SERVICIOS DIGITALES POPULAR

DOCUMENTO TÉCNICO INTEGRACIÓN VÍA API

con Apple Pay

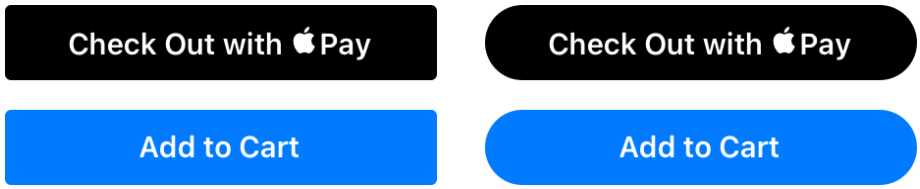
ÍNDICE

Integración con Apple Pay

A continuación, se presentan los métodos de integración disponibles para Apple Pay por Integración vía API de AZUL. Estos escenarios de integración difieren principalmente en el manejo de la sesión de pago y de responsabilidad de la desenciptación del token de pago que genera Apple Pay

1. A través de AZUL como Platform Integrator (Solo para web, no soporta iOS)
2. Mechant Identifier (Independiente para el comercio)
3. Gestión directa del comercio

Es importante tener en cuenta que esta integración es compleja, y en cualquiera de los escenarios el comercio es responsable de la gestión de renderizar y manejar la interfaz de usuario UI y el flujo de sesión y presentación del botón de Apple Pay.



Ejemplos de uso del botón de Apple Pay. Para información sobre los botones entrar al link de "Apple Pay Mark".
Fuente: Apple.

Para familiarizarse con la documentación de Apple Pay previo seleccionar el método de implementación, favor consultar los siguientes enlaces

- Apple Pay on the Web – Interactive Demo
- Apple Pay PassKit
- Apple Pay on the web
- Apple Pay Mark

Integración a través de AZUL como Platform Integrator

Requisitos de integración

1. Ruta /.well-known

Deberá crear una ruta accesible desde internet en el servidor. Esta ruta sirve para agregar el archivo mencionado en el punto 3. Apple lo utiliza para poder validar el dominio en donde presentará el botón de pago.

2. Dominio

Debe enviar al equipo de AZUL (solucionesecommerce@azul.com.do) el dominio en el que estarán integrando el botón de Apple Pay (Tanto para producción como desarrollo).

3. Solicitar archivo para validación del dominio

Este archivo «**apple-developer-merchantid-domain-association.txt**» se lo enviara el equipo de AZUL una vez haya completado el desarrollo y tengan planificado el pase a producción. El archivo debe agregarse en la ruta creada /.well-known.

Escenario Platform Integrator

IMPORTANTE: Este método no es soportado para la implementación en App iOS.

En este escenario, el comercio debe utilizar el **Merchant ID de AZUL** y generar la sesión de pago por la integración vía API de AZUL. El comercio no requiere gestionar certificados vía la consola de administración de Apple Developer, ni tampoco tiene que implementar la desencriptación del payload generado por Apple Pay.

El **MerchantID de AZUL** que debe utilizar al instanciar el **SDK de Apple Pay** para estas implementaciones son:

Ambiente	URL
Pruebas (Sandbox)	payplatformintegrator.pruebas.azul.apple.pay
Producción	platformintegrator.pagos.azul.apple.pay

Al momento de generar el intento de pago, deberán solicitar a través de la función «**ApplePayPaymentSession**» de la integración vía API de AZUL la sesión de pago.

Este endpoint recibe 4 parámetros:

Parámetros de Entrada	
Parámetro	Descripción
Store	MerchantID de AZUL.
Channel	Valor de canal asignado por AZUL.
StoreDisplayName	Nombre del comercio que aparecerá en el prompt de Apple Pay.
InitiativeContext	Dominio base que va a cargar el botón de Apple Pay.

En la respuesta, deberá tomar el valor del campo «**ApplePayPaymentSessionDataJSON**» y pasarlo al evento complete de «**onmerchantvalidation**».

```
// Create PaymentRequest
conts request = new PaymentRequest(paymentMethodData, paymentDetails, paymentOptions);
request.onmerchantvalidation = event => {
       event.complete(ApplePayPaymentSessionDataJSON);
};
```

Al completar el flujo de pago, deberá enviar el payload de Apple Pay en el campo «**PaymentToken**» de Apple Pay.

```
…
<ApplePay>
    <PaymentToken>{Payload Apple Pay sin desencriptar}</PaymentToken>
</ApplePay>
…
```

Debe enviar el request completo de pago, en los siguientes endpoints:

JSON

Ambiente	URL
Pruebas (Sandbox)	https://pruebas.azul.com.do/WebServices/JSON/default.aspx
Producción	https://pagos.azul.com.do/WebServices/JSON/default.aspx

SOAP

Ambiente	URL
Pruebas (Sandbox)	https://pruebas.azul.com.do/WebServices/SOAP/default.asmx
Producción	https://pagos.azul.com.do/WebServices/SOAP/default.asmx

NOTA: Apple siempre devolverá una estructura JSON como resultado de la operación de Apple Pay:

- Para XML puede enviar directamente la estructura que devuelve Apple.
- Para JSON deberá ser enviado en el campo «**PaymentToken**» en formato string, utilizando la codificación correcta de acorde al formato de mensajería de AZUL.

Puede visualizar un ejemplo de un mensaje completo aquí: XML | JSON. Envíe el resto de las opciones o campos tal cual haría en una transaccional tradicional.

Mensaje Completo XML:

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Header>
    <SOAPAuthHeader xmlns="http://Merit/AzulIS/TransactionServices/">
      <Auth1>dccdo</Auth1>
      <Auth2>dccdo</Auth2>
    </SOAPAuthHeader>
  </soap:Header>
  <soap:Body>
    <ProcessPayment xmlns="http://Merit/AzulIS/TransactionServices/">
      <ProcessPaymentRequest>
        <Channel>EC</Channel>
        <Store>39430190017</Store>
        <PosInputMode>E-Commerce</PosInputMode>
        <TrxType>Sale</TrxType>
        <Amount>1075</Amount>
        <Itbis>121</Itbis>
        <ApplePay>
          <PaymentToken>{Payload de Apple Pay Sin Desencriptar}</PaymentToken>
        </ApplePay>
        <CardHolderInfo>
          <Name>Nombe Cardholder</Name>
          <Email></Email>
          <PhoneHome></PhoneHome>
          <PhoneMobile></PhoneMobile>
          <PhoneWork></PhoneWork>
          <BillingAddressLine1></BillingAddressLine1>
          <BillingAddressLine2></BillingAddressLine2>
          <BillingAddressLine3></BillingAddressLine3>
          <BillingAddressCity></BillingAddressCity>
          <BillingAddressState></BillingAddressState>
          <BillingAddressCountry></BillingAddressCountry>
          <BillingAddressZip></BillingAddressZip>
          <ShippingAddressLine1></ShippingAddressLine1>
          <ShippingAddressLine2></ShippingAddressLine2>
          <ShippingAddressLine3></ShippingAddressLine3>
          <ShippingAddressCity></ShippingAddressCity>
          <ShippingAddressState></ShippingAddressState>
          <ShippingAddressCountry></ShippingAddressCountry>
          <ShippingAddressZip></ShippingAddressZip>
        </CardHolderInfo>
        <BrowserInfo>
          <AcceptHeader>text/html,application/xhtml+xml,application/xml;q=0.9,image/avif
,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9</AcceptHeader>
          <IPAddress>127.0.0.1</IPAddress>
          <Language>en-US</Language>
          <ColorDepth>24</ColorDepth>
          <ScreenWidth>2880</ScreenWidth>
          <ScreenHeight>1800</ScreenHeight>
          <TimeZone>240</TimeZone>
          <UserAgent>Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/99.0.4844.51 Safari/537.36</UserAgent>
          <JavaScriptEnabled>true</JavaScriptEnabled>
        </BrowserInfo>
      </ProcessPaymentRequest>
    </ProcessPayment>
  </soap:Body>
</soap:Envelope>
```

Mensaje Completo JSON:

```
{
        "Channel":"EC",
        "Store":"39430190017",
        "PosInputMode":"E-Commerce",
        "TrxType":"Sale",
        "Amount":"1075",
        "Itbis":"121",
        "ApplePay": {
                "PaymentToken":"{json_scape(Respuesta de Apple Pay)}"
        },
        "CardHolderInfo": {
                "BillingAddressCity":"",
                "BillingAddressCountry":"",
                "BillingAddressLine1":"",
                "BillingAddressState":"",
                "BillingAddressZip":"",
                "Email":"",
                "Name":"Nombre Cardholder",
                "PhoneHome":"",
                "PhoneMobile":"",
                "PhoneWork":"",
                "ShippingAddressCity":"",
                "ShippingAddressCountry": "",
                "ShippingAddressLine1":"",
                "ShippingAddressState":"",
                "ShippingAddressZip":""
        },
        "BrowserInfo": {
                "AcceptHeader":
        "text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/a
        png,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9",
                "IPAddress":"127.0.0.1",
                "Language":"en-US",
                "ColorDepth":"24",
                "ScreenWidth":"2880",
                "ScreenHeight":"1800",
                "TimeZone":"240",
                "UserAgent":"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
        (KHTML, like Gecko) Chrome/99.0.4844.51 Safari/537.36",
                "JavaScriptEnabled":"true"
        }
}
```

Merchant Identifier

Este método requiere que el comercio gestione en la **consola de Apple Developer** su certificado de encriptación/desencriptación, manejo de sesión y validación del dominio/app del comercio.

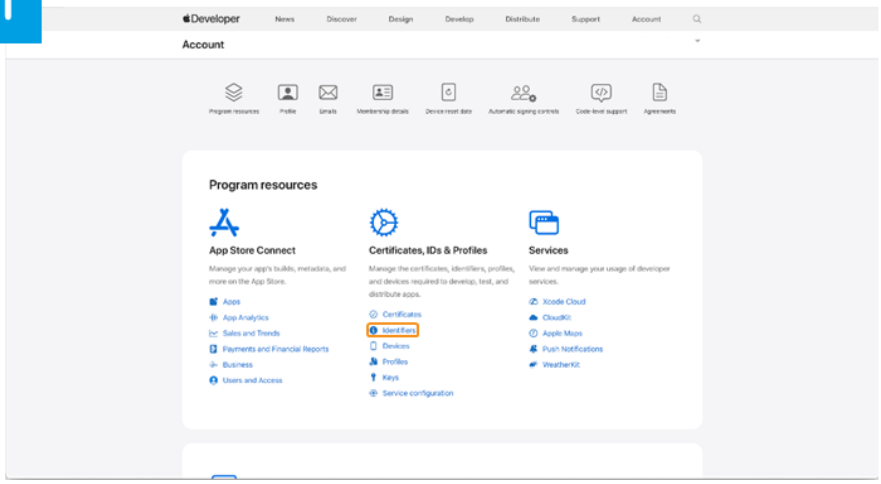
Al momento de realizar el pago, el comercio deberá instanciar la SDK de Apple Pay utilizando su **propio Merchant ID** y generar su sesión de pago con Apple directamente (sin usar endpoint ApplePayPaymentSession se AZUL). El payload generado debe ser enviado a AZUL de la misma forma que el método anterior.

Para poder operar correctamente, el comercio deberá proveerle a AZUL el certificado con la llave privada (PFX) que será utilizado para la desencriptación. Es importante entender que Apple provee 2 certificados.

- **Apple Pay Payment Processing Certificate**: Certificado utilizado para encriptar y desencriptar el payload. Este es el certificado que debe ser proporcionado a AZUL.
- **Apple Pay Merchant Identity Certificate**: Certificado utilizado para identificar al comercio y es funciona para generar la sesión de pago. AZUL no requiere este certificado.

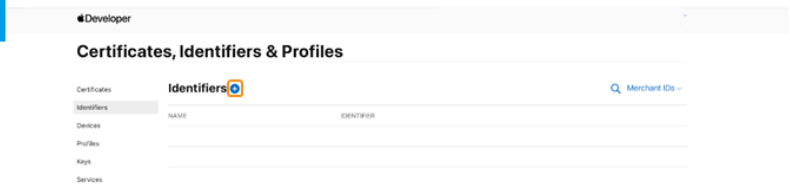
El manejo de los certificados en Apple Pay se realiza con el siguiente flujo:

1



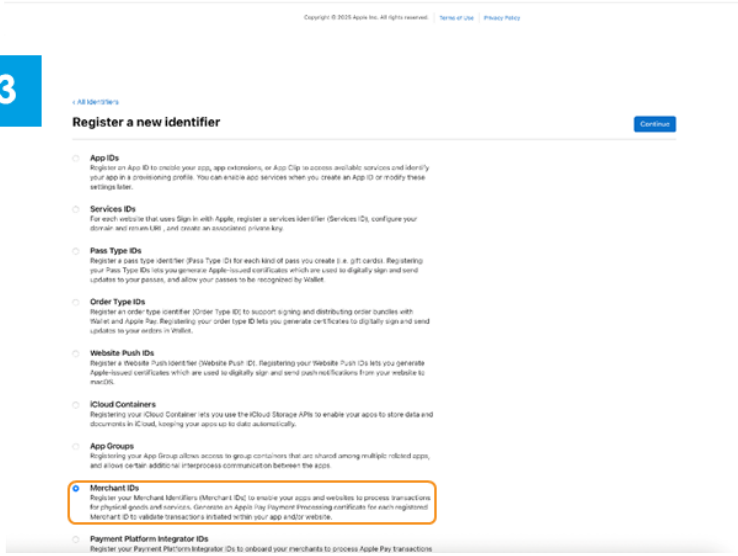
Ir a la sección "**Identifiers**"

2



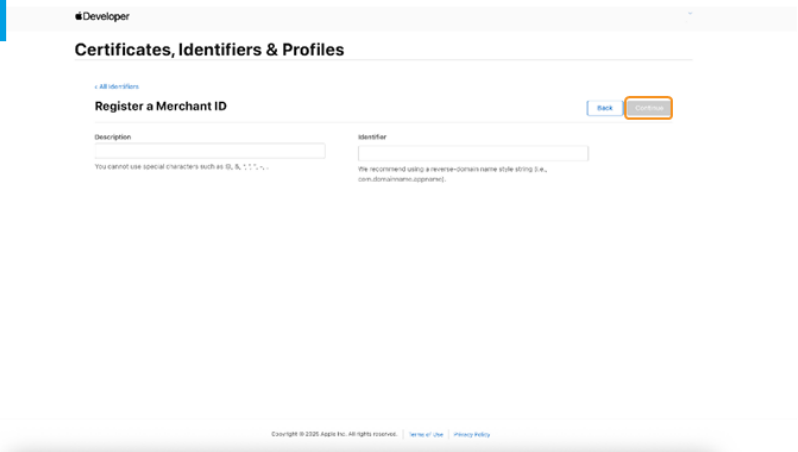
Pulsa ⊕ para agregar un nuevo identificador

3



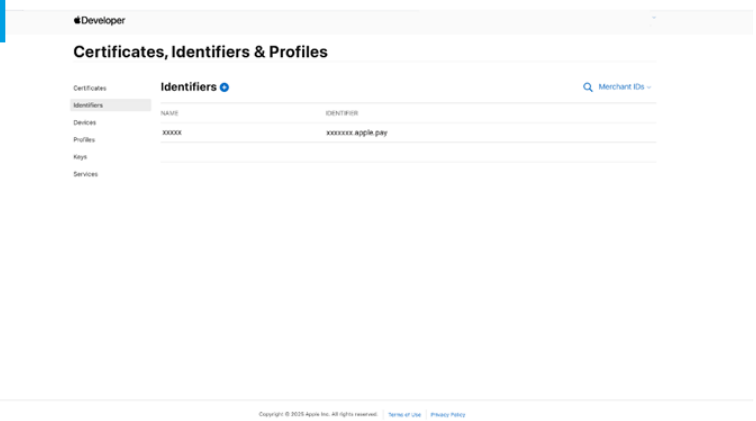
Selecciona en la opción "**Merchant ID**"

4



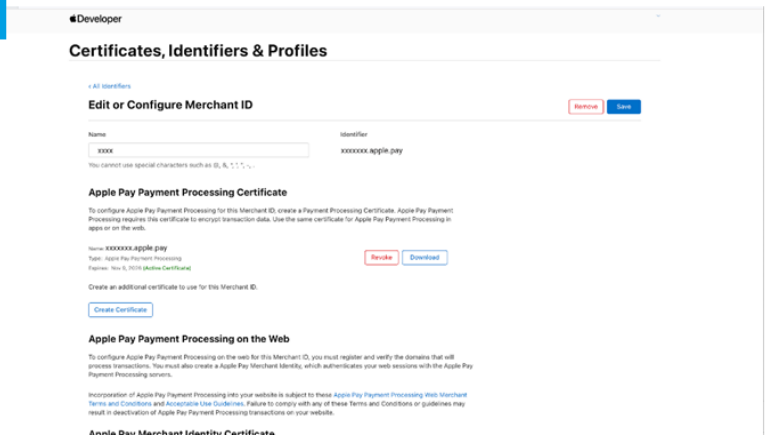
Completa los campos requeridos y pulsa "**Continuar**"

5



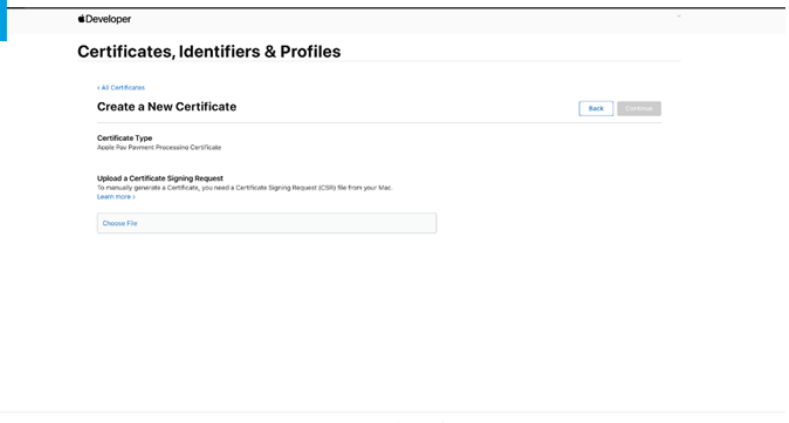
Haz clic sobre el identificador creado para acceder a su configuración

6



Selecciona "**Create Certificate**" en caso de que aún no haya generado uno

7



Subir el archivo CSR. Debajo encontrarás instrucciones sobre cómo generar el archivo.

El proceso para gestionar la creación de los 2 certificados es el siguiente:

Paso 1: Genera una llave privada y un CSR para el certificado de Payment Processing vía openssl:

```
openssl ecparam -genkey -name prime256v1 -out paymentprocessingkey.key
openssl req -new -sha256 -key paymentprocessingkey.key -out paymentprocessingkey.csr -subj
/CN=nombrecomercio.apple.pay
```

Paso 2: Genera una llave privada y CSR para el certificado de Merchant ID, este es un RSA2048:

```
openssl genrsa -out merchantid.key 2048
openssl req -new -sha256 -key merchantid.key -out merchantid.pem
```

Paso 3: Acceder a la consola de Apple Developer e ir a la opción "**Identifiers**". Seleccionar "**Create certificate**" y subir el CSR correspondiente. Hay que hacer el proceso 2 veces, uno para el certificado de Payment Processing y luego con el Merchant Identity. Guardar ambos en disco junto a las llaves privadas con los nombres "paymentprocessing.cer" y "merchantid.cer"

Convertir ambos certificados a formato DER:

```
openssl x509 -inform DER -in paymentprocessing.cer -out paymentprocessing.pem
openssl x509 -inform DER -in merchantid.cer -our merchantid.pem
```

Y luego unificarlo con la llave privada:

```
openssl pkcs12 -export -out paymentprocessing.12 -inkey paymentprocessing.key -in
paymentprocessing.pem
openssl pkcs12 -export -our merchantid.12 -inkey merchantid.key -in merchantid.pem
```

Adicionalmente, en la pantalla de "**Merchant Identifiers**" el comercio debe completar la validación del dominio que usara para implementaciones web en la parte de "**Merchant Domains**".

Gestión por comercio

En este método el comercio es responsable de gestionar su certificado de desencriptación en su plataforma y también desencriptar el payload de Apple Pay de su lado. El payload contiene un número de tarjeta tokenizado que debe ser enviado a AZUL al momento de autorizar la transacción. El certificado de desencriptación no es manejado ni instalado del lado de AZUL ya que es el comercio que realiza la acción.

Para estas transacciones se debe enviar el número de token en el campo «**CardNumber**», e incluir los campos «**ECIIndicator**» y «**Cryptogram**»:

```
…
<CardNumber>4824****0618****7871****8721</CardNumber>
<ECIIndicator>07</ECIIndicator>
<Cryptogram>12345678901234578901=</Cryptogram>
…
```

Para más información sobre como desencriptar el payload de Apple Pay, consulte este link:

- Payment Token Format Reference