歡迎使用創意雲 API

創意雲 API 提供開發者一個方便、快速及便利的雲端空間與軟體連結。讓開發者輕鬆達成跨平台資料存取。同時創意雲 API 也提供文件分享、ibon 列印、關鍵字查詢等…功能讓開發者可以更加靈活運用。

環境說明

由於創意雲 API 主要是使用 XML 與 URL 做資料交換的方式。所以可以適用於任何平台。而以下平台使用 ANDROID 做為開發範例說明。使用環境如下:

1. Android : 平台版本: Android2.3
2. Java : Java 1.6
3. 開發工具 : Eclipse

由於 Android 不同的版本有不同相對應的支援性。因此在開發時，以下文件在不同版本的 Android 上也許需要做部分修改方可使用。

4. 範例程式下載：

   請利用官網下載範例程式。

   https://creative.asuscloud.com/?p=notes

   或直接使用此連結下載 Android 範例程式。

   https://creative.asuscloud.com/content/api/samplecode/AWS_Demo_PHP.zip

開發金鑰

在開發之前。請先至 ASUS Web storage 官方網站取的開發金鑰、以作為開發需求方式。網址如下:

https://creative.asuscloud.com/?p=started

申請完畢後。每位開發者將會取得一組 SID 及 ProgKey 以作為開發者呼叫 API 時的驗證。如果無此驗證碼，開發者將無法呼叫 API。

系統架構說明

由於 ASUS WebStorage Serivce 會依不同的地區的申請者分配最近區域的 SERVER。讓每位使用者取得更加快速的存取速度。所以每位使用者所分配到的 SERVER 並不相同。ASUS WebStorage Serivce 架構主要由 Service Portal ,Service Gateway ,InfoRelay ,WebRelay, Search Server 等組成。

Service Portal：管理使用者服務區域、並取得 Service Gateway Domain 使用。

Service Gateway： 管理使用者服務區域、取得使用者使用狀態及 InfoRelay ,WebRelay, Search Server 等 Domain。

InfoRelay：檔案及資料相關查詢、變更、搬移、刪除等。

WebRelay：對實體檔案做存取、轉檔。

Search Server：做關鍵查詢。

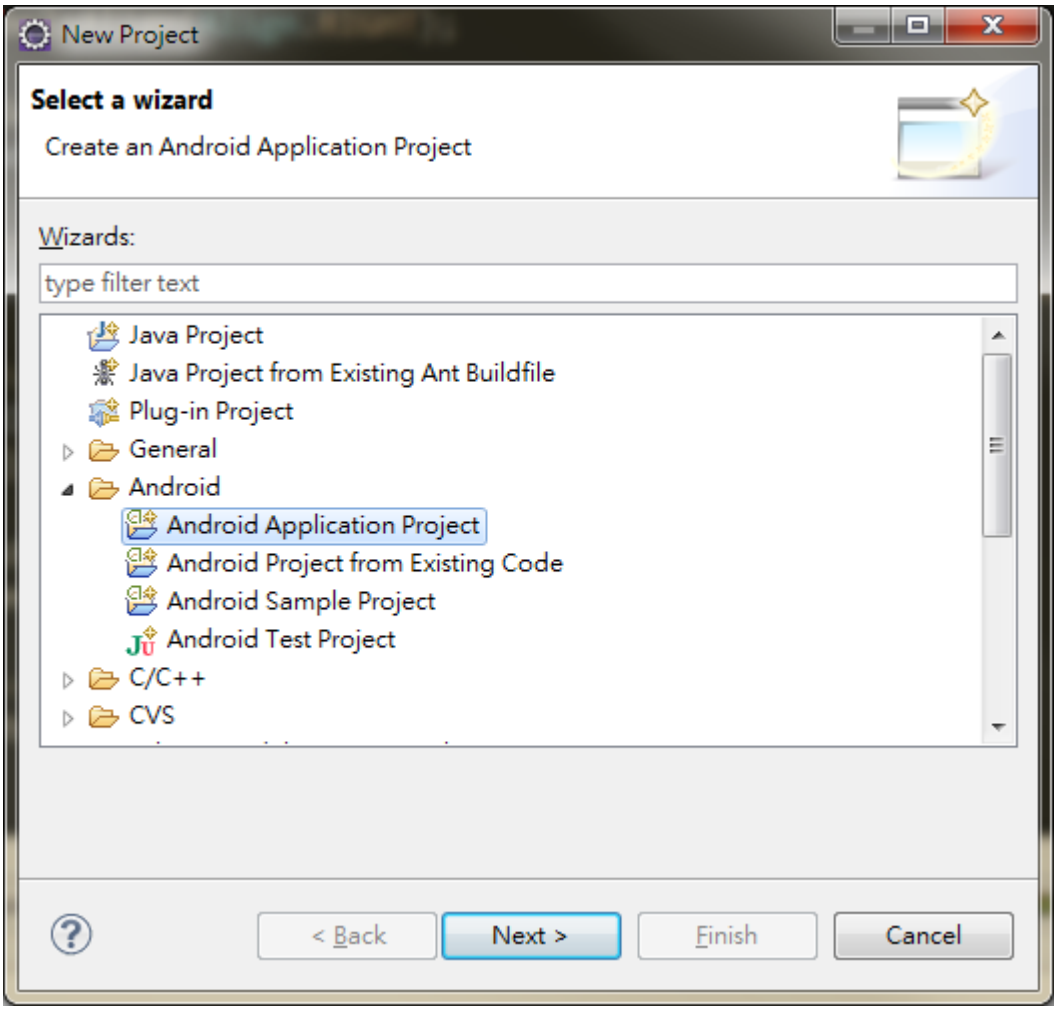
因此每位開發者需要取得使用者相對應的 Server Domain 方可以使用相對應的 API。

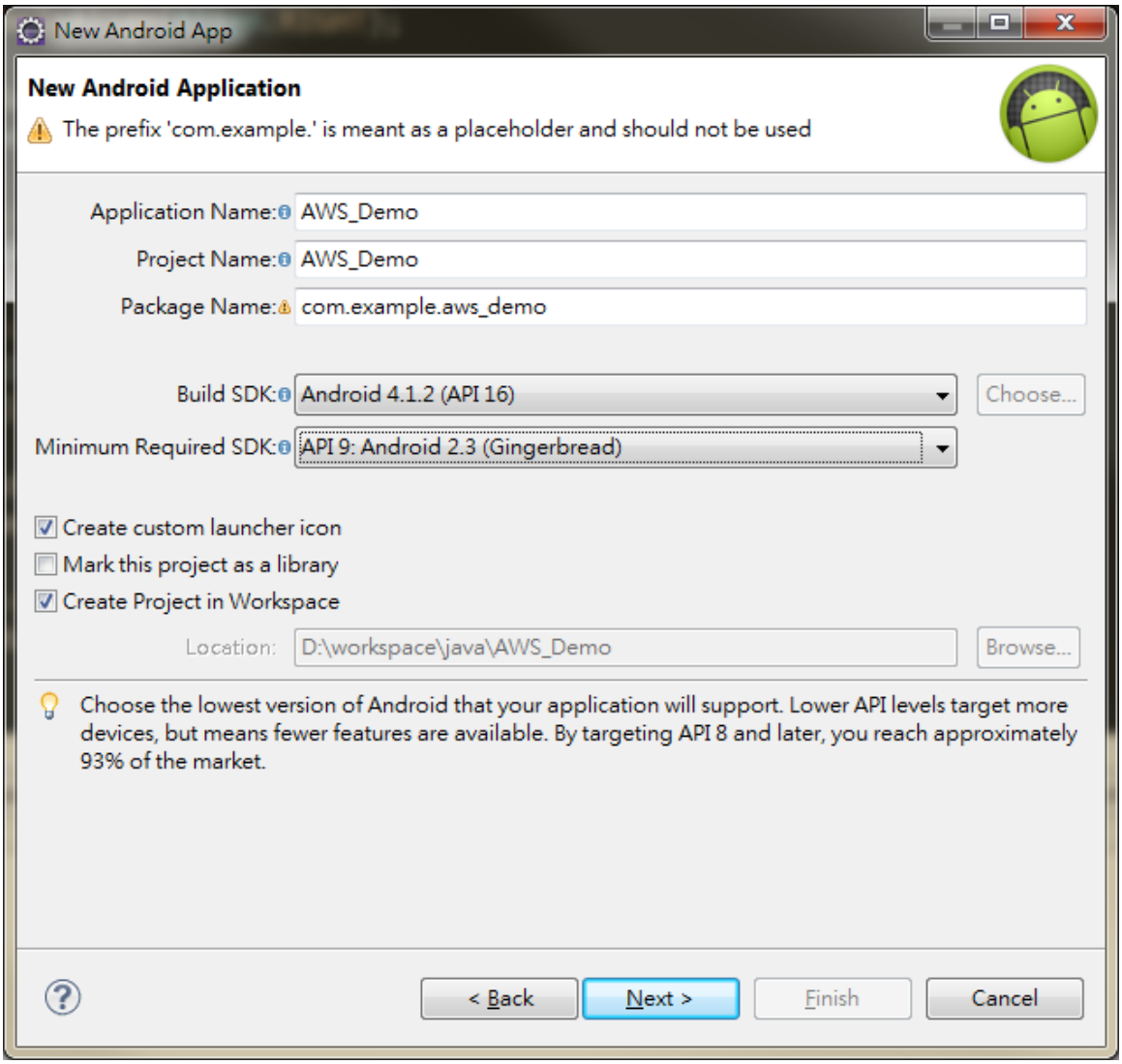
範例程式:雲端檔案

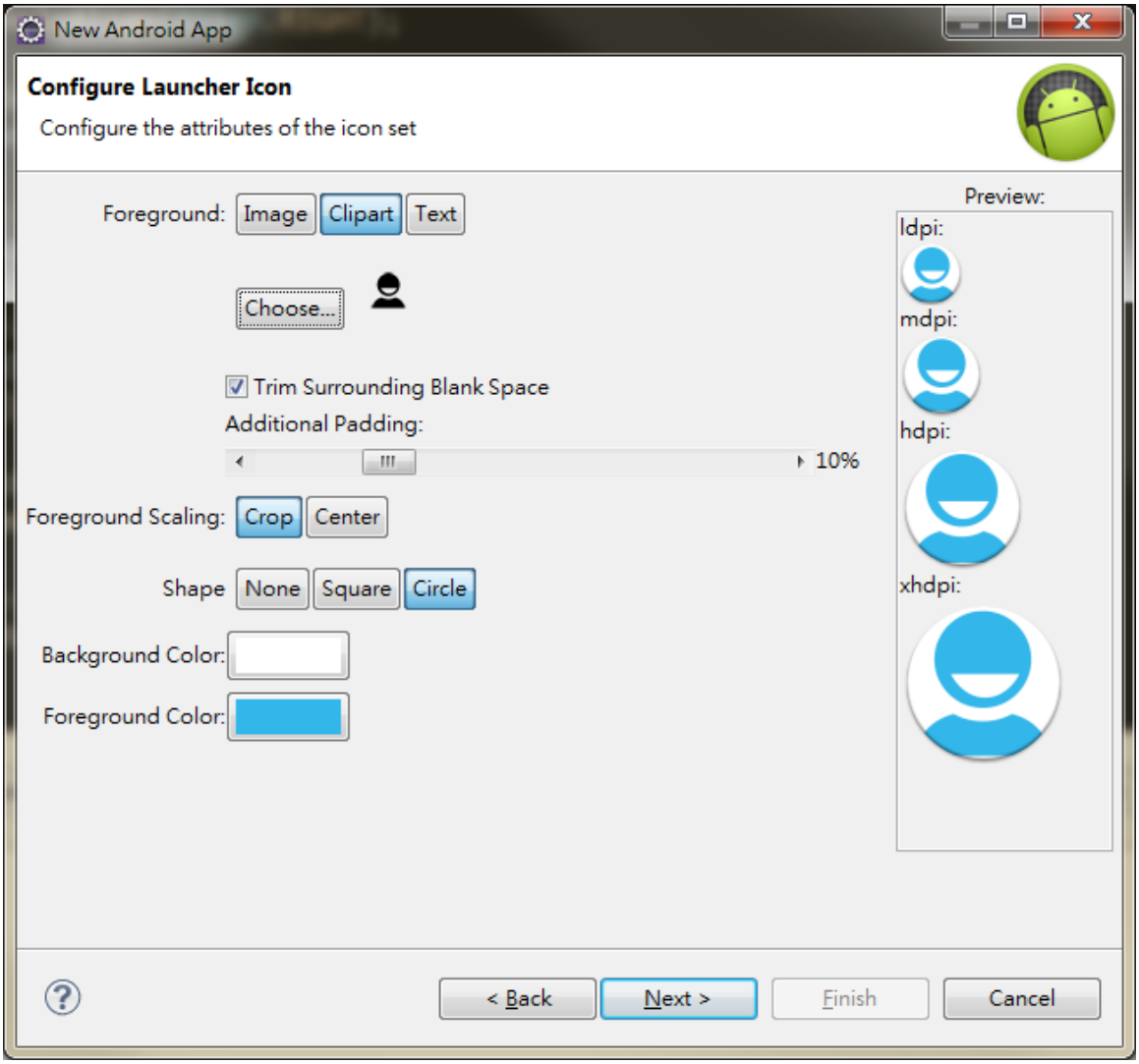
以下將會一步步，使用雲端檔案開啟 app 作為範例，以作為文件閱讀及開發範例。

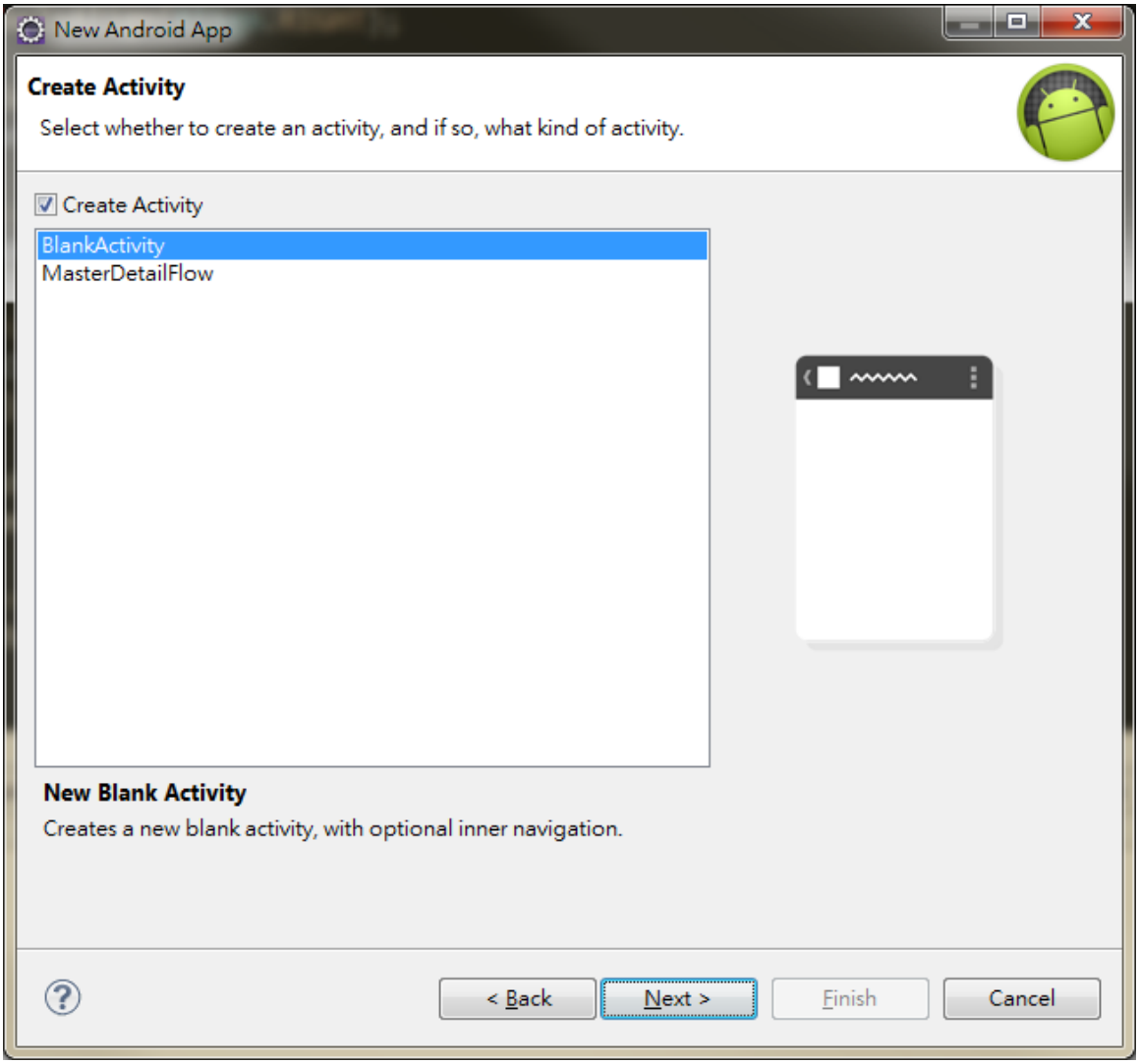
1・使用者登入

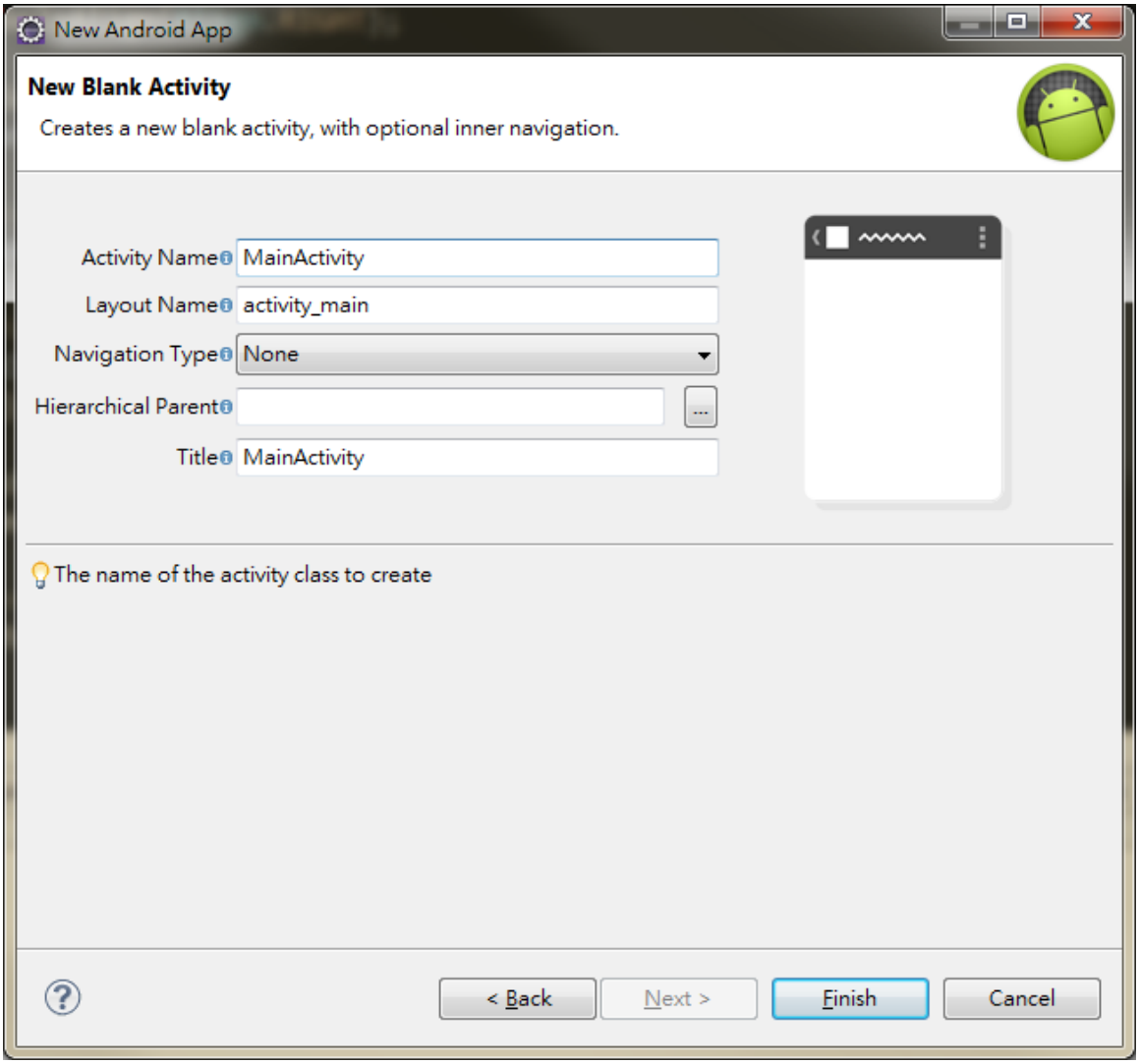
首先開啟 Eclipse 並建立一個 project 以做為程式開發使用。(如果不清楚如何安裝 Android SDK 或是 Eclipse 開發環境的可參閱 http://blog.chinatimes.com/tomsun/archive/2007/11/03/213763.html)




New Android App
New Android Application
The prefix 'com.example.' is meant as a placeholder and should not be used
Application Name:
AWS_Demo
Project Name:
AWS_Demo
Package Name:
com.example.aws_demo
Build SDK:
Android 4.1.2 (API 16)
Choose...
Minimum Required SDK:
API 9: Android 2.3 (Gingerbread)
Create custom launcher icon
Mark this project as a library
Create Project in Workspace
Location:
D:\workspace\java\AWS_Demo
Browse...
Choose the lowest version of Android that your application will support. Lower API levels target more devices, but means fewer features are available. By targeting API 8 and later, you reach approximately 93% of the market.
< Back
Next >
Finish
Cancel


New Android App
Configure Launcher Icon
Configure the attributes of the icon set
Foreground:
Image
Clipart
Text
Choose...
Trim Surrounding Blank Space
Additional Padding:
10%
Foreground Scaling:
Crop
Center
Shape
None
Square
Circle
Background Color:
Foreground Color:
Preview:
ldpi:
mdpi:
hdpi:
xhdpi:
< Back
Next >
Finish
Cancel


New Android App
Create Activity
Select whether to create an activity, and if so, what kind of activity.
Create Activity
BlankActivity
MasterDetailFlow
New Blank Activity
Creates a new blank activity, with optional inner navigation.
< Back
Next >
Finish
Cancel



當建立好 project 之後。

1.請先開啟 AndroidManifest.xml，檔案並加入下列設定。以開啟 APP 存取網路與 SD Card 之權限。

檔案範例如下:

AndroidManifest.xml

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="net.yostore.aws.view.common"
    android:versionCode="1"
    android:versionName="1.0" >
<!--網路與SD Card權限-->
    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
    <uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
    <uses-sdk
        android:minSdkVersion="9"
        android:targetSdkVersion="15" />
```

```
    <application
        android:icon="@drawable/icon"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity android:name=".LoginActivity" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
</manifest>
```

2.建立登入畫面 login.xml

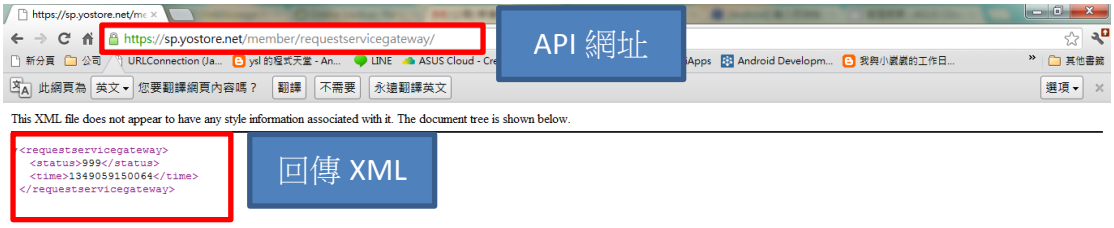
```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical" >
    <EditText
        android:id="@+id/edtUid"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:hint="請輸入使用者帳號"
        android:singleLine="true" >
    </EditText>
    <EditText
        android:id="@+id/edtPwd"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:hint="請輸密碼"
        android:inputType="textPassword"
        android:singleLine="true" />
    <Button
        android:id="@+id/btnLogin"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:gravity="bottom|center_horizontal"
        android:onClick="loginFunction"
        android:text="登入" />
</LinearLayout>
```

5. 建立 java 程式碼

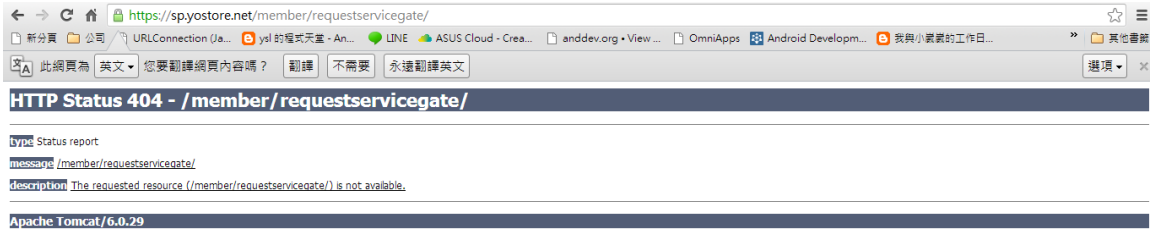
首先須藉由 Service Portal 中**/member/requestservicegateway/**取得使用者相對應的 Service Gateway。由於 API 呼叫方式為均用 API 網址組成方式為”https:// API Server Domain” + “API 路徑” 。Service Portal Domin 為 sp.yostore.net(全部使用者通用不做改變)。

因此呼叫網址為:https://sp.yostore.net/member/requestservicegateway/

如須測試 API 組成是否正確可以直接利用瀏覽器測試



若不存在則會出現下面畫面



出現畫面則為網址錯誤

而 API 相對應的 XML 格式如下:

傳入相對應的使用資料:

```
<?xml version="1.0" encoding="utf-8"?>
<requestservicegateway>
    <userid>{ User ID }</userid>
    <password>{  使用者密碼轉成小寫再做 MD5 編碼的結果  }</password>
    <language>{  使用者的語系，例如：zh_TW }</language>
    <service>[ 1 ]</service>
</requestservicegateway>
```

回傳

```
<?xml version="1.0" encoding="utf-8"?>
```

<requestservicegateway>
<status>{ Status Code }</status>
<servicegateway>{ ServiceGateway 的 host:port。例如：192.168.1.20:8080 }
</servicegateway>
</requestservicegateway>

其中當<status>{ Status Code }</status>只有為 0 時，代表成功回傳資料會有 servicegateway 的資料，非為 0 時代表有錯誤。相對應的錯誤代碼請參閱 API 文件。

而在通用 Header 的部分，需在中 **Cookie 必須指定 sid 欄位值**供 ServicePortal 記錄客戶端的資訊，Cookie 中的「sid」須為小寫英文字母。

1. 先取得使用者輸入的帳號密碼。
   並且判斷是否為空值
   LoginActivity.java

```
public class LoginActivity extends Activity {
    private final String TAG = "Login";
    private EditText txtUid = null;
    private EditText txtPwd = null;

    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.login);
        txtUid = (EditText) findViewById(R.id.edtUid);
        txtPwd = (EditText) findViewById(R.id.edtPwd);
    }

    public void loginFunction(View v) {
        StringBuilder msg = new StringBuilder();

        String uid = txtUid.getText().toString();
        String pwd = txtPwd.getText().toString();

        if (uid.trim().length() == 0 || pwd.trim().length() == 0) {
            msg.delete(0, msg.length());
            msg.append("Login info could not be empty!");
            Log.e(TAG, msg.toString());
```

```
            Toast.makeText(actLogin, msg.toString(), Toast.LENGTH_LONG).show();
            return;
        }
    }
}
```

建立登入後畫面，所以先做一個介面已確定登入完成:

s_browse.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical" >
    <TextView
        android:id="@+id/textView1"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="Login OK" />
</LinearLayout>
```

MyBrowseActivity.java

```
public class MyBrowseActivity extends ListActivity {
    private static final String TAG = "MyBrowseActivity";
    private ApiConfig apiCfg;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.s_browse);
    }
}
```

由於部分資訊會反覆使用，因此，建立一個 ApiConfig.java 作為重復資料傳遞的方式.

```
public class ApiConfig
{
    /**
     * Service Portal Domain = sp.yostore.net
     */
    public static String SERVICEPORTAL = "sp.yostore.net";
```

```
/**
 * ServiceGateway Domain is fetched by UserId & HashedPassword from ServicePortal
 */
public String ServiceGateway = "";

public final String SYNCROOTID   = "-5";
public final String SYNCFOLDERNAME = "MySyncFolder";

/***
 * MySyncFolder Real FolderId
 */
public String mySyncFolderId  = "";
public String parentFolderId  = SYNCROOTID;
public String parentName    = "";
public String currentFolderId = "";
public String folderName    = SYNCFOLDERNAME;
public String userid = "";
/**
 * Hashed Password (lower case) by MD5
 */
public String password = "";

/**
 * InfoRelay Domain (IP)
 */
public String infoRelay = "";
/**
 * Web Relay Domain (IP)
 */
public String webRelay = "";

public String token = "";

/**
 * Account Package Name
 */
public String packageDisplay = "";
public String capacity = "0";
public String usedquota = "";
public String expireDate = "";
```

```
public static final String SEPARATOR = "\n";

public static ApiConfig getFromString(String ss){
    ApiConfig apicfg = null;

    if (ss!=null){
        String str = null;
        BufferedReader reader = new BufferedReader(new StringReader(ss));
        apicfg = new ApiConfig();

        try {

            if ((str = reader.readLine()) != null) apicfg.userid         = str;
            if ((str = reader.readLine()) != null) apicfg.password       = str;
            if ((str = reader.readLine()) != null) apicfg.token      = str;
            if ((str = reader.readLine()) != null) apicfg.ServiceGateway= str;
            if ((str = reader.readLine()) != null) apicfg.infoRelay  = str;
            if ((str = reader.readLine()) != null) apicfg.webRelay       = str;
            if ((str = reader.readLine()) != null) apicfg.packageDisplay= str;
            if ((str = reader.readLine()) != null) apicfg.capacity       = str;
            if ((str = reader.readLine()) != null) apicfg.expireDate = str;
            if ((str = reader.readLine()) != null) apicfg.mySyncFolderId= str;

        } catch(IOException e) {
            e.printStackTrace();
        }
    }
    return apicfg;
}

@Override
public String toString() {
    StringBuilder sb = new StringBuilder();
    sb.append(userid);
    sb.append(SEPARATOR);
    sb.append(password);
    sb.append(SEPARATOR);
    sb.append(token);
    sb.append(SEPARATOR);
```

```
            sb.append(ServiceGateway);
            sb.append(SEPARATOR);
            sb.append(infoRelay);
            sb.append(SEPARATOR);
            sb.append(webRelay);
            sb.append(SEPARATOR);
            sb.append(packageDisplay);
            sb.append(capacity);
            sb.append(expireDate);
            sb.append(SEPARATOR);
            sb.append(mySyncFolderId);
            sb.append(SEPARATOR);
            return sb.toString();
        }
}
```

利用 Asynctask 方式讀取 XML

建立 LoginTask.java

```
public class LoginTask extends AsyncTask<Void, Integer, Integer> {
        String TAG = "LoginTask";
        ProgressDialog _mdialog;
        Context ctx;
        ApiConfig apiCfg;
        String secure;
        Intent startIntent;

        public LoginTask(Context cxt, ApiConfig apiCfg, Intent startIntent,
                        String... secure) {
                this.ctx = cxt;
                this.apiCfg = apiCfg;
                this.startIntent = startIntent;
        }

        @Override
        protected Integer doInBackground(Void... params) {
                this.publishProgress(0);
                return 0;
        }


        @Override
```

```
    protected void onPostExecute(Integer result) {
        this.publishProgress(100);
    }

    @Override
    protected void onProgressUpdate(Integer... values) {
        super.onProgressUpdate(values);

    }
}
```

由於 Asus Web Storage API 使用 Https 連線與 XML 資料傳遞。因此還需要做 Http 連線、產生上傳 XML 與解析回傳 XML。

建立 BaseApi.java

```
public abstract class BaseApi {
    private static final String SIGNATURE_METHOD = "HMAC-SHA1";

    private static final String TAG = "BaseApi";
    public static final int TIMEOUT = 60 * 1000; // 1min
    private String apiSvr;

    public static String clientversion = "";

    protected BaseApi(String ServerDomain) {
        this.apiSvr = ServerDomain;
    }

    protected ApiResponse getResponse(String api, String params,
            BaseSaxHandler handler) throws MalformedURLException,
            ProtocolException, IOException, SAXException {
        String urlStr = "https://" + this.apiSvr + api;
        URL url = new URL(urlStr);
        Log.d(TAG, urlStr);
        Log.d(TAG, "SID:[" + ApiCookies.sid + "], ProgKey:["
                + ApiCookies.progKey + "]");

        HttpsURLConnection conn = (HttpsURLConnection) url.openConnection();
        conn.setHostnameVerifier(HttpsUtil.getverifyAlHostnameVerifier());
        conn.setConnectTimeout(TIMEOUT); // 60 sec
```

```
conn.setReadTimeout(TIMEOUT);
conn.setRequestMethod("POST");

// Compose Developer Authorization String
String authorization = null;
try {
    authorization = composeAuthorizationHeader();
} catch (Exception e) {
    StringBuilder msg = new StringBuilder();
    msg.append("Composing developer authorization string error:")
            .append(e.getMessage());
    throw new MalformedURLException(msg.toString());
}

// Setting developer authorization string into header
conn.addRequestProperty("Authorization", authorization);

StringBuilder cookie = new StringBuilder();
cookie.append("sid=").append(ApiCookies.sid).append(";").append("c=")
        .append(ApiCookies.c_ClientType).append(";").append("v=")
        .append(ApiCookies.v_ClientVersion).append(";")
        .append("EEE_MANU=").append(ApiCookies.EEE_MANU_Maunfactory)
        .append(";").append("EEE_PROD=")
        .append(ApiCookies.EEE_PROD_ProductModal).append(";")
        .append("OS_VER=").append(Build.VERSION.SDK).append(";");

conn.addRequestProperty("cookie", cookie.toString());
conn.setDoOutput(true);
conn.setDoInput(true);
try {
    conn.connect();
} catch (IOException ioe) {
    Log.e(TAG, "Get Connection Error:" + ioe.getMessage(), ioe);
    throw ioe;
}

// OUT
OutputStream out = conn.getOutputStream();
byte[] bytes = params.getBytes("UTF8");
out.write(bytes);
```

```
        out.flush();
        out.close();

        InputStream in = conn.getInputStream();
        Xml.parse(in, Xml.Encoding.UTF_8, handler);
        conn.disconnect();
        ApiResponse response = handler.getResponse();
        Log.d(TAG, "sStatus:" + String.valueOf(response.getStatus()));
        return response;
    }

    public String composeAuthorizationHeader() throws Exception {
        if (ApiCookies.progKey == null
                || ApiCookies.progKey.trim().length() == 0) {
            throw new Exception("There's no program key!");
        }
        StringBuilder authorization = new StringBuilder();
        String nonce = UUID.randomUUID().toString().replaceAll("-", "");
        String timestamp = String.valueOf((long) Calendar.getInstance()
                .getTimeInMillis());
        String signature = null;

        // Step 1, Compose signature string
        StringBuilder signaturePre = new StringBuilder();
        signaturePre.append("nonce=").append(nonce)
                .append("&signature_method=").append(SIGNATURE_METHOD)
                .append("&timestamp=").append(timestamp);

        // Step 2, Doing urlencode before doing hash
        String signatureURLEn = URLEncoder.encode(signaturePre.toString(),
                "UTF-8");
        // Java Only Support HMACSHA1
        String signMethod = SIGNATURE_METHOD.replaceAll("-", "");
        // Step 3, Doing hash signature string by HMAC-SHA1
        SecretKey sk = new SecretKeySpec(ApiCookies.progKey.getBytes("UTF-8"),
                signMethod);
        Mac m = Mac.getInstance(signMethod);
        m.init(sk);
        byte[] mac = m.doFinal(signatureURLEn.getBytes("UTF-8"));
        // Step 4, Doing base64 encoding & doing urlencode again
```

```
        signature = URLEncoder.encode(new String(Base64.encodeToByte(mac),
                "UTF-8"), "UTF-8");
        // Final step, Put all parameters to be authorization header string
        authorization.append("signature_method=\"").append(SIGNATURE_METHOD)
                .append("\",").append("timestamp=\"").append(timestamp)
                .append("\",").append("nonce=\"").append(nonce).append("\",")
                .append("signature=\"").append(signature).append("\"");

        return authorization.toString();
    }
}
```

必要資料說明: **開發者授權傳送規格**：

- 在 Cookie 中指定 sid 欄位，Cookie 中的「sid」必須為小寫英文字母。

  `cookie.append("sid=").append(ApiCookies.sid)`

- 在 Header 中必須帶入 Authorization Header，包含 signature_method、timestamp、nonce 以及 signature 四個參數。
  - **signature_method**：hash 的方式，目前提供以 HMAC-SHA1 演算法做 Hash。
  - **timestamp**：從 1970/01/01 00:00:00 到現在為止的毫秒數。
  - **nonce**：唯一且僅有的亂數，此值在 60 分鐘內不可重覆出現相同的值。

signature：將上述三參數，依字母排列規格以 Query String 方式串接後做 URLEncode，再以 **ProgKey** 為 Key 值，進行指定 signature_method 的 **Hash** 演算，再將 Hash 過的字串加以 **Base64** 的轉換，最後再將 Base64 後的結果，再進行一次 **URLEncode** 的字串，即為 signature 字串。

```
// Step 1, Compose signature string
        StringBuilder signaturePre = new StringBuilder();
        signaturePre.append("nonce=").append(nonce)
                .append("&signature_method=").append(SIGNATURE_METHOD)
                .append("&timestamp=").append(timestamp);

        // Step 2, Doing urlencode before doing hash
        String signatureURLEn = URLEncoder.encode(signaturePre.toString(),
                "UTF-8");

        // Java Only Support HMACSHA1
        String signMethod = SIGNATURE_METHOD.replaceAll("-", "");
```

```
        // Step 3, Doing hash signature string by HMAC-SHA1
        SecretKey sk = new SecretKeySpec(ApiCookies.progKey.getBytes("UTF-8"),
                signMethod);
        Mac m = Mac.getInstance(signMethod);
        m.init(sk);
        byte[] mac = m.doFinal(signatureURLEn.getBytes("UTF-8"));
        // Step 4, Doing base64 encoding & doing urlencode again
        signature = URLEncoder.encode(new String(Base64.encodeToByte(mac),
                "UTF-8"), "UTF-8");
```

由於需要製作要求 Request 使用的 XML 與 解析回傳的 XML 因此

RequestServiceGatewayRequest.java

```
public class RequestServiceGatewayRequest {

    public RequestServiceGatewayRequest(){}

    public RequestServiceGatewayRequest(String uid, String pwd){
        this._userid = uid;
        this._password = pwd;
    }

    private String _userid;
    public String getUserid(){ return this._userid; }
    public void setUserid(String value){ this._userid = value; }

    private String _password;
    public String getPassword(){ return this._password; }
    public void setPassword(String value){ this._password = value; }

    private String _language="en_US";
    public String getLanguage(){ return this._language; }
    public void setLanguage(String value){ this._language = value; }

    private String _service="1";
    public String getService(){ return this._service; }
    public void setService(String value){ this._service = value; }
```

```
private String _client_c="0";
public String getClientC(){ return this._client_c; }
public void setClient(String value){ this._client_c = value; }

private String _client_ver="2.2.0.0";
public String getClientVer(){ return this._client_ver; }
public void setClientVer(String value){ this._client_ver = value; }

private String _os_name="Android";
public String getOsName(){ return this._os_name; }
public void setOsName(String value){ this._os_name = value; }

private String _os_ver=android.os.Build.VERSION.RELEASE;
public String getOsVer(){ return this._os_ver; }
public void setOsVer(String value){ this._os_ver = value; }

private String _time=String.valueOf(System.currentTimeMillis());
public String getTime(){ return this._time; }
public void setTime(String value){ this._time = value; }

public String toXml(){
    XmlSerializer serializer = Xml.newSerializer();
    StringWriter writer = new StringWriter();
    try {
        serializer.setOutput(writer);
        serializer.startDocument("UTF-8", true);
        serializer.startTag("", "requestservicegateway");
        serializer.startTag("", "userid");
        serializer.text(this._userid);
        serializer.endTag("", "userid");
        serializer.startTag("", "password");
        serializer.text(this._password);
        serializer.endTag("", "password");
        serializer.startTag("", "language");
        serializer.text(this._language);
        serializer.endTag("", "language");
        serializer.startTag("", "service");
        serializer.text(this._service);
        serializer.endTag("", "service");
        serializer.startTag("", "client");
```

```
            serializer.startTag("", "c");
            serializer.text(this._client_c);
            serializer.endTag("", "c");
            serializer.startTag("", "ver");
            serializer.text(this._client_ver);
            serializer.endTag("", "ver");
            serializer.endTag("", "client");
            serializer.startTag("", "os");
            serializer.startTag("", "name");
            serializer.text(this._os_name);
            serializer.endTag("", "name");
            serializer.startTag("", "ver");
            serializer.text(this._os_ver);
            serializer.endTag("", "ver");
            serializer.endTag("", "os");
            serializer.startTag("", "time");
            serializer.text(this._time);
            serializer.endTag("", "time");
            serializer.endTag("", "requestservicegateway");
            serializer.endDocument();
            return writer.toString();
        } catch (Exception e) {
            throw new RuntimeException(e);
        }
    }
}// end class
```

RequestServiceGatewayResponse.java

```
public class RequestServiceGatewayResponse extends ApiResponse{


    private String _servicegateway;
    public String getServicegateway(){ return this._servicegateway; }
    public void setServicegateway(String value){ this._servicegateway = value; }
    private String _time;
    public String getTime(){ return this._time; }
    public void setTime(String value){ this._time = value; }
    public String toXml(){
        XmlSerializer serializer = Xml.newSerializer();
```

```
        StringWriter writer = new StringWriter();
        try {
            serializer.setOutput(writer);
            serializer.startDocument("UTF-8", true);
            serializer.startTag("", "requestservicegateway");
            serializer.startTag("", "status");
            serializer.text(String.valueOf(this._status));
            serializer.endTag("", "status");
            serializer.startTag("", "servicegateway");
            serializer.text(this._servicegateway);
            serializer.endTag("", "servicegateway");
            serializer.startTag("", "time");
            serializer.text(this._time);
            serializer.endTag("", "time");
            serializer.endTag("", "requestservicegateway");
            serializer.endDocument();
            return writer.toString();
        } catch (Exception e) {
            throw new RuntimeException(e);
        }
    }
}// end class
```

RequestServiceGateway.java

```
public class RequestServiceGateway extends BaseSaxHandler{

    private RequestServiceGatewayResponse response = new RequestServiceGatewayResponse();

    @Override
    public void endElement(String uri, String localName, String qName)
            throws SAXException {
        super.endElement(uri, localName, qName);
       if (localName.equalsIgnoreCase("status")){
        response.setStatus(Integer.parseInt(builder.toString().trim()));
       } else if (localName.equalsIgnoreCase("servicegateway")){
        response.setServicegateway(builder.toString().trim());
       } else if (localName.equalsIgnoreCase("time")){
        response.setTime(builder.toString().trim());
       }
```

```
        builder.setLength(0);
    }
    @Override
    public ApiResponse getResponse() {
        return this.response;
    }
}
```

ApiResponse.java

```
public abstract class ApiResponse {
    protected int _status;
    public int getStatus(){ return this._status; }
    public void setStatus(int value){ this._status = value; }
}
```

建立一個 HELPER 並將上述程式結合在一起:

RequestServiceGatewayHelper.java

```
public class RequestServiceGatewayHelper extends BaseHelper{

    @Override
    protected ApiResponse doApi(ApiConfig apicfg) throws MalformedURLException, ProtocolException,
IOException, SAXException {

        RequestServiceGatewayRequest request = new RequestServiceGatewayRequest(apicfg.userid,
apicfg.password);
        ServicePortalApi spl = new ServicePortalApi(ApiConfig.SERVICEPORTAL);
        return  spl.requestServiceGateway(request);

    }
}
```

這時候修改先前的 LoginTask.java 使用並取的相對應的資料。

```
@Override
    protected Integer doInBackground(Void... params) {
        this.publishProgress(0);
```

```
    StringBuilder msg = new StringBuilder();
    // Fetching ServiceGateway Domain from ServicePortal By UserId & hashed
    // Password
    if (apiCfg.ServiceGateway == null
            || apiCfg.ServiceGateway.trim().length() == 0) {
        msg.delete(0, msg.length());
        try {
            RequestServiceGatewayHelper rsgh = new RequestServiceGatewayHelper();
            RequestServiceGatewayResponse rsp = (RequestServiceGatewayResponse) rsgh
                    .process(apiCfg);
            if (rsp.getStatus() == 0) {
                apiCfg.ServiceGateway = rsp.getServicegateway();

                // Only for lab debugging
                if (apiCfg.ServiceGateway.lastIndexOf(":") > 0) {
                    int pos = apiCfg.ServiceGateway.lastIndexOf(":");
                    apiCfg.ServiceGateway = apiCfg.ServiceGateway
                            .substring(0, pos);
                }

                msg.append("Got ServiceGateway : [")
                        .append(apiCfg.ServiceGateway).append("]");
                Log.d(TAG, msg.toString());

            } else {
                msg.append("Request ServiceGateway Fail, Status:").append(
                        rsp.getStatus());
                Log.e(TAG, msg.toString());
                return rsp.getStatus();
            }
        } catch (APIException e) {
            msg.append("Request ServiceGateway Error:").append(
                    e.getMessage());
            Log.e(TAG, msg.toString(), e);
            return e.status;
        }
    }
    return 0;
}
```

並將 LoginTask 這支加入 LoginActivity 因此修改 LoginActivity 中 loginFunction

```
public void loginFunction(View v) {
        StringBuilder msg = new StringBuilder();

        String uid = txtUid.getText().toString();
        String pwd = txtPwd.getText().toString();

        if (uid.trim().length() == 0 || pwd.trim().length() == 0) {
            msg.delete(0, msg.length());
            msg.append("Login info could not be empty!");
            Log.e(TAG, msg.toString());
            Toast.makeText(actLogin, msg.toString(), Toast.LENGTH_LONG).show();
            return;
        }

        apiCfg.userid = uid.trim();

        // Hash password :
        // 1. make lower case of password
        // 2. Doing hash by MD5
        // 3. Converting to be Hex Text
        apiCfg.password = MD5.encode(pwd.trim().toLowerCase());

        // Switch to my space activity
        Intent intent = new Intent();
        intent.setClass(actLogin, MyBrowseActivity.class);

        LoginTask loginTask = new LoginTask(actLogin, apiCfg, intent) {

            @Override
            protected void loginFail(Integer result) {
                txtUid.setText("");
                txtPwd.setText("");
                    //錯誤訊息
                    finish();
                    return;

            }
        };
```

```
        loginTask.execute((Void) null);
    }
```

其中由於使用者密碼使用 MD5 編碼方式傳遞所以須先將此字串編譯過。而 `loginFail` 則為登入錯誤訊息。

當取得 Service Gateway 後，藉由使用 Service Gateway 底下的**/member/acquiretoken/**取的其他相關 Service 與 token。**/member/acquiretoken/**所需的 XML 如下:

```
<?xml version="1.0" encoding="utf-8"?>
<aaa>
    <userid>{ User ID }</userid>
    <password>{  使用者密碼轉小寫經 MD5 編碼後的字串  }</password>
</aaa>
```

呼叫**API**網址為 https:// API Server Domain” + “API路徑”。所以當要呼叫**/member/acquiretoken/**時，Service Gateway 為：sg01.asuswebstorage.com(筆者帳號所取得，若不同帳號會因地區而有所不同)。所組合出來呼叫的網址為:https://sg01.asuswebstorage.com**/member/acquiretoken/**。
回傳的 XML 為:

```
<?xml version="1.0" encoding="utf-8"?>
<aaa>
    <status>{ Status Code }</status>
    <token>{ Token }</token>
    <!--  以下 IP:PORT 資料，不填寫 PORT 時預設值為 80 -->
    <inforelay>{ InfoRelay 的  IP:PORT Ex: 192.168.1.201:8081 }</inforelay>
    <webrelay>{ WebRelay 的 IP:PORT }</webrelay>
    <searchserver>{ Search Server 的 IP:Port }</searchserver>
    <credential>{  有使用 OTP 認證機制之用戶的 Credential ID。}
    </credential><!--  未使用此機制的用戶此欄為空字串  -->
    <credentialstate>{  有使用 OTP 認證機制之用戶此欄顯示 Credential ID 的現行狀態常
    數。}</credentialstate><!--未使用此機制的用戶此欄為空字串  -->
    <package>
        <id>{ Package ID }</id>
        <display>{ EeePC-20G }</display><!-- package name -->
        <capacity>{  容量大小。Ex:2000 }</capacity><!--  計量單位 MB -->
        <uploadbandwidth>{  頻寬。Ex:128 }</uploadbandwidth><!--  計量單位 KB -->
        <downloadbandwidth>{  頻寬。Ex:128 }</downloadbandwidth><!--  計量單位 KB -->
        <upload>{ Ex:128 }</upload><!--  計量單位 MB -->
        <download>{ Ex:128 }</download><!--  計量單位 MB -->
```

```
        <concurrentsession>{ # 幾個 session(數值) }</concurrentsession>
        <maxfilesize>{ 數值 }</maxfilesize><!-- 單位為 MB -->
        <hasencryption>[ 0 | 1 ]</hasencryption >
        <expire>{ 到期日(格林威治時間)，yyyy-MM-dd HH:mm:ss }</expire>
        <maxbackuppc>{ 備份電腦數上限 }</maxbackuppc>
    </package>
    <!-- 當 Status Code 為 504、508 時表示必須使用 Auxiliary Password 驗證 -->
    <auxpasswordurl>{ 若為 OTP 則此欄為空字串 | 若為 CAPTCHA 則此欄為圖型驗證碼的 URL(網址經過 URL encoded) }</auxpasswordurl>
    <time>{ time stamp, this is for different the payload }</time>
</aaa>
```

同時需要取得其他的 Service Domain，並傳送解析相關的 XML 所以需要下列 Class 以做解析。

1. AcquireTokenRequest.java

```
public class AcquireTokenRequest {

    public AcquireTokenRequest(){}

    public AcquireTokenRequest(String uid, String pwd){
        this._userid = uid;
        this._password = pwd;
    }

    public AcquireTokenRequest(String uid, String pwd, String auxpassword){
        this._userid = uid;
        this._password = pwd;
        this.auxpassword = auxpassword;
    }

    private String auxpassword;
    public String getAuxpassword()
    {
        return auxpassword;
    }
    public void setAuxpassword(String auxpassword)
    {
        this.auxpassword = auxpassword;
    }
```

```
private String _userid;
public String getUserid(){ return this._userid; }
public void setUserid(String value){ this._userid = value; }

private String _password;
public String getPassword(){ return this._password; }
public void setPassword(String value){ this._password = value; }

private String _token = null;
public String getToken(){ return this._token; }
public void setToken(String value){ this._token = value; }

private String _time=String.valueOf(System.currentTimeMillis());
public String getTime(){ return this._time; }
public void setTime(String value){ this._time = value; }

public String toXml(){
    XmlSerializer serializer = Xml.newSerializer();
    StringWriter writer = new StringWriter();
    try {
        serializer.setOutput(writer);
        serializer.startDocument("UTF-8", true);
        serializer.startTag("", "aaa");
        serializer.startTag("", "userid");
        serializer.text(this._userid);
        serializer.endTag("", "userid");
        serializer.startTag("", "password");
        serializer.text(this._password);
        serializer.endTag("", "password");
        if (this._token!=null){
            serializer.startTag("", "token");
            serializer.text(this._token);
            serializer.endTag("", "token");
        }
        serializer.startTag("", "time");
        serializer.text(this._time);
        serializer.endTag("", "time");
        if(this.auxpassword!=null && this.auxpassword.trim().length()>0){
            serializer.startTag("", "auxpassword");
```

```
                serializer.text(this.auxpassword);
                serializer.endTag("", "auxpassword");
            }
            serializer.endTag("", "aaa");
            serializer.endDocument();
            return writer.toString();
        } catch (Exception e) {
            throw new RuntimeException(e);
        }
    }
}// end class
```

2. AcquireTokenResponse.java

```
public class AcquireTokenResponse extends ApiResponse{
    private static final String TAG = "AcquireTokenResponse";
    boolean isFutureList = false;

    private PackageInfo packageinfo;
    public PackageInfo getPackageinfo() {return packageinfo;}
    public void setPackageinfo(PackageInfo packageinfo) {this.packageinfo = packageinfo;}

    private String _credential;
    public String getCredential() { return this._credential; }
    public void setCredential(String value) { this._credential = value; }

    private String _credentialState;
    public String getCredentialState() { return this._credentialState; }
    public void setCredentialState(String value) { this._credentialState = value; }

    private String _token;
    public String getToken(){ return this._token; }
    public void setToken(String value){ this._token = value; }

    private String _inforelay;
    public String getInforelay(){ return this._inforelay; }
    public void setInforelay(String value){
        this._inforelay = value;
        Log.d(TAG, "setInforelay=" + value);
    }
```

```
    private String _webrelay;
    public String getWebrelay(){ return this._webrelay; }
    public void setWebrelay(String value){
        this._webrelay = value;
        Log.d(TAG, "setWebrelay=" + value);
    }

    private String _time;
    public String getTime(){ return this._time; }
    public void setTime(String value){ this._time = value; }

    private String auxpasswordurl;
    public String getAuxpasswordurl()
    {
        return auxpasswordurl;
    }
    public void setAuxpasswordurl(String auxpasswordurl)
    {
        this.auxpasswordurl = auxpasswordurl;
    }

    public String toString()
    {
        StringBuilder msg = new StringBuilder();

        msg.append("Status:").append(this._status).append("\n")
           .append("Token:").append(this._token).append("\n")
           .append("InfoRelay:").append(this._inforelay).append("\n")
           .append("WebRelay:").append(this._webrelay).append("\n")
           .append("PackageDisplay:").append(this.packageinfo != null ?
this.packageinfo.getDisplay() : null)  ;

        return msg.toString();
    }

}// end class
```

3. ServiceGatewayApi.java

```
public class ServiceGatewayApi extends BaseApi {
```

```
    private static final String TAG = "ServiceGatewayApi";

    public ServiceGatewayApi(String Url) {
        super(Url);
    }

    public AcquireTokenResponse acquireToken( AcquireTokenRequest request) throws IOException,
SAXException{
        String params = request.toXml();
        Log.d(TAG, params);
        return (AcquireTokenResponse)super.getResponse("/member/acquiretoken/", params, new
AcquireToken());
    }
}
```

4. AcquireTokenHelper.java

```
public class AcquireTokenHelper extends BaseHelper{
    private String auxpassword = null;
    private String TAG="AcquireTokenHelper";
    @Override
    protected ApiResponse doApi(ApiConfig apicfg) throws MalformedURLException,
ProtocolException,IOException, SAXException {

        AcquireTokenRequest request = null;

        if(this.auxpassword==null || this.auxpassword.trim().length()==0)
            request = new AcquireTokenRequest(apicfg.userid, apicfg.password);
        else
            request = new AcquireTokenRequest(apicfg.userid, apicfg.password, this.auxpassword);

        ServiceGatewayApi sgw = new ServiceGatewayApi(apicfg.ServiceGateway);
        return sgw.acquireToken(request);
    }

    public String getAuxpassword()
    {
        return auxpassword;
    }
    public void setAuxpassword(String auxpassword)
    {
```

```
        this.auxpassword = auxpassword;
    }

}
```

完成之後使用AcquireTokenHelper取得相關資訊。因此修改LoginTask.JAVA因同時加入登入及錯誤判斷。

```
public class LoginTask extends AsyncTask<Void, Integer, Integer> {
    String TAG = "LoginTask";
    ProgressDialog _mdialog;
    Context ctx;
    ApiConfig apiCfg;

    Intent startIntent;

    public LoginTask(Context cxt, ApiConfig apiCfg, Intent startIntent) {
        this.ctx = cxt;
        this.apiCfg = apiCfg;
        this.startIntent = startIntent;



    }

    @Override
    protected Integer doInBackground(Void... params) {
        this.publishProgress(0);

        StringBuilder msg = new StringBuilder();

        // Fetching ServiceGateway Domain from ServicePortal By UserId & hashed
        // Password
        if (apiCfg.ServiceGateway == null
                || apiCfg.ServiceGateway.trim().length() == 0) {
            msg.delete(0, msg.length());
            try {
                RequestServiceGatewayHelper rsgh = new RequestServiceGatewayHelper();
                RequestServiceGatewayResponse rsp = (RequestServiceGatewayResponse) rsgh
                        .process(apiCfg);
                if (rsp.getStatus() == 0) {
                    apiCfg.ServiceGateway = rsp.getServicegateway();
```

```
            // Only for lab debugging
            if (apiCfg.ServiceGateway.lastIndexOf(":") > 0) {
                int pos = apiCfg.ServiceGateway.lastIndexOf(":");
                apiCfg.ServiceGateway = apiCfg.ServiceGateway
                        .substring(0, pos);
            }

            msg.append("Got ServiceGateway : [")
                    .append(apiCfg.ServiceGateway).append("]");
            Log.d(TAG, msg.toString());

        } else {
            msg.append("Request ServiceGateway Fail, Status:").append(
                    rsp.getStatus());
            Log.e(TAG, msg.toString());
            return rsp.getStatus();
        }
    } catch (APIException e) {
        msg.append("Request ServiceGateway Error:").append(
                e.getMessage());
        Log.e(TAG, msg.toString(), e);
        return e.status;
    }
}

// Acquire Token from ServiceGateway
try {
    msg.delete(0, msg.length());
    AcquireTokenHelper ath = new AcquireTokenHelper();
    AcquireTokenResponse rsp = (AcquireTokenResponse) ath.process(apiCfg);

    if (rsp.getStatus() != 0) {
        msg.append("Acquire Token Fail, Status:[").append(rsp.getStatus()).append("]");
        Log.e(TAG, msg.toString());
        return rsp.getStatus();

    }

    //
```

```
            // Fetcing Token & User Package Info
            apiCfg.token = rsp.getToken();
            apiCfg.infoRelay = rsp.getInforelay();
            apiCfg.webRelay = rsp.getWebrelay();
            apiCfg.packageDisplay = rsp.getPackageinfo().getDisplay();
            apiCfg.capacity = rsp.getPackageinfo().getCapacity();
            apiCfg.expireDate = rsp.getPackageinfo().getExpire();

            msg.append("Got the token:[").append(apiCfg.token).append("]\n")
                    .append("InfoRelay:[").append(apiCfg.infoRelay)
                    .append("]\n").append("WebRelay:[").append(apiCfg.webRelay)
                    .append("]\n").append("Package:[")
                    .append(apiCfg.packageDisplay).append("]\n")
                    .append("Capacity:[").append(apiCfg.capacity).append("]\n")
                    .append("ExpireDate:[").append(apiCfg.expireDate)
                    .append("]");

            Log.i(TAG, msg.toString());
            } catch (APIException e) {
            msg.append("Acquire Token Error:").append(e.getMessage());
            Log.e(TAG, msg.toString(), e);
            return e.status;
        }
        return 0;
    }

    @Override
    protected void onCancelled() {
        super.onCancelled();

        if (_mdialog != null) {
            try {
                _mdialog.dismiss();
            } catch (Exception e) {
            }
        }
    }

    @Override
    protected void onPostExecute(Integer result) {
```

```
        this.publishProgress(100);
        switch (result) {
        case APIException.GENERAL_SUCC:
            goNextActivity();
            break;
        case APIException.EXC_AAA: // UserId & password authorize fail
            loginFail(result);
            break;
        case APIException.EXC_OAUTH: // SID & ProgKey authorize fail
            authFail(result);
            break;
        default:
            loginFail(result);
            break;
        }
    }

    @Override
    protected void onProgressUpdate(Integer... values) {
        super.onProgressUpdate(values);
        if (values[0] == 0) {
            try {
                _mdialog = ProgressDialog
                        .show(ctx,

ctx.getString(net.yostore.aws.view.common.R.string.app_name),
                                "Connecting to server...", true, true,
                                new OnCancelListener() {
                                    @Override
                                    public void onCancel(DialogInterface dialog) {
                                    }
                                });
            } catch (Exception e) {
            }
        } else {
            if (_mdialog != null) {
                try {
                    _mdialog.dismiss();
                } catch (Exception e) {
                }
```

```
            }
        }
    }

    protected void loginFail(Integer result) {
        ((Activity) ctx).finish();
    }

    protected void authFail(Integer result) {
        // MessageDialog.show(ctx,
        // ctx.getString(R.string.app_name),"Application authorizing fail!");
        ((Activity) ctx).finish();
    }

    protected void goNextActivity() {
        ctx.startActivity(this.startIntent);
        ((Activity) ctx).finish();
    }
}
```

2・檔案讀取

當進入登入畫面後，首先需要讀取檔案列表。由於所有的檔案根目錄為MySyncFloder因此，需要取得MySyncFloder ID方可繼續使用下去。因此需要使用InfoRelay 下/folder/getmysyncfolder/ 這支API

此範例所使用的網址為:https://ir01.asuswebstorage.com/**member/acquiretoken/**
(呼叫 **API** 網址：https:// 使用者 InfoRelay Domain” + “/folder/getmysyncfolder/)。

API 所需 XML 格式：

```
<?xml version="1.0" encoding="utf-8" ?>
<getmysyncfolder>
    <userid>{ User ID }</userid>
    <token>{ Token }</token>
</getmysyncfolder>
```

回傳 XML:

```
<?xml version="1.0" encoding="utf-8" ?>
<getmysyncfolder>
    <status>{ Status Code }</status>
    <id>{ Folder ID }</id>
```

</getmysyncfolder>

傳送解析相關的XML所以需要下列Class以做解析。

1. GetMySyncFolderRequest.java

```
public class GetMySyncFolderRequest
{
    private String _token;
    private String _userId;

    public GetMySyncFolderRequest(){};
    public GetMySyncFolderRequest(String token, String uid)
    {
        this._token  = token;
        this._userId = uid;
    }
    public String getToken()
    {
        return _token;
    }
    public void setToken(String _token)
    {
        this._token = _token;
    }
    public String getUserId()
    {
        return _userId;
    }
    public void setUserId(String _userId)
    {
        this._userId = _userId;
    }
    public String toXml(){
        XmlSerializer serializer = Xml.newSerializer();
        StringWriter writer = new StringWriter();
        try {
            serializer.setOutput(writer);
            serializer.startDocument("UTF-8", true);
            serializer.startTag("", "getmysyncfolder");
            serializer.startTag("", "token");
```

```
            serializer.text(this._token);
            serializer.endTag("", "token");
            serializer.startTag("", "userid");
            serializer.text(this._userId);
            serializer.endTag("", "userid");
            serializer.endTag("", "getmysyncfolder");
            serializer.endDocument();
            return writer.toString();

        } catch (Exception e) {
            throw new RuntimeException(e);
        }
    }
}
```

2. GetMySyncFolderResponse.java

```
public class GetMySyncFolderResponse extends ApiResponse
{
    private String _id;

    public String getId()
    {
        return _id;
    }

    public void setId(String _id)
    {
        this._id = _id;
    }
    public String toXml(){
        XmlSerializer serializer = Xml.newSerializer();
        StringWriter writer = new StringWriter();
        try {
            serializer.setOutput(writer);
            serializer.startDocument("UTF-8", true);
            serializer.startTag("", "getmysyncfolder");
            serializer.startTag("", "status");
            serializer.text(String.valueOf(this._status));
            serializer.endTag("", "status");
            serializer.startTag("", "id");
            serializer.text(String.valueOf(this._id));
```

```
            serializer.endTag("", "id");
            serializer.endTag("", "getmysyncfolder");
            serializer.endDocument();
            return writer.toString();
        } catch (Exception e) {
            throw new RuntimeException(e);
        }
    }
}
```

3. GetMySyncFolderHelper.java

```
public class GetMySyncFolderHelper extends BaseHelper
{

    @Override
    protected ApiResponse doApi(ApiConfig apicfg) throws MalformedURLException, ProtocolException,
IOException, SAXException
    {
        GetMySyncFolderRequest request = new GetMySyncFolderRequest(apicfg.token, apicfg.userid);

        InfoRelayApi ir = new InfoRelayApi(apicfg.infoRelay);
        return ir.getMySyncFolder(request);
    }

}
```

4. InfoRelayApi.java

```
public class InfoRelayApi extends BaseApi {
    private static final String TAG = "InfoRelayApi";
    public InfoRelayApi(String Url) {
        super(Url);
    }


    public GetMySyncFolderResponse getMySyncFolder(GetMySyncFolderRequest request) throws
IOException, SAXException{
        String params = request.toXml();
        Log.d(TAG, params);
        return (GetMySyncFolderResponse)super.getResponse("/folder/getmysyncfolder/", params,
new GetMySyncFolder());
    }
```

```
}
```

並將下列程式碼加入在LoginTask.java之中。

```
//Finding MySyncFolder real folderId
        if ( apiCfg.mySyncFolderId == null || apiCfg.mySyncFolderId.trim().length() == 0 )
        {
            msg.delete(0, msg.length());

            // if SyncFolderId is empty, do propFind to get sync folderId
            GetMySyncFolderHelper mySyncFolder = new GetMySyncFolderHelper();
            try
            {
                GetMySyncFolderResponse rsp =
(GetMySyncFolderResponse)mySyncFolder.process(apiCfg);
                if ( rsp.getStatus() != APIException.GENERAL_SUCC )
                {
                    msg.append("Get sync folder fail, status:").append(rsp.getStatus());
                    Log.e(TAG, msg.toString());

                    return rsp.getStatus();
                }
                apiCfg.mySyncFolderId = rsp.getId();
                apiCfg.parentFolderId = apiCfg.SYNCROOTID;
                apiCfg.currentFolderId= rsp.getId();
                apiCfg.folderName       = apiCfg.SYNCFOLDERNAME;
            }
            catch ( APIException e )
            {
                msg.append("Get sync folder error:").append(e.getMessage());
                Log.e(TAG, msg.toString(), e);

                return APIException.GENERAL_ERR;
            }
```

並使用infoRelay 中/folder/browse/ 瀏覽資料夾。此時利用android listview去作為檔案列表所以需要自訂ListView 所使用的UI檔案如下:

1. s_browse.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="?android:attr/listPreferredItemHeight"
    android:background="@drawable/listview_background" >
    <ProgressBar android:id="@+android:id/progress"
        android:layout_width="36dip"
        android:layout_height="36dip"
        android:layout_marginLeft="15dip"
        android:layout_marginRight="8dip"
        android:layout_gravity="center_vertical"
        android:visibility="gone"/>
    <ImageView
         android:id="@+id/iconid"
        android:layout_width="36dip"
        android:layout_height="fill_parent"
        android:layout_marginLeft="15dip"
        android:layout_marginRight="8dip"
        android:src="@drawable/icon_list_doc"
        android:visibility="visible" />
    <LinearLayout
        android:orientation="vertical"
        android:layout_width="0dip"
        android:layout_weight="1"
        android:layout_height="fill_parent">
        <TextView
            android:id="@+id/toptext"
            android:layout_width="fill_parent"
            android:layout_height="0dip"
            android:layout_weight="1"
            android:text="item"
            android:textColor="#4b78bb"
            android:textSize="15sp"
            android:gravity="bottom"
            android:singleLine="true"
            android:visibility="visible" />
        <LinearLayout
             android:orientation="horizontal"
          android:layout_width="fill_parent"
```

```
          android:layout_height="0dip"
            android:layout_weight="1"
              android:gravity="center_vertical">

             <TextView
                 android:id="@+id/bottomtext"
                 android:layout_width="0dp"
               android:layout_weight="1"
                 android:layout_height="wrap_content"
                 android:text="26MB, Last Modified-2010/01/01 14:20"
                 android:textColor="#9d9d9d"
                 android:textSize="11sp"
                 android:singleLine="true"
                 android:visibility="visible"
               />
            </LinearLayout>
    </LinearLayout>
</LinearLayout>
```

2. s_browse_item.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical" >

    <!-- Header -->

    <LinearLayout
        xmlns:android="http://schemas.android.com/apk/res/android"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:background="@drawable/bg_header"
        android:orientation="horizontal"
        android:padding="5dp" >

        <LinearLayout
            android:layout_width="fill_parent"
            android:layout_height="fill_parent"
```

```
    android:layout_marginLeft="10dp"
    android:layout_weight="2"
    android:gravity="center_vertical" >

    <ImageButton
        android:id="@+id/homeBt"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:background="@android:color/transparent"
        android:src="@drawable/btn_home"
        android:visibility="visible" />

    <ImageButton
        android:id="@+id/backBt"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:background="@android:color/transparent"
        android:onClick="backFunction"
        android:src="@drawable/btn_back"
        android:visibility="gone" />
</LinearLayout>

<LinearLayout
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:layout_weight="1"
    android:gravity="center" >

    <TextView
        android:id="@+id/mPath"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:text="MySyncFolder"
        android:textColor="@color/headertext"
        android:textSize="15sp" />
</LinearLayout>

<LinearLayout
    android:layout_width="fill_parent"
```

```
    android:layout_height="fill_parent"
    android:layout_marginRight="10dp"
    android:layout_weight="2"
    android:gravity="right|center_vertical" >

    <ImageButton
        android:id="@+id/GoRootBt"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:background="@android:color/transparent"
        android:onClick="goHomeFunction"
        android:src="@drawable/btn_root"
        android:visibility="gone" />

    <ImageButton
        android:id="@+id/saveSearchBt"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:background="@android:color/transparent"
        android:onClick="saveSearchFunction"
        android:src="@drawable/btn_save"
        android:visibility="gone" />
</LinearLayout>

<LinearLayout
    android:layout_width="0dip"
    android:layout_height="0dip"
    android:gravity="left|center_vertical"
    android:visibility="gone" >

    <ImageButton
        android:id="@+id/searchBt"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:background="@android:color/transparent"
        android:onClick="searchData"
        android:src="@drawable/icon_search_up" />

    <ImageButton
        android:id="@+id/newBt"
```

```
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:background="@android:color/transparent"
            android:onClick="createNewFunction"
            android:src="@drawable/icon_new" />

        <ImageButton
            android:id="@+id/uploadBt"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:background="@android:color/transparent"
            android:onClick="uploadFunction"
            android:src="@drawable/icon_upload" />
    </LinearLayout>
</LinearLayout>
<!-- Space List -->

<ListView
    android:id="@+id/android:list"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent" />

<ProgressBar
    android:id="@+id/fetchFolderProgress"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:indeterminate="false"
    android:max="100"
    android:orientation="horizontal"
    android:progress="0"
    android:visibility="gone" />

<TextView
    android:id="@+id/android:empty"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:text="@string/common_no_item_found"
    android:visibility="gone" />

</LinearLayout>
```

使用 InfoRelay 底下瀏覽目錄/folder/browse/ API。

此範例所使用的網址為:https://ir01.asuswebstorage.com/folder/browse/
(呼叫 **API** 網址：https:// 使用者 InfoRelay Domain” + “/folder/browse/)。
同樣的需要做XML解析。
/folder/browse/ 傳出XML

```
<?xml version="1.0" encoding="utf-8"?>
<browse>
  <token>{ Token }</token>
  <language>{ 使用者的語系，例如：zh_TW }</language>
  <userid>{ User ID }</userid>
  <folderid>{ Folder ID }</folderid>
  <fileext>{ 過濾出欲索取之檔案的副檔名。副檔名之間以',' (逗號)分隔。例如：jpg,mp3 }</fileext>
  <!-- 選擇性欄位。無此欄位時，系統預設為全部顯示 -->
  <page>
    <pageno>{ 要索取的頁面號碼#(第一頁為 1) }</pageno>
    <pagesize>{ 指定一頁要顯示的筆數 }</pagesize>
    <enable>[ 0 | 1 ]</enable><!-- 0：表示 pagesize 參數無作用，每頁不限筆數(預設值) -->
  </page>
  <filter><!-- 選擇性欄位 -->
    <starttime>{ 開始時間，格式為 yyyyMMddHHmmssSSS }</starttime>
    <endtime>{ 結束時間，格式為 yyyyMMddHHmmssSSS }</endtime>
    <!-- 以 starttime、endtime 限定搜尋出來的目錄，二者須成對出現。表示方式如下例：開始時間為 2010 年 3 月 31 日 15 時 3 分 8 秒 57 毫秒；結束時間為 2010 年 4 月 1 日 15 時 3 分 8 秒 57 毫秒，以此表示為<starttime>20100331150308057</starttime>
    <endtime>20100401150308057</endtime> -->
  </filter>
  <sortby>[ 1 | 2 ]</sortby><!-- 1：以名稱排序 | 2：以最後修改時間(Last ChangeTime)排序 -->
  <sortdirection>[ 0 | 1 ]</sortdirection><!-- 0：升冪排序(ASC) | 1：降冪排序(DESC) -->
  <issibiling>[ 0 | 1 ]</issibiling><!-- 選擇性欄位。0：列出子節點內容(預設值) | 1：列出兄弟目錄內容 -->
</browse>
```

接收的XML格式：

```
<?xml version="1.0" encoding="utf-8" ?>
<browse>
  <status>{ Status Code }</status>
```

<parentfolder>
<name>{ 目錄名稱(Folder Name) }</name>
<id>{ Folder ID }</id>
</parentfolder>
<page>
<pageno>{ 當前頁面號碼#(第一頁為 1) }</pageno><!-- 當 pageno 值小於 1 則伺服器回應值為 1；當指定 pageno 大於最大頁數，則伺服器回應為最後一頁 -->
<pagesize>{ 一頁所包含的目錄 + 檔案之筆數# }</pagesize>
<totalcount>{ 此層目錄 + 檔案總筆數 }</totalcount>
<hasnextpage>[*0* | *1*]</hasnextpage>
</page>
<folder>
<id>{ Folder ID }</id>
<display>{ 以 Base64(UTF-8)編碼的目錄名稱 }</display>
<!-- 若您使用的開發語言為 Ruby，請用 Base64.strict_encode64，避免“\n”在編碼之後出現。 -->
<attribute>{ Data [註 1] }</attribute>
<isencrypted>[*0* | *1*]</isencrypted>
<isowner>[*0* | *1*]</isowner>
<isbackup>[*0* | *1*]</isbackup><!-- 若為 1 表示是備份的檔案或目錄，否則為一般檔案或目錄 -->
<isorigdeleted>[*0* | *1*]</isorigdeleted><!-- 0：原始檔未刪除 | 1：原始檔已刪除 -->
<ispublic>[*0* | *1*]</ispublic><!-- 使用者是否已設定為分享目錄。0：意指 PRIVATE | 1：意指 PUBLIC(已分享) -->
<createdtime>{ yyyy-MM-dd HH:mm:ss [註 1] }</createdtime>
<markid>{ 附屬於該目錄的標示 ID(Mark ID [註 3])，多個標示 ID(Mark ID)之間以空白字元分隔 }</markid>
</folder>
…
<file>
<id>{ File ID }</id>
<status>[*0* | *1*]</status><!-- 表示此檔案是否可被分享。 -->
<display>{ 以 Base64(UTF-8)編碼的檔案名稱 }</display>
<!-- 若您使用的開發語言為 Ruby，請用 Base64.strict_encode64，避免“\n”在編碼之後出現。 -->
<attribute>{ Data [註 1] }</attribute>
<isencrypted>[*0* | *1*]</isencrypted>
<size>{ 檔案大小(File Size) }</size>
<isowner>[*0* | *1*]</isowner>
<isbackup>[*0* | *1*]</isbackup><!-- 若為 1 表示是備份的檔案或目錄，否則為一般檔案或目錄

-->
<isorigdeleted>[*0* | *1*]</isorigdeleted><!-- 0：原始檔未刪除 | 1：原始檔已刪除 -->
<isinfected>[*0* | *1*]</isinfected><!-- 此欄位用以判斷檔案是否感染病毒。0：表示未感染 | 1：表示已感染 -->
<ispublic>[*0* | *1*]</ispublic><!-- 使用者是否已設定為分享，0：意指 PRIVATE | 1：意指 PUBLIC(已分享) -->
<headversion>{ 檔案的 Version Number# }</headversion>
<createdtime>{ yyyy-MM-dd HH:mm:ss [註 1] }</createdtime>
<markid>{ 附屬於此目錄的標示 ID(Mark ID [註 3])，多個標示 ID(Mark ID)之間以空白字元分隔 }</markid>
</file>
…
</browse>

FolderBrowseRequest.java

```
public class FolderBrowseRequest {

    private HashMap<String, String> r2v = new HashMap<String, String>();

    public FolderBrowseRequest() {
        initHashMap();
    }

    public FolderBrowseRequest(String uid, String token, String folderid,
            int sort, int sortByDesc) {
        this._userid = uid;
        this._token = token;
        this._folderid = folderid;
        this.sort = sort;
        this.sortByDesc = sortByDesc;
        initHashMap();
    }

    private void initHashMap() {
        r2v.put("-1", "system.my.encrypted.root");
        r2v.put("-3", "system.backup.root");
        r2v.put("-5", "system.sync.root");
        r2v.put("-7", "system.oeo.root");
        // r2v.put("-9", "");
```

```
        // r2v.put("-11", "");
        r2v.put("-13", "system.searchcriteria.root");
        r2v.put("-15", "system.familymemo.root");
    }

    private String _token;

    public String getToken() {
        return this._token;
    }

    public void setToken(String value) {
        this._token = value;
    }

    private String _scrip = String.valueOf(System.currentTimeMillis());

    public String getScrip() {
        return this._scrip;
    }

    public void setScrip(String value) {
        this._scrip = value;
    }

    private String _language = "en_US";

    public String getLanguage() {
        return this._language;
    }

    public void setLanguage(String value) {
        this._language = value;
    }

    private String _userid;

    public String getUserid() {
        return this._userid;
    }
```

```
public void setUserid(String value) {
    this._userid = value;
}

private String _folderid;

public String getFolderid() {
    return this._folderid;
}

public void setFolderid(String value) {
    this._folderid = value;
}

private String _computerseq = null;

public String getComputerseq() {
    return this._computerseq;
}

public void setComputerseq(String value) {
    this._computerseq = value;
}

private String _fileext = null;

public String getFileext() {
    return this._fileext;
}

public void setFileext(String value) {
    this._fileext = value;
}

private int _pageno = -1;

public int getPageno() {
    return this._pageno;
}
```

```
public void setPageno(int value) {
    this._pageno = value;
}

private int _pagesize = 500;

public int getPagesize() {
    return this._pagesize;
}

public void setPagesize(int value) {
    this._pagesize = value;
}

private boolean pageEnable = false;

public boolean isPageEnable() {
    return pageEnable;
}

public void setPageEnable(boolean pageEnable) {
    this.pageEnable = pageEnable;
}

private long starttime = 0l;

public long getStarttime() {
    return starttime;
}

public void setStarttime(long starttime) {
    this.starttime = starttime;
}

private long endtime = 0l;

public long getEndtime() {
    return endtime;
}
```

```
public void setEndtime(long endtime) {
    this.endtime = endtime;
}

private int sort = 1;
private int sortByDesc = 0;

public int getSort() {
    return sort;
}

public void setSort(int sort) {
    this.sort = sort;
}

public int getSortByDesc() {
    return sortByDesc;
}

public void setSortByDesc(int sortByDesc) {
    this.sortByDesc = sortByDesc;
}

public String toXml() {
    XmlSerializer serializer = Xml.newSerializer();
    StringWriter writer = new StringWriter();
    try {
        serializer.setOutput(writer);
        serializer.startDocument("UTF-8", true);
        serializer.startTag("", "browse");
        serializer.startTag("", "token");
        serializer.text(this._token);
        serializer.endTag("", "token");
        serializer.startTag("", "scrip");
        serializer.text(this._scrip);
        serializer.endTag("", "scrip");
        serializer.startTag("", "language");
        serializer.text(this._language);
        serializer.endTag("", "language");
```

```
serializer.startTag("", "userid");
serializer.text(this._userid);
serializer.endTag("", "userid");
serializer.startTag("", "folderid");
serializer.text(map2virtual());
serializer.endTag("", "folderid");
if (this._computerseq != null) {
    serializer.startTag("", "computerseq");
    serializer.text(this._computerseq);
    serializer.endTag("", "computerseq");
}
if (this._fileext != null) {
    serializer.startTag("", "fileext");
    serializer.text(this._fileext);
    serializer.endTag("", "fileext");
}
if (pageEnable && this._pageno > 0) {
    serializer.startTag("", "page");
    serializer.startTag("", "pageno");
    serializer.text(String.valueOf(this._pageno));
    serializer.endTag("", "pageno");
    serializer.startTag("", "pagesize");
    serializer.text(String.valueOf(this._pagesize));
    serializer.endTag("", "pagesize");
    serializer.startTag("", "enable");
    serializer.text(pageEnable ? "1" : "0");
    serializer.endTag("", "enable");
    serializer.endTag("", "page");
}
if (endtime > starttime && starttime > 0
        && starttime > 199012312359590001) {
    serializer.startTag("", "filter");
    serializer.startTag("", "starttime");
    serializer.text(String.valueOf(this.starttime));
    serializer.endTag("", "starttime");
    serializer.startTag("", "endtime");
    serializer.text(String.valueOf(this.endtime));
    serializer.endTag("", "endtime");
    serializer.endTag("", "filter");
}
```

```
            serializer.startTag("", "sortby");
            serializer.text(String.valueOf(this.sort));
            serializer.endTag("", "sortby");
            serializer.startTag("", "sortdirection");
            serializer.text(String.valueOf(this.sortByDesc));
            serializer.endTag("", "sortdirection");
            serializer.endTag("", "browse");
            serializer.endDocument();
            return writer.toString();
        } catch (Exception e) {
            throw new RuntimeException(e);
        }
    }

    private String map2virtual() {
        return this._folderid;
    }
}// end class
```

FolderBrowseResponse.java

```
public class FolderBrowseResponse extends ApiResponse{
    private List<FileInfo> fileList = new LinkedList<FileInfo>();
    public List<FileInfo> getFileList() {return fileList;}
    public void setFileList(List<FileInfo> fileList) {this.fileList = fileList;}

    private List<FolderInfo> folderList=new LinkedList<FolderInfo>();
    public List<FolderInfo> getFolderList() {return folderList;}
    public void setFolderList(List<FolderInfo> folderList) {this.folderList = folderList;}



    private ParentFolder _parentFolder;
    public ParentFolder getParentFolder(){ return this._parentFolder; }
    public void setParentFolder(ParentFolder value){ this._parentFolder = value; }

    private Attribute _attribute;
    public Attribute getAttribute(){ return this._attribute; }
    public void setAttribute(Attribute value){ this._attribute = value; }

    private String _scrip;
```

```
    public String getScrip(){ return this._scrip; }
    public void setScrip(String value){ this._scrip = value; }

    private int _pageno;
    public int getPageno(){ return this._pageno; }
    public void setPageno(int value){ this._pageno = value; }

    private int _pagesize;
    public int getPagesize(){ return this._pagesize; }
    public void setPagesize(int value){ this._pagesize = value; }

    private int _totalcount;
    public int getTotalcount(){ return this._totalcount; }
    public void setTotalcount(int value){ this._totalcount = value; }

    private boolean _hasnextpage;
    public boolean getHasnextpage(){ return this._hasnextpage; }
    public void setHasnextpage(boolean value){ this._hasnextpage = value; }

}// end class
```

FolderInfo.java

```
public class FolderInfo {
    private String _id;
    public String getId(){ return this._id; }
    public void setId(String value){ this._id = value; }

    private String _display;
    public String getDisplay(){ return this._display; }
    public void setDisplay(String value){ this._display = value; }

    private Attribute _attribute;
    public Attribute getAttribute(){ return this._attribute; }
    public void setAttribute(Attribute value){ this._attribute = value; }

    private boolean _issharing;
    public boolean getIssharing(){ return this._issharing; }
    public void setIssharing(boolean value){ this._issharing = value; }
```

```
private boolean _isencrypted;
public boolean getIsencrypted(){ return this._isencrypted; }
public void setIsencrypted(boolean value){ this._isencrypted = value; }

private boolean _isowner;
public boolean getIsowner(){ return this._isowner; }
public void setIsowner(boolean value){ this._isowner = value; }

private boolean _isbackup;
public boolean getIsbackup(){ return this._isbackup; }
public void setIsbackup(boolean value){ this._isbackup = value; }

private boolean _isorigdeleted;
public boolean getIsorigdeleted(){ return this._isorigdeleted; }
public void setIsorigdeleted(boolean value){ this._isorigdeleted = value; }

private boolean _ispublic;
public boolean getIspublic(){ return this._ispublic; }
public void setIspublic(boolean value){ this._ispublic = value; }

private String _createdtime;
public String getCreatedtime(){ return this._createdtime; }
public void setCreatedtime(String value){ this._createdtime = value; }

private String _markid;
public String getMarkid(){ return this._markid; }
public void setMarkid(String value){ this._markid = value; }

public String toXml(){
    XmlSerializer serializer = Xml.newSerializer();
    StringWriter writer = new StringWriter();
    try {
        serializer.setOutput(writer);
        serializer.startDocument("UTF-8", true);
        serializer.startTag("", "folder");
        serializer.startTag("", "id");
        serializer.text(this._id);
        serializer.endTag("", "id");
        serializer.startTag("", "display");
        serializer.text(this._display);
```

```
            serializer.endTag("", "display");
            serializer.startTag("", "attribute");
            serializer.text(this._attribute.toXml());
            serializer.endTag("", "attribute");
            serializer.startTag("", "issharing");
            serializer.text(this._issharing?"1":"0");
            serializer.endTag("", "issharing");
            serializer.startTag("", "isencrypted");
            serializer.text(this._isencrypted?"1":"0");
            serializer.endTag("", "isencrypted");
            serializer.startTag("", "isowner");
            serializer.text(this._isowner?"1":"0");
            serializer.endTag("", "isowner");
            serializer.startTag("", "isbackup");
            serializer.text(this._isbackup?"1":"0");
            serializer.endTag("", "isbackup");
            serializer.startTag("", "isorigdeleted");
            serializer.text(this._isorigdeleted?"1":"0");
            serializer.endTag("", "isorigdeleted");
            serializer.startTag("", "ispublic");
            serializer.text(this._ispublic?"1":"0");
            serializer.endTag("", "ispublic");
            serializer.startTag("", "createdtime");
            serializer.text(this._createdtime);
            serializer.endTag("", "createdtime");
            serializer.startTag("", "markid");
            serializer.text(this._markid);
            serializer.endTag("", "markid");
            serializer.endTag("", "folder");
            serializer.endDocument();
            return writer.toString();
        } catch (Exception e) {
            throw new RuntimeException(e);
        }
    }
}
```

FolderBrowse.java

```
public class FolderBrowse extends BaseSaxHandler{
```

```
private FolderBrowseResponse response = new FolderBrowseResponse();
boolean isFolder = false;
boolean isFile = false;
boolean isAttr = false;
boolean isAlbumInfo = false;
boolean isParentFolder = false;
FileInfo fi;
FolderInfo fo;
Attribute at;
ParentFolder pf;

@Override
public void endElement(String uri, String localName, String qName)
        throws SAXException {
    super.endElement(uri, localName, qName);

    if (this.isFolder){
        if (localName.equalsIgnoreCase("folder")){
            response.getFolderList().add(fo);
            this.isFolder = false;
        } else if (localName.equalsIgnoreCase("id")){
            fo.setId(builder.toString().trim());
        } else if (localName.equalsIgnoreCase("display")){
            fo.setDisplay(this.b64decode(builder.toString().trim()));
        } else if (localName.equalsIgnoreCase("issharing")){
            fo.setIssharing("1".equalsIgnoreCase(builder.toString().trim())?true:false);
        } else if (localName.equalsIgnoreCase("isencrypted")){

fo.setIsencrypted("1".equalsIgnoreCase(builder.toString().trim())?true:false);
        } else if (localName.equalsIgnoreCase("isowner")){
            fo.setIsowner("1".equalsIgnoreCase(builder.toString().trim())?true:false);
        } else if (localName.equalsIgnoreCase("isbackup")){
            fo.setIsbackup("1".equalsIgnoreCase(builder.toString().trim())?true:false);
        } else if (localName.equalsIgnoreCase("isorigdeleted")){

fo.setIsorigdeleted("1".equalsIgnoreCase(builder.toString().trim())?true:false);
        } else if (localName.equalsIgnoreCase("ispublic")){
            fo.setIspublic("1".equalsIgnoreCase(builder.toString().trim())?true:false);
        } else if (localName.equalsIgnoreCase("createdtime")){
```

```
                fo.setCreatedtime(builder.toString().trim());
            } else if (localName.equalsIgnoreCase("markid")){
                fo.setMarkid(builder.toString().trim());
            }
        } else if(this.isFile){
            if (localName.equalsIgnoreCase("file")){
                response.getFileList().add(fi);
                this.isFile = false;
            } else if (localName.equalsIgnoreCase("id")){
                fi.setId(builder.toString().trim());
            } else if (localName.equalsIgnoreCase("status")){
                fi.setStatus(Integer.parseInt(builder.toString().trim()));
            } else if (localName.equalsIgnoreCase("display")){
                fi.setDisplay(this.b64decode(builder.toString().trim()));
            } else if (localName.equalsIgnoreCase("isencrypted")){

fi.setIsencrypted("1".equalsIgnoreCase(builder.toString().trim())?true:false);
            } else if (localName.equalsIgnoreCase("size")){
                fi.setSize(Long.parseLong(builder.toString().trim()));
            } else if (localName.equalsIgnoreCase("isowner")){
                fi.setIsowner("1".equalsIgnoreCase(builder.toString().trim())?true:false);
            } else if (localName.equalsIgnoreCase("isbackup")){
                fi.setIsbackup("1".equalsIgnoreCase(builder.toString().trim())?true:false);
            } else if (localName.equalsIgnoreCase("isorigdeleted")){

fi.setIsorigdeleted("1".equalsIgnoreCase(builder.toString().trim())?true:false);
            } else if (localName.equalsIgnoreCase("isinfected")){
                fi.setIsinfected("1".equalsIgnoreCase(builder.toString().trim())?true:false);
            } else if (localName.equalsIgnoreCase("ispublic")){
                fi.setIspublic("1".equalsIgnoreCase(builder.toString().trim())?true:false);
            } else if (localName.equalsIgnoreCase("headversion")){
                fi.setHeadversion(Integer.parseInt(builder.toString().trim()));
            } else if (localName.equalsIgnoreCase("createdtime")){
                fi.setCreatedtime(builder.toString().trim());
            } else if (localName.equalsIgnoreCase("markid")){
                fi.setMarkid(builder.toString().trim());
            }

        } else if(isParentFolder){
           if (localName.equalsIgnoreCase("parentfolder")){
```

```
            response.setParentFolder(pf);
            this.isParentFolder = false;
          } else if (localName.equalsIgnoreCase("name")){
                pf.setName(builder.toString().trim());
            } else if (localName.equalsIgnoreCase("id")){
                pf.setId(builder.toString().trim());
            }
        } else{
          if (localName.equalsIgnoreCase("status")){
            response.setStatus(Integer.parseInt(builder.toString().trim()));
            } else if (localName.equalsIgnoreCase("scrip")){
                response.setScrip(builder.toString().trim());
            } else if (localName.equalsIgnoreCase("pageno")){
                response.setPageno(Integer.parseInt(builder.toString().trim()));
            } else if (localName.equalsIgnoreCase("pagesize")){
                response.setPagesize(Integer.parseInt(builder.toString().trim()));
            } else if (localName.equalsIgnoreCase("totalcount")){
                response.setTotalcount(Integer.parseInt(builder.toString().trim()));
            } else if (localName.equalsIgnoreCase("hasnextpage")){

response.setHasnextpage("1".equalsIgnoreCase(builder.toString().trim())?true:false);
            }
        }
        if(isAttr){
          if (localName.equalsIgnoreCase("attribute")){
            if(this.isFile){
                fi.setAttribute(at);
            }else if(this.isFolder){
                fo.setAttribute(at);
            }else{
                response.setAttribute(at);
            }
            this.isAttr=false;
          } else if (localName.equalsIgnoreCase("creationtime")){
                at.setCreationtime(builder.toString().trim());
            } else if (localName.equalsIgnoreCase("lastaccesstime")){
                at.setLastaccesstime(builder.toString().trim());
            } else if (localName.equalsIgnoreCase("lastwritetime")){
                at.setLastwritetime(builder.toString().trim());
            }
```

```
            }
           builder.setLength(0);

        }
        @Override
        public void startElement(String uri, String localName, String qName,
                Attributes attributes) throws SAXException {
            super.startElement(uri, localName, qName, attributes);
            if (localName.equalsIgnoreCase("folder")){
                this.fo = new FolderInfo();
                this.isFolder=true;
            } else if (localName.equalsIgnoreCase("file")){
                this.fi = new FileInfo();
                this.isFile=true;
            }

            if (localName.equalsIgnoreCase("attribute")){
                at = new Attribute();
                this.isAttr = true;
            }

            if (localName.equalsIgnoreCase("parentfolder")){
                pf = new ParentFolder();
                this.isParentFolder = true;
            }
        }
        @Override
        public ApiResponse getResponse() {
            return this.response;
        }
        private String b64decode(String b64){
            try {
                return new String(Base64.decodeFast(b64), "UTF8");
            } catch (UnsupportedEncodingException e) {
                e.printStackTrace();
                return "";
            }
        }
}
```

在InfoRelayApi.java 中加入

```
public FolderBrowseResponse folderBrowse(FolderBrowseRequest request) throws IOException,
SAXException{
        String params = request.toXml();
        Log.d(TAG, params);
        return (FolderBrowseResponse)super.getResponse("/folder/browse/", params, new
FolderBrowse());
    }
```

FolderBrowseHelper.java

```
public class FolderBrowseHelper extends BaseHelper{

    private String parent;
    private int pagesize=500;
    private int page=-1;
    private long starttime = 0l;
    private long endtime = 0l;
    private int sort = 1;
    private int sortByDesc = 0;
    private boolean pageEnable = false;

    public FolderBrowseHelper(String parent, int sort, int sortByDesc,boolean pageEnable, int
pagesize, int page){
        this.parent = parent;
        this.sort = sort;
        this.sortByDesc = sortByDesc;
        this.pagesize = pagesize;
        this.page = page;
        this.pageEnable = pageEnable;
    }

    public void setParent(String parent)
    {
        this.parent = parent;
    }

    public void setPagesize(int pagesize)
```

```
{
    this.pagesize = pagesize;
}

public void setPage(int page)
{
    this.page = page;
}

public void setStarttime(long starttime)
{
    this.starttime = starttime;
}

public void setEndtime(long endtime)
{
    this.endtime = endtime;
}

public int getSort()
{
    return sort;
}

public void setSort(int sort)
{
    this.sort = sort;
}

public int getSortByDesc()
{
    return sortByDesc;
}

public void setSortByDesc(int sortByDesc)
{
    this.sortByDesc = sortByDesc;
}

public boolean isPageEnable()
```

```
    {
        return pageEnable;
    }

    public void setPageEnable(boolean pageEnable)
    {
        this.pageEnable = pageEnable;
    }

    @Override
    protected ApiResponse doApi(ApiConfig apicfg) throws MalformedURLException,
ProtocolException,IOException, SAXException {

        FolderBrowseRequest request = new FolderBrowseRequest(
                apicfg.userid,
                apicfg.token,
                this.parent,//"system." + ApiConfig.packageDisplay + ".home.root"
                this.sort,
                this.sortByDesc
        );

        if(page>0 && pagesize>0 && pageEnable){
            request.setPageno(page);
            request.setPagesize(pagesize);
        }
        if(endtime>starttime && starttime>0){
            request.setStarttime(starttime);
            request.setEndtime(endtime);
        }
        request.setPageEnable(pageEnable);
        InfoRelayApi ir = new InfoRelayApi(apicfg.infoRelay);
        return ir.folderBrowse(request);

    }
}
```

BrowseTask.java

```
public class BrowseTask extends AsyncTask<Void, Integer, Integer> {
    private static final String TAG = "BrowseTask";
```

```
private AsyncTask<Void, Integer, Integer> task;
private ApiConfig apiCfg;
private Context ctx;

ProgressDialog _mdialog;
private List<FsInfo> fsInfos;
private FolderBrowseResponse fbRsp = null;

private int errCode = APIException.GENERAL_SUCC;
private String errMsg = "";

public BrowseTask(Context ctx, String folderId, List<FsInfo> fsInfos,
        ApiConfig apiCfg) {
    this.apiCfg = apiCfg;
    this.ctx = ctx;
    this.fsInfos = fsInfos;

    if (folderId != null && folderId.trim().length() > 0)
        apiCfg.currentFolderId = folderId;
}

@Override
protected Integer doInBackground(Void... arg0) {
    StringBuilder msg = new StringBuilder();

    // Setting progressBar initial value
    this.publishProgress(0);

    if (apiCfg.currentFolderId == null
            || apiCfg.currentFolderId.trim().length() == 0) {
        apiCfg.currentFolderId = "-2997926";//apiCfg.mySyncFolderId;
        apiCfg.folderName = apiCfg.SYNCFOLDERNAME;
    }

    Log.d(TAG, "currentFolderId :" + apiCfg.currentFolderId);

    msg.delete(0, msg.length());

    // Fetching folders & files information, that under specify folder
    FolderBrowseHelper fbHelp = new FolderBrowseHelper(
```

```
                apiCfg.currentFolderId, 1, 0, false, 0, 1);
        try {
            fbRsp = (FolderBrowseResponse) fbHelp.process(apiCfg);

            if (fbRsp.getStatus() != APIException.GENERAL_SUCC) {
                errCode = fbRsp.getStatus();
                errMsg = "";
                msg.append("Doing folderBrowse fail, status:").append(errCode);
                Log.e(TAG, msg.toString());

                return errCode;
            } else {
                // Get parent folderId;
                apiCfg.parentFolderId = String.valueOf(fbRsp.getParentFolder()
                        .getId());
                apiCfg.parentName = fbRsp.getParentFolder().getName();

                return APIException.GENERAL_SUCC;

            }
        } catch (APIException e) {
            errCode = e.status;
            errMsg = e.getMessage();
            msg.append("Doing folderBrowse error:").append(errMsg);
            Log.e(TAG, msg.toString(), e);

            return errCode;
        }
    }

    @Override
    protected void onCancelled() {
        super.onCancelled();
        if (_mdialog != null) {
            if (_mdialog != null)
                try {
                    _mdialog.dismiss();
                } catch (Exception e) {
                }
        }
```

```
}

@Override
protected void onProgressUpdate(Integer... values) {
    super.onProgressUpdate(values);
    if (values[0] == 0) {
        try {
            _mdialog = ProgressDialog.show(ctx,
                    ctx.getString(R.string.app_name), "Loading...", true,
                    true, new OnCancelListener() {
                        @Override
                        public void onCancel(DialogInterface dialog) {
                            task.cancel(true);
                            ((Activity) ctx).finish();
                        }
                    });
        } catch (Exception e) {
        }
    } else {
        if (_mdialog != null)
            try {
                _mdialog.dismiss();
            } catch (Exception e) {
            }
    }
}

@Override
protected void onPostExecute(Integer result) {
    this.publishProgress(100);
    List<FsInfo> tmpList = new LinkedList<FsInfo>();
    if (result == APIException.GENERAL_SUCC) {
        if (fbRsp.getTotalcount() > 0) {
            // Adding folder info into List
            Iterator<FolderInfo> foIter = fbRsp.getFolderList().iterator();
            while (foIter.hasNext()) {
                FolderInfo fo = foIter.next();
                tmpList.add(new FsInfo(fo));
            }
```

```
                // Adding file info into List
                Iterator<FileInfo> fiIter = fbRsp.getFileList().iterator();
                while (fiIter.hasNext()) {
                    FileInfo fi = fiIter.next();
                    tmpList.add(new FsInfo(fi));
                }
            }
        } else {
            StringBuilder msg = new StringBuilder();
            msg.append("Connecting server fail (Code:").append(errCode)
                    .append(", Message:").append(errMsg)
                    .append("), please retry later!");
            MessageDialog.show(ctx, ctx.getString(R.string.app_name),
                    msg.toString());
        }
        this.fsInfos = tmpList;
        refreshList(this.fsInfos);
    }

    protected void refreshList(List<FsInfo> fsInfos) {
    }
}
```

將上面所解析的XML放入ListView並做簡單能：

```
public class MyBrowseActivity extends ListActivity
{
    private ListActivity actBrow;

    private static final String TAG = "MyBrowseActivity";
    private ApiConfig apiCfg;
    private BrowseTask browse;
    private BrowseAdapter browAdapter;

    //Declare component variables
    private TextView  mPath;
    private ImageButton btnHome;
    private ImageButton btnBack;
```

```
    @Override
    protected void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.s_browse);

        actBrow = this;
        apiCfg  = ASUSWebStorage.apiCfg;

        //finding component's instance
        mPath   = (TextView) findViewById(R.id.mPath);
        btnHome = (ImageButton) findViewById(R.id.homeBt);
        btnBack = (ImageButton) findViewById(R.id.backBt);

        //Initial List Event Action
        initList();

        this.setListAdapter(new BrowseAdapter(actBrow, R.layout.s_browse_item,
ASUSWebStorage.processList));

        buildBrowsTask();
    }

    private void initList()
    {
        ListView lv = getListView();
        lv.setTextFilterEnabled(true);

        lv.setOnItemClickListener(new OnItemClickListener()
        {
            @Override
            public void onItemClick(AdapterView<?> parent, View view, int position, long id)
            {
                if ( browAdapter == null || browAdapter.getList() == null || position >=
browAdapter.getList().size() )
                    return;
                FsInfo fi = browAdapter.getList().get(position);
                if ( fi.entryType == EntryType.Folder )
                {
                    apiCfg.parentFolderId  = apiCfg.currentFolderId;
```

```
                apiCfg.parentName   = apiCfg.folderName;
                apiCfg.currentFolderId = fi.entryId;
                apiCfg.folderName   = fi.display;

                buildBrowsTask();
            }
            else if ( fi.entryType == EntryType.File )
            {
                // Recognize file type
                int type = recognizeFileType(fi.display);

                if ( type > 0 )
                {
                    Intent intent = new Intent();
                    intent.setClass(actBrow, FilePreviewActivity.class);

                    Bundle bundle = new Bundle();
                    bundle.putLong("fileId", Long.parseLong(fi.entryId));
                    bundle.putString("fileName", fi.display);
                    bundle.putInt("type", type);
                    intent.putExtras(bundle);

                    startActivity(intent);
                }
            }
        }
    });

    lv.setOnItemLongClickListener(new AdapterView.OnItemLongClickListener()
    {
        @Override
        public boolean onItemLongClick(AdapterView<?> parent, View view, int position, long
id)
        {
            if ( browAdapter == null || browAdapter.getList() == null )
                return false;
            FsInfo fi = browAdapter.getList().get(position);
            if ( fi.entryType == FsInfo.EntryType.Folder )
            {
                showFolderContextMenu(position);
```

```
                    return true;
                }
                else if ( fi.entryType == FsInfo.EntryType.File )
                {
                    showFileContextMenu(position);
                    return true;
                }
                else
                    return false;
            }
        });
    }

    public void backFunction(View v)
    {
        apiCfg.currentFolderId = apiCfg.parentFolderId;
        apiCfg.folderName   = apiCfg.parentName;

        buildBrowsTask();
    }

    protected void showFolderContextMenu(final int position)
    {
        final FsInfo fi = browAdapter.getList().get(position);

        String[] itemArray = getResources().getStringArray(R.array.folder_long_click);
        TypedArray itemIconArray =
getResources().obtainTypedArray(R.array.folder_long_click_icon);
        List<IconContextMenuItem> itemList = new ArrayList<IconContextMenuItem>();
        for ( int i=0 ; i < itemArray.length ; i++ )
        {
            IconContextMenuItem _item;
            _item = new
IconContextMenuItem(getResources(),itemArray[i],itemIconArray.getResourceId(i, -1),i);
            itemList.add(_item);
        }
        IconMenuAdapter tmpAdapter = new IconMenuAdapter(actBrow, R.layout.s_context_menu_item,
itemList);
        AlertDialog dialog = new AlertDialog.Builder(actBrow)
        .setTitle(fi.display)
```

```
        .setAdapter(tmpAdapter, new DialogInterface.OnClickListener()
        {
               public void onClick(DialogInterface dialog, int which)
               {
                 switch ( which )
                 {
                      case 0 :// share
                               new ShareEntryTask(actBrow, apiCfg).execute(fi);
                               break;
                           case 1 :// Rename
                               showRenameDialog(actBrow, apiCfg, fi, position);
                               break;
                           case 2 :// delete
                               showRemoveDialog(actBrow, apiCfg, fi);
                               break;
                 }
               }
        })
       .create();
        dialog.show();
    }

    protected void showFileContextMenu(final int position)
    {
        final FsInfo fi = browAdapter.getList().get(position);
        Log.d(TAG, "Now Position -->" + position);

        int menuid = R.array.file_marked_long_click;
        int menuIconid = R.array.file_marked_long_click_icon;

        String[] itemArray = getResources().getStringArray(menuid);
        TypedArray itemIconArray = getResources().obtainTypedArray(menuIconid);
        List<IconContextMenuItem> itemList = new ArrayList<IconContextMenuItem>();
        for (int i = 0 ; i < itemArray.length ; i++ )
        {
            IconContextMenuItem _item;
            _item = new
IconContextMenuItem(getResources(),itemArray[i],itemIconArray.getResourceId(i, -1),i);
            itemList.add(_item);
        }
```

```
        IconMenuAdapter tmpAdapter = new IconMenuAdapter(actBrow, R.layout.s_context_menu_item,
itemList);
        AlertDialog dialog = new AlertDialog.Builder(actBrow)
        .setTitle(fi.display)
        .setAdapter(tmpAdapter, new DialogInterface.OnClickListener()
        {
              public void onClick(DialogInterface dialog, int which)
              {
                switch ( which )
                {
                     case 0 :// download
                          FileDownloadTask fdTask = new FileDownloadTask(actBrow,
Long.valueOf(fi.entryId), fi.display, apiCfg)
                               {

                                        @Override
                                        protected void onSuccess()
                                        {
                                             super.onSuccess();
                                        }

                                        @Override
                                        protected void onFail(int errCode, String errMessage)
                                        {
                                             super.onFail(errCode, errMessage);
                                        }

                               };
                          fdTask.execute((Void)null);

                          break;
                     case 1 :// share
                          new ShareEntryTask(actBrow, apiCfg).execute(fi);
                          break;
                     case 2 :// rename
                          showRenameDialog(actBrow, apiCfg, fi, position);
                          break;
                     case 3 :// delete
                          showRemoveDialog(actBrow, apiCfg, fi);
                          break;
```

```
                }
            }
        })
        .create();
         dialog.show();
    }

    private void showRenameDialog(final Context ctx, final ApiConfig apicfg, final FsInfo fInfo,
final int position)
    {
         final int dotPos = fInfo.display.lastIndexOf(".");
         String oldName;
         if ( dotPos > 0 )
         {
              oldName = fInfo.display.substring(0, dotPos);
         }
         else
         {
              oldName = fInfo.display;
         }

         AlertDialog.Builder builder = new AlertDialog.Builder(ctx);
         builder.setTitle("Rename");

         final EditText editName = new EditText(ctx);

         editName.setLayoutParams(new
LinearLayout.LayoutParams(LinearLayout.LayoutParams.FILL_PARENT,
LinearLayout.LayoutParams.WRAP_CONTENT));
         editName.setText(oldName);
         builder.setNegativeButton("Cancel", new DialogInterface.OnClickListener()
         {
              @Override
              public void onClick(DialogInterface dialog, int which)
              {
              }

         });
         builder.setPositiveButton("OK", new DialogInterface.OnClickListener()
         {
```

```
            @Override
            public void onClick(DialogInterface dialog, int which)
            {
                String fileExt;
                if ( dotPos > -1 )
                {
                    fileExt = fInfo.display.substring(dotPos).trim();
                }
                else
                {
                    fileExt = "";
                }
                final String newName = editName.getEditableText().toString().trim()+fileExt;

                if ( !chkName(newName) )
                {
                    AlertDialog.Builder builder = new AlertDialog.Builder(ctx);
                    builder.setTitle("Rename");
                    builder.setNeutralButton("OK", null);
                    builder.setMessage("Please enter correct name. The\\ /:*?\"&lt;&gt;| symbols
are not accepted.");
                    builder.create().show();
                    return;
                }
                final FsInfo fs = new FsInfo();
                fs.display    = newName;
                fs.entryType = fInfo.entryType;
                fs.entryId    = fInfo.entryId;
                fs.parent     = fInfo.parent;
                fs.attribute = fInfo.attribute;
                fs.size       = fInfo.size;
                fs.icon       = fInfo.icon;

                new EntryRenameTask(ctx, apicfg)
                {

                    @Override
                    protected void onSuccess()
                    {
```

```
                    //Removed original entity
                    browAdapter.remove(fInfo);

                    //Insert the entity with new name at original position
                    browAdapter.insert(fs, position);
                    actBrow.setListAdapter(browAdapter);

                    super.onSuccess();
                }

                @Override
                protected void onFail(int errCode, String errMsg)
                {
                    super.onFail(errCode, errMsg);
                }

            }
            .execute(fs);
        }

    });
    builder.setView(editName);
    builder.create().show();
}

private void showRemoveDialog(final Context ctx, final ApiConfig apicfg, final FsInfo fInfo){
    AlertDialog.Builder builder = new AlertDialog.Builder(ctx);
    builder.setTitle("Delete");
    builder.setMessage(String.format("Are you sure to delete %s?", fInfo.display));

    builder.setNegativeButton("Cancel", new DialogInterface.OnClickListener()
    {
        @Override
        public void onClick(DialogInterface dialog, int which)
        {
        }

    });
    builder.setPositiveButton("OK", new DialogInterface.OnClickListener()
```

```
        {

            @Override
            public void onClick(DialogInterface dialog, int which)
            {
                new EntryRemoveTask(ctx, apicfg)
                {

                    @Override
                    protected void onSuccess()
                    {
                        super.onSuccess();

                        //Removing entity from List
                        browAdapter.remove(fInfo);
                        actBrow.setListAdapter(browAdapter);
                    }

                    @Override
                    protected void onFail(int errCode, String errMsg)
                    {
                        super.onFail(errCode, errMsg);
                    }

                }
                .execute(fInfo);
            }
        });
        builder.create().show();
    }

    private boolean chkName(String name)
    {
        return ( !( name.indexOf("\\") > -1 || name.indexOf("/") > -1 || name.indexOf(":") > -1
|| name.indexOf("*") > -1 || name.indexOf("?") > -1
                || name.indexOf(">") > -1 || name.indexOf("|") > -1 || name.indexOf("<") > -1 ||
name.indexOf("\"") > -1
                )
                && name.length() > 0 && name.length() < 250 );
    }
```

```
    private void fileUploadFunction(String folderId){
        Intent intent = new Intent();
        intent.setClass(actBrow, UploadActivity.class);

        Bundle bundle = new Bundle();
        bundle.putString("uploadFolder", folderId);
        intent.putExtras(bundle);

        startActivityForResult(intent, 0);
//      startActivity(intent);
    }

    private void folderCreateFunction(String folderId)
    {
        showCreateFolderDialog(actBrow, this.apiCfg, this.apiCfg.currentFolderId);
    }

    private void showCreateFolderDialog(final Context ctx, final ApiConfig apicfg, String parent)
    {
        AlertDialog.Builder builder = new AlertDialog.Builder(ctx);
        builder.setTitle("Creating new folder");
        final EditText editName = new EditText(ctx);
        editName.setLayoutParams(new
LinearLayout.LayoutParams(LinearLayout.LayoutParams.FILL_PARENT,
LinearLayout.LayoutParams.WRAP_CONTENT));
        editName.setHint("Please input new folder name");
        builder.setNegativeButton("Cancel", new DialogInterface.OnClickListener()
        {
            @Override
            public void onClick(DialogInterface dialog, int which)
            {
//              task.cancel(true);
            }

        });
        builder.setPositiveButton("OK", new DialogInterface.OnClickListener()
        {

            @Override
```

```
            public void onClick(DialogInterface dialog, int which)
            {

                String newName = editName.getEditableText().toString().trim();

            if ( !chkName(newName) )
                {
                    AlertDialog.Builder builder = new AlertDialog.Builder(ctx);
                    builder.setTitle("Creating new folder");
                    builder.setNeutralButton("OK", null);
                    builder.setMessage("Please enter correct name. The\\ /:*?\"&lt;&gt;| symbols
are not accepted.");
                    builder.create().show();
                    return;
                }

                FolderCreateTask fcTask = new FolderCreateTask(actBrow, apiCfg,
apiCfg.currentFolderId)
                {

                    @Override
                    protected void onSuccess(long folderId)
                    {
                        buildBrowsTask();
                    }

                    @Override
                    protected void onFail(int errCode, String errMsg)
                    {
                        StringBuilder msg = new StringBuilder();

                        msg.append("Create folder error:(").append(errCode)
                           .append(")").append(errMsg);

                        MessageDialog.show(actBrow, "Creating new folder", msg.toString());
                    }

                };
                fcTask.execute(newName);
            }
```

```
        });
        builder.setView(editName);
        builder.create().show();
    }

    private void buildBrowsTask()
    {
        if ( apiCfg.mySyncFolderId.equals(apiCfg.currentFolderId) && btnHome.getVisibility() ==
View.GONE )
        {
            btnBack.setVisibility(View.GONE);
            btnHome.setVisibility(View.VISIBLE);
        }
       else if ( !apiCfg.mySyncFolderId.equals(apiCfg.currentFolderId) && btnBack.getVisibility()
== View.GONE )
        {
            btnHome.setVisibility(View.GONE);
            btnBack.setVisibility(View.VISIBLE);
        }

        if ( browse != null && !browse.isCancelled() )
            browse.cancel(true);

        browse =  new BrowseTask(actBrow, null, ASUSWebStorage.processList, apiCfg)
        {
            @Override
            protected void onProgressUpdate(Integer... values)
            {
                if(values[0]==0)
                {
                    mPath.setText(apiCfg.folderName);
                    actBrow.setListAdapter(new BrowseAdapter(actBrow, R.layout.s_browse_item,
ASUSWebStorage.processList));
                }
            }

            @Override
            protected void refreshList(List<FsInfo> fsInfos)
            {
```

```
            browAdapter = new BrowseAdapter(actBrow, R.layout.s_browse_item, fsInfos);
            actBrow.setListAdapter(browAdapter);
        }
    };
    browse.execute((Void)null);
}

protected int recognizeFileType(String name)
{
    String fileType = FsInfo.parseFileType(name);
    int type = 0;

    if ( fileType.equals("image/*") )
        type = 1;
    else if ( fileType.equals("video/*") )
        type = 3;
    else if ( fileType.equals("application/zip") )
        type = 0;
    else
    {
        type = 2;
    }
    return type;
}

@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data)
{
    if ( resultCode == RESULT_OK )
    {
        Toast.makeText(actBrow, "File Uploaded Successful!", Toast.LENGTH_LONG);

        //Refresh list
        buildBrowsTask();
    }
    else
    {
        int    errCode = APIException.GENERAL_ERR;
        String errMsg  = "";
```

```
        if ( data != null )
        {
            Bundle bundle  = data.getExtras();
            if ( bundle != null )
            {
                errCode = bundle.getInt("errCode", APIException.GENERAL_ERR);
                errMsg  = bundle.getString("errMsg");
            }
        }

        StringBuilder msg = new StringBuilder();
        msg.append("File uploading fail, status:").append(errCode)
           .append(", message:").append(errMsg);

        MessageDialog.show(actBrow, "File Uploading", msg.toString());
    }
}

@Override
public boolean onCreateOptionsMenu(Menu menu)
{
    MenuInflater inf = getMenuInflater();
    inf.inflate(R.menu.folder_menu, menu);
    return super.onCreateOptionsMenu(menu);
}

@Override
public boolean onOptionsItemSelected(MenuItem item)
{
    boolean bRtn = super.onOptionsItemSelected(item);
    switch ( item.getItemId() )
    {
        case R.id.mmuAbout:
            ActivityMenu.openOptionsDialog(actBrow);
            break;
        case R.id.mmuFCreate:
            folderCreateFunction(apiCfg.currentFolderId);
            break;
        case R.id.mmuUpload:
            fileUploadFunction(apiCfg.currentFolderId);
```

```
                break;
            case R.id.mmuExit:
                finish();
        }
        return bRtn;
    }
}
```

同時需要做一個檔案資料的容器方便後續讀取：

FsInfo.java

```
public class FsInfo
{
    public static enum EntryType
    {
        File(0),
        Folder(1),
    NUll(5),
    Process(7);

        EntryType(int keyId)
        {
            this.key_id = keyId;
        }
       private final int key_id;

       public int getInt() {
           return key_id;
       }

       public String getString() {
           return String.valueOf(key_id);
       }
    };

    private final DateFormat createTimeFormat = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");
//  private final DateFormat timeFormat = new SimpleDateFormat("yyyyMMddHHmmss");
    private final DateFormat attTimeFormat = new SimpleDateFormat("yyyy/MM/dd HH:mm:ss");
```

```
    public EntryType entryType = EntryType.File;
    public String     entryId;
    public String     display;
    public String     parent;
    public Attribute attribute;
    public Long   size = null;
    public int        icon = 0;

    public FsInfo(){}

    public FsInfo(EntryType entryType, String display)
    {
        this.entryType = entryType;
        this.display = display;
    }

    public FsInfo(FolderInfo fi)
    {
        this.entryId   = fi.getId();
        this.display   = fi.getDisplay();
        this.attribute = fi.getAttribute();

        if ( attribute == null ||
            ((attribute.getLastwritetime() == null ||
attribute.getLastwritetime().trim().length() <= 0 ) &&
             (attribute.getCreationtime() == null || attribute.getCreationtime().trim().length()
<= 0)))
        {
            attribute = new Attribute();
            if ( fi.getCreatedtime() != null && fi.getCreatedtime().trim().length() > 0 )
            {
                try
                {
                    Date d = createTimeFormat.parse(fi.getCreatedtime().trim());
                    attribute.setCreationtime(String.valueOf((d.getTime()/1000)));
                }
                catch ( ParseException e ){}
            }
        }
```

```
        this.entryType = EntryType.Folder;

        this.icon = getBrowseRawIcon(this.display, this.entryType);
        parseAttribute();
    }

    public FsInfo(FileInfo fi)
    {
        this.entryId   = fi.getId();
        this.display   = fi.getDisplay();
        this.attribute = fi.getAttribute();

        if ( attribute == null ||
            ((attribute.getLastwritetime() == null ||
attribute.getLastwritetime().trim().length() <=0 ) &&
             (attribute.getCreationtime() == null || attribute.getCreationtime().trim().length()
<=0 )))
        {
            attribute = new Attribute();
            if( fi.getCreatedtime() != null && fi.getCreatedtime().trim().length() > 0 )
            {
                try
                {
                    long ctime = Long.parseLong(fi.getCreatedtime());
                    attribute.setCreationtime(String.valueOf((long)(ctime/1000)));
                }
                catch ( NumberFormatException e )
                {
                    try
                    {
                        Date d = createTimeFormat.parse(fi.getCreatedtime().trim());
                        attribute.setCreationtime(String.valueOf((long)(d.getTime()/1000)));
                    }
                    catch ( ParseException pe )
                    {
                        Log.e("FsInfo", e.getMessage(), pe);
                    }
                }
                if ( attribute.getCreationtime() != null )
                {
```

```
                attribute.setLastaccesstime(attribute.getCreationtime());
                attribute.setLastwritetime(attribute.getCreationtime());
            }
        }
    }

    this.entryType = EntryType.File;

    this.size = fi.getSize();
    this.icon = getBrowseRawIcon(this.display, this.entryType);
    parseAttribute();
}

private final long KB = 1024L;
private final long MB = 1024 * 1024L;
private final long GB = 1024 * 1024 * 1024L;
private final long TB = 1024 * 1024 * 1024 * 1024L;
public String getSizeDisp()
{
    if (size > TB) return String.valueOf(Math.round(size/TB)) + " TB";
    else if(size > GB) return String.valueOf(Math.round(size/GB)) + " GB";
    else if(size > MB) return String.valueOf(Math.round(size/MB)) + " MB";
    else if(size > KB) return String.valueOf(Math.round(size/KB)) + " KB";
    else if(size >= 0) return String.valueOf(size) + " B";
    else return "";
}

public int getBrowseRawIcon(String disp, EntryType entryType)
{
    int rtn = 0;
    if ( entryType==EntryType.Folder )
    {
        rtn = R.drawable.icon_list_folder;
    }
    else if ( entryType==EntryType.File )
    {
        String tp = FsInfo.parseFileType(disp);
        if ( tp.startsWith("audio/") )
            rtn = R.drawable.icon_list_music;
        else if ( tp.startsWith("video/") )
```

```
                rtn = R.drawable.icon_list_video;
            else if ( tp.startsWith("image/") )
                rtn = R.drawable.icon_list_photo;
            else if ( "application/pdf".equals(tp) )
                rtn = R.drawable.icon_list_pdf;
            else if ( "application/msword".equals(tp) )
                rtn = R.drawable.icon_list_doc;
            else if ( "application/vnd.ms-excel".equals(tp) )
                rtn = R.drawable.icon_list_excel;
            else if ( "application/vnd.ms-powerpoint".equals(tp) )
                rtn = R.drawable.icon_list_ppt;
            else if ( "text/*".equals(tp) )
                rtn = R.drawable.icon_list_txt;
            else if ( "application/zip".equals(tp) )
                rtn = R.drawable.icon_list_zip;
            else if ( "application/epub+zip".equals(tp) )
                rtn = R.drawable.icon_list_epub;
            else if ( "code".equals(tp) )
                rtn = R.drawable.icon_list_code;
            else
                rtn = R.drawable.icon_list_other;
        }
        return rtn;
    }

    public static String parseFileType(String end)
    {
        end = end.substring(end.lastIndexOf(".") + 1, end.length()).toLowerCase();

        if ( end.equals("mp3") )
        {
            return "audio/mp3";
        }
        else if ( end.equals("m4a") || end.equals("mid") || end.equals("xmf") || end.equals("ogg")
|| end.equals("wav") || end.equals("amr") )
        {
            return "audio/*";
        }
        else if ( end.equals("avi") || end.equals("mp4") || end.equals("mpeg") || end.equals("mpg")
|| end.equals("m4v") || end.equals("mov")
```

```
                || end.equals("mkv") || end.equals("vob") || end.equals("vcd") ||
end.equals("svcd") || end.equals("rm") || end.equals("rmvb")
                || end.equals("divx") || end.equals("wmv") || end.equals("3gp") ||
end.equals("3gpp") || end.equals("flv") )
        {
            return "video/*";
        }
        else if ( end.equals("jpg") || end.equals("gif") || end.equals("png") || end.equals("jpeg")
|| end.equals("bmp") )
        {
            return "image/*";
        }
        else if ( end.equals("pdf") )
        {
            return "application/pdf";
        }
        else if ( end.equals("doc") || end.equals("docx") || end.equals("rtf") )
        {
            return "application/msword";
        }
        else if ( end.equals("xls") || end.equals("xlsx") )
        {
            return "application/vnd.ms-excel";
        }
        else if ( end.equals("ppt") || end.equals("pptx") )
        {
            return "application/vnd.ms-powerpoint";
        }
        else if ( end.equals("txt") || end.equals("odt") || end.equals("ods") ||
end.equals("odp") )
        {
            return "text/*";
        }
        else if ( end.equals("zip") || end.equals("rar") )
        {
            return "application/zip";
        }
        else if ( end.equals("epub") )
        {
            return "application/epub+zip";
```

```
        }
        else if ( end.equals("htm") || end.equals("html") || end.equals("xml") || end.equals("js")
|| end.equals("css") || end.equals("java")
                || end.equals("aidl") || end.equals("vb") || end.equals("c") || end.equals("h") )
        {
            return "code";
        }
        else
        {
            return "*/*";
        }
    }


    private void parseAttribute()
    {
        long _nowDateTime = new Date().getTime();
        if ( attribute != null )
        {
            if ( attribute.getCreationtime() != null &&
attribute.getCreationtime().trim().length() > 0 )
            {
                try
                {
                    Date d = attTimeFormat.parse(attribute.getCreationtime().trim());
                    attribute.setCreationtime(String.valueOf((d.getTime()/1000)));
                }
                catch ( ParseException e1 )
                {
                    long _d = Long.parseLong(attribute.getCreationtime().trim())*1000;
                    if( _d > _nowDateTime*10 )
                    {
                        attribute.setCreationtime(String.valueOf((new
Date(_d/1000000).getTime())));
                    }
                }
            }
            if ( attribute.getLastaccesstime() != null &&
attribute.getLastaccesstime().trim().length() > 0 )
            {
```

```
                try
                {
                    Date d = attTimeFormat.parse(attribute.getLastaccesstime().trim());
                    attribute.setLastaccesstime(String.valueOf((d.getTime()/1000000)));
                }
                catch ( ParseException e1 )
                {
                    long _d = Long.parseLong(attribute.getLastaccesstime().trim())*1000;
                    if ( _d > _nowDateTime*10 )
                    {
                        attribute.setLastaccesstime(String.valueOf((new
Date(_d/1000000).getTime())));
                    }
                }
            }
            if ( attribute.getLastwritetime() != null &&
attribute.getLastwritetime().trim().length() > 0 )
            {
                try
                {
                    Date d = attTimeFormat.parse(attribute.getLastwritetime().trim());
                    attribute.setLastwritetime(String.valueOf((d.getTime()/1000000)));
                }
                catch ( ParseException e1 )
                {
                    long _d = Long.parseLong(attribute.getLastwritetime().trim())*1000;
                    if ( _d > _nowDateTime*10 )
                    {
                        attribute.setLastwritetime(String.valueOf((new
Date(_d/1000000).getTime())));
                    }
                }
            }
        }
    }
}
```

此時就可以呈現相關資料訊息。