Implementación

Updated on 17 Dec 2025 · 8 Minutes to read · Contributors

La implementación de **ePayco Smart Checkout** se realiza mediante una integración JavaScript simple y directa que te permite desplegar el checkout con cualquier evento en tu aplicación web.

Uso

Creación de la sesión desde el backend

El primer paso para implementar **ePayco Smart Checkout** es crear una sesión de checkout cuando el cliente esté listo para realizar la compra. Esta sesión debe ser creada desde tu backend favorito mediante una petición a nuestras APIs, la cual generará un ID de sesión único que posteriormente utilizarás para inicializar el componente de **ePayco Smart Checkout** en el frontend y garantizar que la sesión esté correctamente vinculada con tu sistema de gestión de órdenes.

Autenticación con Apify

Antes de crear la sesión de **ePayco Smart Checkout**, debes autenticarte con los servicios de **Apify** (https://api.epayco.co/) de ePayco para obtener un token de acceso. Este proceso requiere que seas un usuario registrado en ePayco y utiliza autenticación Basic enviando tu `PUBLIC_KEY` y `PRIVATE_KEY` (las cuales puedes obtener en tu **dashboard de ePayco** (https://dashboard.epayco.com/configuration) en la sección de Configuración → Personalizaciones → Llaves secretas. en los headers de la petición. El token que obtendrás de este endpoint te permitirá firmar las solicitudes posteriores necesarias para crear la sesión de **ePayco Smart Checkout** de forma segura.

Solicitud

Bash | Copy

```bash
curl --location --request POST 'https://apify.epayco.co/login' \
--header 'Content-Type: application/json' \
--header 'Authorization: Basic <PUBLIC_KEY:PRIVATE_KEY en base64>'
```

Respuesta exitosa

JSON Copy

```
{
    "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJhcGlmeWVQYXljb
}
```

Autenticación con Apify

Una vez obtenido el token de autenticación, el siguiente paso es crear la sesión de **Smart Checkout** que generará el `sessionId` necesario para inicializar el componente en el frontend. Esta petición debe incluir la información de la transacción como el nombre del producto, descripción, monto, moneda, etc. El `sessionId` retornado será utilizado para abrir el **Smart Checkout** de pago en tu sitio web.

Solicitud

Bash Copy

```
curl --location 'https://apify.epayco.co/payment/session/create' \
--header 'Content-Type: application/json' \
--header 'Authorization: Bearer {{APIFY_TOKEN}}' \
--data '{
    "checkout_version": "2",
    "name": "Shops Online S.A.S",
    "currency": "COP",
    "amount": 200000
}'
```

> Todas las propiedades disponibles que describen más abajo.

Respuesta exitosa

JSON Copy


```
{
    "success": true,
    "titleResponse": "Session created successfully",
    "textResponse": "Session created successfully",
    "lastAction": "create session",
    "data": {
        "sessionId": "68eefce71d65f1d39c0f6dad",
        "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6IjY4ZWVmY2U3M
    }
}
```


Propiedades

A continuación se describen todas las propiedades que puedes utilizar al crear una sesión de **ePayco Smart Checkout**. Las propiedades están organizadas en requeridas y opcionales, incluyendo objetos anidados para configuraciones avanzadas como división de pagos, campos personalizados y datos de facturación.

Requeridos | Opcionales | Billing (opcional… | Extras (opcional… | Configu

Propiedad	Tipo	Descripción
checkout_version	string	Versión del checkout de ePayco. V
name	string	Nombre del comercio o tienda
currency	string	Código de la moneda (ej: `"COP"` ,
amount	number	Monto total de la transacción

Ejemplo Completo

A continuación se muestra un objeto JSON completo con todas las propiedades disponibles para crear una sesión de **ePayco Smart Checkout**:

JSON Copy


```
{
    // Requeridos: Información básica de la transacción
    "checkout_version": "2",
```

```
    "name": "Shops Online S.A.S",
    "currency": "COP",
    "amount": 20000.00, // Total a pagar
    // Opcionales: Datos adicionales
    "description": "Buzo con capucha color negro unisex",
    "lang": "ES", // Idioma: ES | EN
    "country": "CO", //País: Código alpha-2
    "taxBase": 16806.72,
    "tax": 3193.28,
    "taxIco": 0,
    "response": "https://mysite.com",
    "confirmation": "https://webhook.site/8b4bb363-099e-42e8-afe7-0bf11c59ee
    // Configuración: Lógicas aplicadas a la transacción
    "methodsDisable": [],
    "method": "POST", // Método de confirmación de la transacción: GET | POS
    "dues": 1, // Cuotas default en los formularios
    // Split payment: disperción de pagos
    "splitPayment": {
        "type": "percentage",
        "receivers": [
            {
                "merchantId": 9898,
                "amount": 15000.00,
                "taxBase": 12605.04,
                "tax": 2394.96,
                "fee": 1
            },
            {
                "merchantId": 1485968,
                "amount": 5000,
                "taxBase": 4201.68,
                "tax": 798.32,
                "fee": 1
            }
        ]
    },
    // Extras: Información adicional
    "extras": {
        "extra1": "extra1",
        "extra2": "extra2",
        "extra3": "extra3",
        "extra4": "extra4",
        "extra5": "extra5",
        "extra6": "extra6",
        "extra7": "extra7",
        "extra8": "extra8",
```

```
        "extra9": "extra9",
        "extra10": "extra10",
        "extra11": "extra11"
    },
    // Billing: Información del comprador para autocompletar formularios
    "billing": {
        "email": "cliente@gmail.com",
        "name": "CLiente Martinez",
        "address": "AV 18 # 18 - 17",
        "typeDoc": "CC",
        "numberDoc": "103242123",
        "callingCode": "+57",
        "mobilePhone": "312456654"
    }
}
```

> **Nota:** Este ejemplo incluye todas las propiedades disponibles. Solo las propiedades requeridas `checkout_version`, `name`, `currency`, `amount`) son obligatorias. El resto son opcionales y puedes incluirlas según las necesidades de tu implementación.

Implementación en el frontend

Una vez que hayas obtenido el `sessionId` desde tu backend, el último paso es inicializar y mostrar el componente de **ePayco Smart Checkout** en tu sitio web. Primero debes incluir la librería JavaScript de ePayco en tu página, luego configurar el **Smart Checkout** con el `sessionId` obtenido y finalmente abrirlo con el evento de tu preferencia.

1. Incluir la librería de ePayco

Agrega el siguiente script en tu página HTML:

XML Copy

```xml
<script src="https://checkout.epayco.co/checkout-v2.js"></script>
```

2. Configurar y abrir el checkout

JavaScript Copy

```
const checkout = ePayco.checkout.configure({
      sessionId: sessionId,
      type: "onpage", // "onpage" | "standard"
      test: true // Test: true | false
});

// Abrir el checkout
checkout.open();
```

3. Event handlers (opcionales)

ePayco Smart Checkout proporciona un sistema de eventos que permite responder a diferentes etapas del flujo de pago. Estos eventos están disponibles para los tipos de implementación `onpage` .

JavaScript Copy

```
checkout.onCreated(() => {
    console.log("Evento cuando se crea transacción:");
    // Guardar marcación de apertura.
});

checkout.onErrors((errors) => {
    console.error("Evento que notifica un error");
    console.error(errors);
    // Mostrar mensaje de errores procedentes de la creación de la transacci
});

checkout.onClosed(() => {
    console.log("Evento cuando se cierra el Smart Checkout");
    // Limpiar estado, verificar estado de transacción, redirigir de manera
});
```

Descripción de cada evento:

- `onCreated(callback)` Se dispara cuando se crea abre exitosamente el **Smart Checkout**.

- `onErrors(callback)` Se activa cuando ocurre un error durante el proceso de obtener la transacción en el **Smart Checkout**. Permite manejar errores de forma personalizada y mostrar mensajes apropiados al usuario.

- `onClosed(callback)` Se dispara cuando el usuario cierra el **Smart Checkout**, ya sea después de completar el pago o cancelar el proceso. Ideal para limpiar el estado de la aplicación o redirigir al usuario.

Ejemplo de implementación completa con eventos:

XML Copy

```xml
<!DOCTYPE html>
<html lang="es">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Smart Checkout ePayco</title>
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootst
</head>

<body class="bg-light">

    <div class="container mt-5">
        <div class="row justify-content-center">
            <div class="col-md-6">

                <div class="text-center">
                    <h2 class="mb-4">💳 Smart Checkout ePayco</h2>
                    <p class="text-muted mb-4">Demo funcional del sistema de

                    <button id="payBtn" class="btn btn-success btn-lg px-5 p
                        💳 Pagar $100.000 COP
                    </button>
                </div>

            </div>
        </div>
    </div>

    <!-- Scripts -->
    <script src="https://checkout.epayco.co/checkout-v2.js"></script>
    <script>
        // Función que se ejecuta al hacer clic
        async function handlePayment() {
            console.log("🔄 Iniciando proceso de pago...");

            // 1. Obtener sessionId del backend
```

```
            const response = await fetch('/api/create-session', {
                method: 'POST',
                headers: { 'Content-Type': 'application/json' },
                body: JSON.stringify({ amount: 10000, currency: 'COP' })
            });

            const { sessionId } = await response.json();
            console.log("✅ SessionId obtenido:", sessionId);

            // 2. Configurar checkout
            const checkout = ePayco.checkout.configure({
                sessionId,
                type: "onpage",
                test: true
            });

            // 3. Eventos
            checkout.onCreated(() => console.log("✅ Checkout creado"));
            checkout.onErrors(e => console.error("❌ Error:", e));
            checkout.onClosed(() => console.log("🔒 Checkout cerrado"));

            // 4. Abrir checkout
            checkout.open();
        }

        // Asignar evento al botón
        document.addEventListener('DOMContentLoaded', () => {
            document.getElementById("payBtn").onclick = handlePayment;
        });
    </script>
</body>

</html>
```


Previous
General

Next
Páginas de Respuesta y Confirmación