PayNowDoubleCheck_G

V1.1 (2022/09/20)

正式：https://www.paynow.com.tw/service/PayNowAPI_JS.aspx

測試：https://test.paynow.com.tw/service/ PayNowAPI_JS.aspx

編碼格式：所有參數傳遞時，請以 URL Encode 編碼，所有網頁字集為 UTF-8

操作流程：

交易狀態查詢 API 的使用流程

1.商家拋送 OP=GPZ 及 JStr 取得 PassCode 及 CheckNum; JStr 的內容為 json 格式並以 aes256 加密做 urlencoden 送出

2.paynow 以 aes256 解密取得 JStr 的內容，回吐 PassCode 及 CheckNum；組成 Json 格式後以 aes256 加密並做 urlencode 後回傳

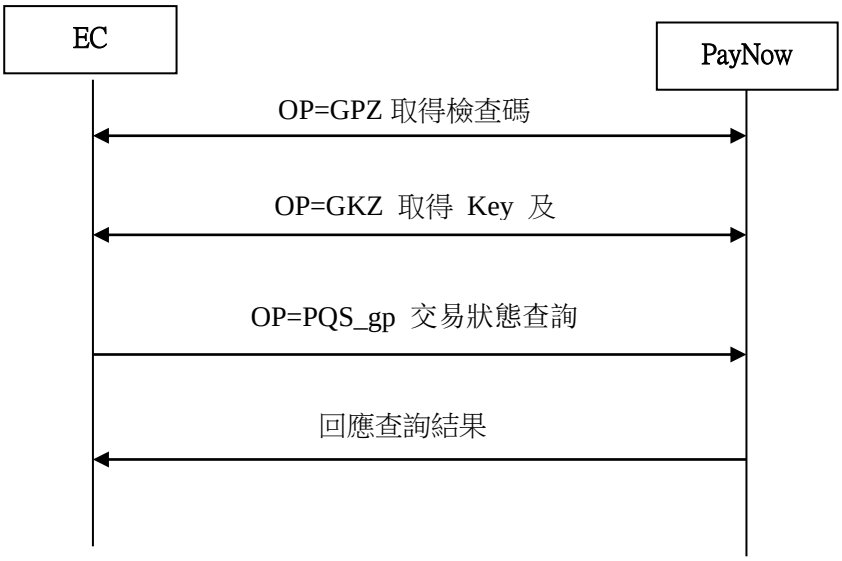
3.商家接收回應字串做 urldecode 後以 aes256 解密，解析 json 格式取得 CheckNum 並驗證 PassCode

4.商家以步驟 3 取得的 CheckNum 呼叫 OP=GKZ 的 API，JStr 的內容為 json 格式並以 aes256 加密做 urlencoden 送出

5.paynow 以 aes256 解密取得 JStr 的內容，回吐 PassCode、EncryptionKey 及 EncryptionIV；組成 Json 格式後以 aes256 加密並做 urlencode 後回傳

6.商家呼叫 OP=PQS_gp 的 API，以文件描述需要的參數組成 JsonString 後，利用步驟 5 取得的 EncryptionKey 及 EncryptionIV 將 JsonString 以 AES256 加密後，拆分成 JStr1 及 JStr2 並做 urlencode，拋送文件描述的參數共 6 個

7.paynow 以商家帳號、TimeStr 及 CheckNum 取得解密用的 AES256 密碼解密後，回傳查詢的交易狀態字串



1.檢核碼系統串接

傳遞方式：Http POST method UrlEncode

參數名稱	描述	型態	必須	備註
傳送 (Object)				
OP	判斷代號	string	Y	GPZ = 取得檢查碼 GKZ = 取得 Key 及 IV
JStr	檢核驗證資訊	string	Y	將下列參數組成 Json 格式字串，以 AES256 加密傳遞 (註 1) GPZ 取得隨機檢查碼： mem_cid：商家帳號(統編/身分證) PassCode：交易驗證碼 PassCode 產生方式請見(註 3) TimeStr：時間戳 時間戳產生方式請見(註 4) GKZ 取得 Key 及 IV： 包含以上參數以及 CheckNum：檢核碼 (8 碼)
回應 (Object)				
	WS 回應	string		Json 字串以 AES256 加密回傳，請以 AES256 解密 (註 2)
	Json 解密後 Object 內容			
	GPZ：			
	mem_cid	string		商家帳號
	PassCode	string		回覆驗證碼 (註 3)
	TimeStr	string		時間戳
	CheckNum	string		隨機檢核碼 (8 碼)
	GKZ：			
	PassCode	string		回覆驗證碼 (註 3)
	EncryptionKey	string		加密 Key
	EncryptionIV	string		加密 IV

檢核碼使用的 AES256 加密(註 1 及註 2)key 及 IV 固定為下：

Key：paynowencryptpaynowcomtw28229955
IV：encrypt282299550

2.交易狀態查詢

傳遞方式：Http POST method UrlEncode

參數名稱	描述	型態	必須	備註
傳送				
OP	服務代號	**string**	**Y**	**OP = QPS_gp**
JStr1/JStr2	**Json 字串**	**string**	**Y**	**請將下列參數組成 Json 格式字串** **Mem_cid：商家帳號** **PassCode：交易驗證碼** **請以下列組合成字串後使用 SHA-1 雜湊函數加密並轉大寫** **2822Mem_cidOrderNo 商家交易密碼 9955** **OrderNO：商家自訂編號** 將上列參數組成 Json 字串後，以檢核碼系統發出的 Key 及 IV(註 3)做 AES256 加密(註 1)後 ,將加密後字串對分拆解成 JStr1 及 JStr2 兩個參數，再做 UrlEncode 上傳
mem_cid	商家帳號	string	Y	商家帳號(統編/身分證)
TimeStr	時間戳	string	Y	加密傳送字串時使用的 TimeStr
CheckNum	隨機檢核碼	string	Y	加密傳送字串取得隨機檢查碼時回傳的 CheckNum
回應				

	回應字串	**string**		※urlencode 回覆內容※ 交易成功：1 開頭並以逗號分隔，逗號前為成功筆數，逗號後為 19 碼 paynow 訂單編號及(卡號後四碼或虛擬帳號)以底線分隔，若是信用卡交易最後一組底線分隔為分期期數(若非分期交易則預設為 1)。 ex： 1,5000001111146998321_3211_1 Or 1,5000001111146998321_95533725300857 交易失敗(有 paynow 訂單，授權失敗或者未完成)：2 開頭並以逗號分隔，逗號後為 19 碼 paynow 訂單編號及卡號後四碼或虛擬帳號及錯誤代碼(錯誤碼可能為空值)，以底線分隔單號、卡號及錯誤代碼；單號底線卡號底線錯誤碼為一組，若有多組將以逗號分隔 ex： 2,5000001111146998321_3211_05 Or 2,5000001111146998321_95533725300857_ Or 2.5000001111146998321_95533725300857_,5000001111146998531_95533725300866_,5000001111146999102_95533725300871_ 退貨交易：3 開頭以逗號分隔退貨狀態 (0:買家申請退貨 1:買賣家確認 2:銀行退款 3.賣家申請) ex:3,1 or 3,0 交易失敗(無交易訂單)：回覆數字 4 ex: 4 **(定單待確認，使用者可能未送出授權，無 paynow 交易訂單，無法確認狀態)** 其它(重覆交易)： 02 成功交易兩筆 03 成功交易三筆 以此類推 ；逗號前為成功筆數，逗號後為 19 碼 paynow 訂單編號及(卡號後四碼或虛擬帳號)以底線分隔，若是信用卡交易最後一組底線分隔為分期期數(若非分期交易則預設為 1)，之後以逗號分隔每筆訂單。 ex： **兩筆重覆交易** 02,5000001111146998321_3211_1,5000001111146699323_4322_3 Or

				02,5000001111146998321_95533725300857,500000111114669 9323_4322_1

註 1、AES256 加密：

說明：下列範例使用 C#.net 編寫，僅供參考使用
此系統傳遞之AES256加解密，以檢核碼系統取得的Key值(**EncryptionKey**)及IV值(**EncryptionIV**)做加解密
Content：需要加密的內容
Keu：**EncryptionKey**
IV：**EncryptionIV**

Ex：C# code

```
public string AES256_Encrypt(string Content, string key, string IV)
        {
            byte[] ByteString = Encoding.UTF8.GetBytes(Content);
            byte[] ByteKeyString = Encoding.UTF8.GetBytes(key);
            byte[] ByteIVString = Encoding.UTF8.GetBytes(IV);
            RijndaelManaged RDEL = new RijndaelManaged();
            RDEL.Key = ByteKeyString;
            RDEL.IV = ByteIVString;
            RDEL.Mode = CipherMode.CBC;
            RDEL.Padding = PaddingMode.Zeros;
            ICryptoTransform cTransform = RDEL.CreateEncryptor();

            byte[] ResultByte = null;
            ResultByte = cTransform.TransformFinalBlock(ByteString, 0, ByteString.Length);
            string ReturnStr = Convert.ToBase64String(ResultByte, 0, ResultByte.Length);
            return ReturnStr;
        }
```

註 2、AES256 解密：

說明：下列範例使用 C#.net 編寫，僅供參考使用
此系統傳遞之AES256加解密，以檢核碼系統取得的Key值(**EncryptionKey**)及IV值(**EncryptionIV**)做加解密
Content：需要加密的內容
Keu：**EncryptionKey**
IV：**EncryptionIV**

Ex：C# code

```
public string AES256_Decrypt(string Content, string key, string IV)
        {
            byte[] ByteString = Convert.FromBase64String(Content);
            byte[] ByteKeyString = Encoding.UTF8.GetBytes(key);
            byte[] ByteIVString = Encoding.UTF8.GetBytes(IV);
            RijndaelManaged RDEL = new RijndaelManaged();
            RDEL.Key = ByteKeyString;
            RDEL.IV = ByteIVString;
            RDEL.Mode = CipherMode.CBC;
            RDEL.Padding = PaddingMode.Zeros;
            ICryptoTransform cTransform = RDEL.CreateDecryptor();

            byte[] ResultByte = null;
            ResultByte = cTransform.TransformFinalBlock(ByteString, 0, ByteString.Length);
            string ReturnStr = Encoding.UTF8.GetString(ResultByte);
            return ReturnStr;
```

註 3、 PassCode 產生範例：

GPZ 取得隨機檢查碼

以 mem_cid 及 GPZ規則加權檢核碼 (註5) 串接後做 SHA256雜湊加密字串(註6)後轉大寫帶入

回覆時的PassCode規則：

以 mem_cid 及 **GKZ規則**加權檢核碼 (註5) 串接後，以隨機檢查碼(CheckNum)為加密鑰 做 SHA256雜湊加密字串(註7)後轉大寫帶入

GKZ 取得Key及IV

以 mem_cid 及 GKZ規則加權檢核碼 (註5) 串接後，以隨機檢查碼(CheckNum)為加密鑰 做 SHA256雜湊加密字串(註7)後轉大寫帶入

回覆時的PassCode規則：

以 mem_cid 及 **GPZ規則**加權檢核碼 (註5) 串接後做 SHA256雜湊加密字串(註6)後轉大寫帶入

註 4、 TimeStr 時間戳格式：

Ex：以日期 2019-11-24 00:50:18 產生 10 碼數字

第 1 至 4 碼：太陽日，第 1 碼取西元年最後一位 2019 = 9

2 至 4 碼為一整年的起算天數，11/24 為今年的 328 天，所以 2 至 4 碼 = 328

第 5 至 10 碼：5-6 為小時數字 = 00；7-8 為分鐘數字 = 50；9-10 為秒分數字 = 18

1 至 10 碼組成時間戳 TimeStr = 9328005018

註 5、加權檢核碼產生規則：

分別以加權權數(23 碼)與加權基數(23 碼)各個數字相乘之後取個位數字，再取得**以 10 減去個數字相加後除以 10 得餘數後的數字**獲得加權權數檢查碼 1 碼 (10-餘數)，若為 10 則取 0；之後將加權權數前 15 碼字串與加權權數檢查碼 1 碼串接取得 16 碼數字；GP 與 GK 模式取得規則如下：

GPZ：取得隨機檢查碼

EX：商家帳號(mem_cid) = 028229955，TimeStr = 9328005018

加權基數(23 碼)：93193193193193193193193

加權權數(23 碼) :商家帳號 9 碼 ,不足左補 0 ;取商家帳號**前 5 碼(5 碼)** = 02822 + TimeStr(10 碼) = 9328005018

+ TimeStr 前 4 碼(4 碼) = 9328 + 商家帳號**後 4 碼**(4 碼) = 9955，加權權數(23 碼) = 02822932800501893289955

將權數與基數相乘取其個位數字相加

0	2	8	2	2	9	3	2	8	0	0	5	0	1	8	9	3	2	8	9	9	5	5
X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
9	3	1	9	3	1	9	3	1	9	3	1	9	3	1	9	3	1	9	3	1	9	3
=	=	=	=	=	=	=	=	=	=	=	=	=	=	=	=	=	=	=	=	=	=	=
0	6	8	18	6	9	27	6	8	0	0	5	0	3	8	81	9	2	72	27	9	45	15

取個數分別得到並相加：

0 + 6 + 8 + 8 + 6 + 9 + 7 + 6 + 8 +0 + 0 + 5 + 0 + 3 + 8 + 1 + 9 + 2 + 2 + 7 + 9 + 5 + 5 = 114

114 / 10 得餘數 4 ：加權檢查碼 取 10 – 4 = 6 加權檢核碼 = 0282293280050186

※組成回覆 PassCode 時的加權檢核碼會以 GKZ 的規則替換

GKZ：取得 Key 及 IV

EX：商家帳號(mem_cid) = 028229955，TimeStr = 9328005018

加權基數(23 碼)：93193193193193193193193

加權權數(23 碼) :商家帳號 9 碼，不足左補 0 ;取商家帳號**後 5 碼**(5 碼) = 29955 + TimeStr(10 碼) = 9328005018 + TimeStr 前 4 碼(4 碼) = 9328 + 商家帳號**前 4 碼(4 碼)** = 0282，加權權數(23 碼) = 29955932800501893280282

將權數與基數相乘取其個位數字相加

2	9	9	5	5	9	3	2	8	0	0	5	0	1	8	9	3	2	8	0	2	8	2
X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
9	3	1	9	3	1	9	3	1	9	3	1	9	3	1	9	3	1	9	3	1	9	3
=	=	=	=	=	=	=	=	=	=	=	=	=	=	=	=	=	=	=	=	=	=	=
18	27	9	45	15	9	27	6	8	0	0	5	0	3	8	81	9	2	72	0	2	72	6

取個數分別得到並相加：

8 + 7 + 9 + 5 + 5 + 9 + 7 + 6 + 8 +0 + 0 + 5 + 0 + 3 + 8 + 1 + 9 + 2 + 2 + 0 + 2 + 2 + 6 = 104

104 / 10 得餘數 4 ：加權檢查碼 取 10 – 4 = 6

加權檢核碼 = 29955932800501866

※組成回覆 PassCode 時的加權檢核碼會以 GPZ 的規則替換

註 6、SHA256 雜湊加密字串：

Ex：C# code

```
public string SHA256_Encrypt(string Content)
{
System.Security.Cryptography.SHA256CryptoServiceProvider sha256 = new System.Security.Cryptography.SHA256CryptoServiceProvider();
                byte[] ByteString = System.Text.Encoding.ASCII.GetBytes(Content);
                ByteString = sha256.ComputeHash(ByteString);
                string ReturnStr = null;
                foreach (byte bt in ByteString)
                      ReturnStr += bt.ToString("x2");
                return ReturnStr.ToUpper();
          }
```

註 7、SHA256 雜湊加密字串：

Ex：C# code

```
public string SHA256_HMACSHA256(string Content, string Key)
{
                byte[] ByteString = System.Text.Encoding.ASCII.GetBytes(Content);
                byte[] ByteKeyString = Encoding.ASCII.GetBytes(Key);
                byte[] HasBytes;
                HMACSHA256 sha256 = new HMACSHA256(ByteKeyString);
                HasBytes = sha256.ComputeHash(ByteString);
                string ReturnStr = BitConverter.ToString(HasBytes).Replace("-", "").ToUpper();
                return ReturnStr;
}
```