 其中Access Key ID、Secret Access Key可至 `IBM Cloud Object Storage/服務憑證/查看憑證` 中查詢
- Provisioning Code可依據bucket所建置的位置在下圖中找尋對應值
  (可參考https://reurl.cc/zvdV6)

US Geo: `us-standard` / `us-vault` / `us-cold` / `us-flex`
US East: `us-east-standard` / `us-east-vault` / `us-east-cold` / `us-east-flex`
US South: `us-south-standard` / `us-south-vault` / `us-south-cold` / `us-south-flex`
EU Geo: `eu-standard` / `eu-vault` / `eu-cold` / `eu-flex`
EU Great Britain: `eu-gb-standard` / `eu-gb-vault` / `eu-gb-cold` / `eu-gb-flex`
EU Germany: `eu-de-standard` / `eu-de-vault` / `eu-de-cold` / `eu-de-flex`
AP Geo: `ap-standard` / `ap-vault` / `ap-cold` / `ap-flex`
AP Japan: `jp-tok-standard` / `jp-tok-vault` / `jp-tok-cold` / `jp-tok-flex`
AP Australia: `au-syd-standard` / `au-syd-vault` / `au-syd-cold` / `au-syd-flex`
Amsterdam: `ams03-standard` / `ams03-vault` / `ams03-cold` / `ams03-flex`
Chennai: `che01-standard` / `che01-vault` / `che01-cold` / `che01-flex`
Hong Kong: `hkg02-standard` / `hkg02-vault` / `hkg02-cold` / `hkg02-flex`
Melbourne: `mel01-standard` / `mel01-vault` / `mel01-cold` / `mel01-flex`
Mexico: `mex01-standard` / `mex01-vault` / `mex01-cold` / `mex01-flex`
Milan: `mil01-standard` / `mil01-vault` / `mil01-cold` / `mil01-flex`
Montréal: `mon01-standard` / `mon01-vault` / `mon01-cold` / `mon01-flex`
Oslo: `osl01-standard` / `osl01-vault` / `osl01-cold` / `osl01-flex`
San Jose: `sjc04-standard` / `sjc04-vault` / `sjc04-cold` / `sjc04-flex`
São Paulo: `sao01-standard` / `sao01-vault` / `sao01-cold` / `sao01-flex`
Seoul: `seo01-standard` / `seo01-vault` / `seo01-cold` / `seo01-flex`
Toronto: `tor01-standard` / `tor01-vault` / `tor01-cold` / `tor01-flex`

Step3：

在終端機輸入以下指令

```
aws --endpoint-url=端點位置 s3api  put-bucket-cors    --bucket bucket名稱  --cors-configuration file://json檔案名稱
```