



Manual de Integração

Versão 1.0



Índice

1. Considerações Iniciais	3
2. Ambiente de Testes	4
3. Token da Conta	5
4. Tipos de Integração.....	6
5. Integração Post	7
5.1 Integração em 1 minuto.....	7
5.2 Integração Completa via POST.....	7
6. Integração via API	9
6.1 Integração de Transação via API.....	9
6.2 Integração de Carrinho de Compras via API	15
7. NAS – Notificação Automática de Status.....	19
8. Módulos para Softwares de Terceiros	25
9. Tabelas Auxiliares.....	26

1. Considerações Iniciais

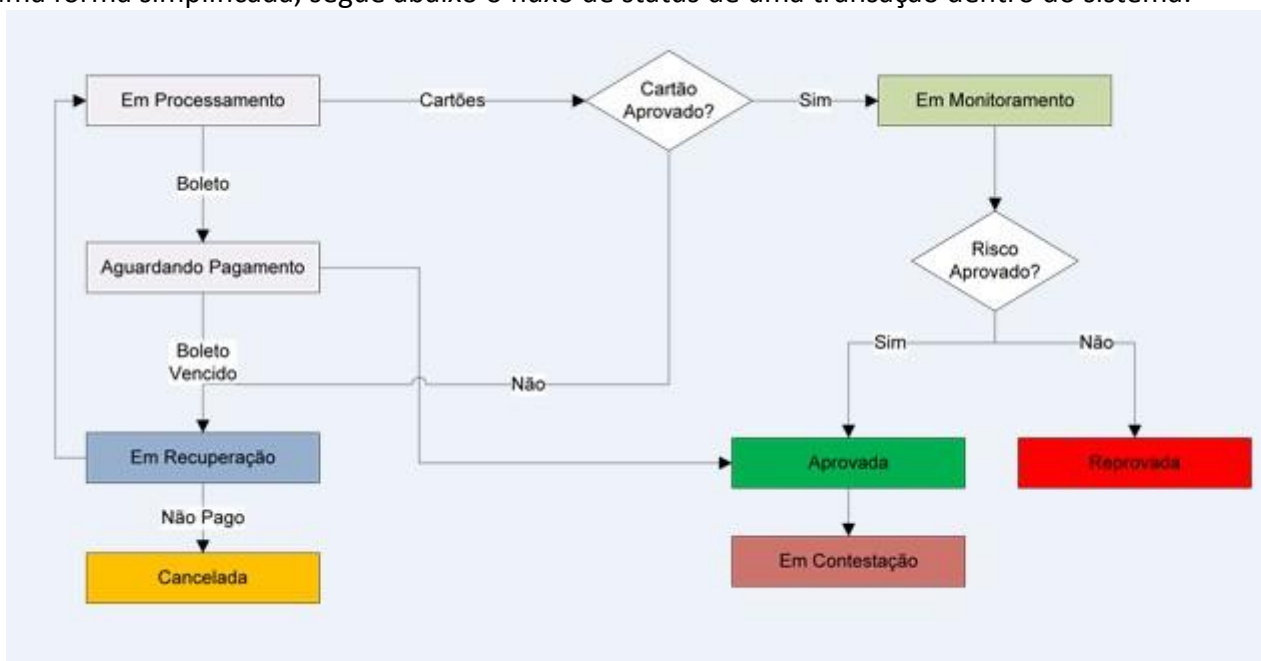
O TrayCheckout é um facilitador de pagamento que oferece benefícios aos lojistas e aos compradores. Focado em praticidade e conversão, possibilita que as lojas virtuais ofereçam diversas formas de pagamento, sem burocracia ou necessidade de contrato com operadoras financeiras.

Para a utilização, basta possuir uma conta no sistema. A criação poderá ser feita através do endereço: <http://www.traycheckout.com.br/criar-conta>.

As taxas referentes ao serviço podem ser consultadas em: <http://www.traycheckout.com.br/tarifas>.

O guia completo sobre a utilização do TrayCheckout pode ser acessado em: <http://wiki.traycheckout.com.br>.

De uma forma simplificada, segue abaixo o fluxo de status de uma transação dentro do sistema:



Caso encontrar dificuldades no processo de integração, poderá ser feito contato com nossa equipe de Integração, que irá auxiliar na resolução dos problemas. O contato poderá ser realizado através do e-mail: integracao@traycheckout.com.br.

2. Ambiente de Testes

Para realizar as integrações, o TrayCheckout disponibiliza um ambiente de testes chamado Sandbox. Este ambiente permite que sejam efetuadas todas as operações disponíveis em produção, com simulações sem a necessidade de uso de dados reais.

Mesmo que você já possua uma conta no ambiente de produção, será necessário criar uma conta específica para testes acessando <http://sandbox.traycheckout.com.br/signin>.

Durante os testes em Sandbox, as compras que forem processadas com cartões de crédito poderão ser aprovadas ou reprovadas em tempo real. Este procedimento poderá ser realizado de acordo com o parcelamento escolhido:

- Aprovada: transações com parcelamento ímpar (1, 3, 5, 7, 9, 11);
- Reprovada: transações com parcelamento par (2, 4, 6, 8, 10, 12).

3. Token da Conta

Para as integrações será necessário possuir o token da conta que será integrada.

Para obter esta informação, acesse o site www.traycheckout.com.br e clique no botão “Acesse sua conta”, localizado no canto direito superior da página. Informe seu e-mail e senha, e então terá acesso a sua conta.

Clique então no menu “Minha Conta” e então no item “Minha Conta”.



4. Tipos de Integração

O TrayCheckout permite duas formas de integração: POST e API.

Para a utilização da modalidade POST será necessário muito pouco conhecimento em programação, bastando apenas a inclusão de um código HTML em seu site. Esta modalidade fará com que o seu cliente seja redirecionado para o TrayCheckout, onde finalizará sua compra.



Para a utilização da modalidade API é necessário conhecimento de programação, pois necessitará a integração de seu sistema com as APIs disponibilizadas pelo TrayCheckout. Esta modalidade tornará o processo de compra transparente para o usuário, sem necessidade de criação de novos cadastros e sem o redirecionamento para um ambiente externo.



Abaixo uma comparação sobre os tipos de integração:

	Integração via POST	Integração via API
Dificuldade de Integração	Baixa	Alta
Necessidade de conhecer programação	Não	Sim
Finalização da compra	Redirecionamento	Transparente
Processamento de transação	Sim	Sim
Carrinho de compras TrayCheckout	Não	Sim
NAS – Notificação Automática de Status	Sim	Sim
Compra com conta Facebook	Sim	Não

NAS – Notificação Automática de Status

Para ambos os tipos de integração o estabelecimento poderá receber as atualizações de status dos pedidos. Este assunto será melhor tratado no capítulo 7.

5. Integração Post

A integração POST permite que usuários com pouco conhecimento em programação possam integrar-se com o TrayCheckout. Esta integração irá direcionar o cliente de seu site para o TrayCheckout, então será feita a finalização da compra.

5.1 Integração em 1 minuto

O TrayCheckout disponibiliza um gerador de botões padrões. Com este gerador, você informará o token da conta, identificação do pedido e dados referentes ao produto que está sendo comprado; após informar os dados, você obterá o código fonte HTML que deverá ser copiado e colado em seu site, no local que deseja exibir o botão de compra.

Para obter o botão, acesse: <http://www.traycheckout.com.br/desenvolvedores?tab=buttons>.

5.2 Integração Completa via POST

Caso desejar uma integração com mais dados e produtos, poderá ser utilizada uma integração mais avançada com o TrayCheckout. Esta transação irá receber além dos dados da conta e pedido, uma maior quantidade de produtos.

Para a integração basta ser realizado um POST para a API:

Endereço para Integração	
Ambiente de Testes	https://checkout.sandbox.tray.com.br/payment/transaction
Ambiente de Produção	https://checkout.tray.com.br/payment/transaction
Protocolo	Rest/HTTP

Os dados que podem ser enviados neste POST HTML são:

Dados de Entrada	Obrigatório	Formato / Tamanho Max	Descrição
token_account	Sim	Texto / 15	Token de identificação da loja
transaction_product[][description]	Sim	Texto / 150	Descrição do produto
transaction_product[][quantity]	Sim	Número Inteiro / 5	Quantidade do produto
transaction_product[][price_unit]	Sim	Decimal / Ex: 0.00	Valor do produto
transaction_product[][code]		Número / 20	Código do produto
transaction_product[][extra]		Texto / 45	Informações extras do produto
transaction_product[][url_img]		Url / 255	URL da imagem do produto
transaction_product[][sku_code]		Texto / 20	Código SKU do produto
order_number		Número Inteiro/20	Número do pedido
free		Texto / 100	Informações extras da transação
price_additional		Decimal / Ex: 0.00	Valor adicional da transação
price_discount		Decimal / Ex: 0.00	Valor do desconto da transação
url_notification		Url / 255	Endereço do site de notificação
shipping_type		Texto / 15	Tipo de envio

shipping_price		Decimal / Ex: 0.00	Valor do envio
sub_store		Texto / 20	Sub-Loja
url_css		Url / 255	URL do CSS customizado

Após receber o post de dados, o TrayCheckout irá automaticamente redirecionar o usuário para a página de finalização de compra, onde o mesmo poderá escolher a forma de pagamento e informar os dados do cartão de crédito.

Dados de Produtos

Note que nas informações acima os dados de produtos possuem uma característica diferente, tendo um elemento [] dentro de sua formatação, como por ex: `transaction_product[][description]`. Isso ocorre justamente para permitir que sejam enviados diversos produtos no mesmo POST.

Veja abaixo um POST de exemplo com 2 produtos:

```
<form method="post" action="https://checkout.tray.com.br/payment/transaction">

  <!-- Campos obrigatórios -->
  <input type="hidden" name="token_account" value="SEU_TOKEN_AQUI">

  <!-- Itens do produto 1 -->
  <input type="hidden" name="transaction_product[][description]" value="Notebook Prata">
  <input type="hidden" name="transaction_product[][quantity]" value="1">
  <input type="hidden" name="transaction_product[][price_unit]" value="2430.00">

  <!-- Itens do produto 2 -->
  <input type="hidden" name="transaction_product[][description]" value="Notebook Branco">
  <input type="hidden" name="transaction_product[][quantity]" value="1">
  <input type="hidden" name="transaction_product[][price_unit]" value="2599.00">

  <!-- Código de referência do pagamento no seu sistema (opcional) -->
  <input type="hidden" name="order_number" value="56789">

  <!-- submit do form -->
  <input type="submit" name="submit" value="Pagar com TrayCheckout" >

</form>
```


6. Integração via API

A integração via API permite que o desenvolvedor utilize o TrayCheckout de uma forma transparente, integrando com sua Loja Virtual, Aplicativo Mobile ou qualquer outro tipo de software, personalizando as integrações com o usuário para melhor se adequar ao software existente.

Para garantir que as integrações não sejam impactadas com mudanças nas APIs, o TrayCheckout utiliza uma nomenclatura específica para versioná-las:

Ambiente de Produção

- Versão Atual: <https://api.traycheckout.com.br/edge>
- Versão Específica: <https://api.traycheckout.com.br/v2>

Ambiente de Testes - Sandbox

- Versão Atual: <https://api.sandbox.traycheckout.com.br/edge>
- Versão Específica: <https://api.sandbox.traycheckout.com.br/v2>

Onde **v2** significa o número da versão que deseja utilizar, neste caso tratando-se da versão v2.

6.1 Integração de Transação via API

O TrayCheckout disponibiliza uma versão transparente para a integração de transações, permitindo que o usuário efetue o processamento dos pedidos sem necessitar o redirecionamento para outra aplicação e preenchimento de novos formulários.

Através do CPF do cliente é feita a consulta pela existência do seu cadastro e então as transações são atreladas ao mesmo. Caso não exista uma conta, o sistema irá criar uma nova conta com os dados que forem submetidos na integração.



Para esta integração, deverá ser feito uso da API a seguir:

Endereço de Integração	
Ambiente de Testes	https://api.sandbox.traycheckout.com.br/v2/transactions/pay_complete
Ambiente de Produção	https://api.traycheckout.com.br/v2/transactions/pay_complete
Protocolo	Rest/HTTP

Os dados que podem ser enviados nessa API são:

Dados de Entrada	Obrigatório	Formato / Tamanho Max	Descrição
token_account	Sim	Texto /20	Token de identificação do vendedor
customer[name]	Sim	Texto /100	Nome do Comprador
customer[cpf]	Sim	Texto /14	CPF do Comprador
customer[email]	Sim	Texto /100	E-mail do Cliente
customer[addresses][][type_contact]	Sim	Texto /1	Tipo do Contato. Veja ANEXO
customer[contacts][][number_contact]	Sim	Texto /10	Número do telefone do cliente
customer[addresses][][type_address]	Sim	Texto /1	Tipo do Endereço. Veja ANEXO
customer[addresses][][postal_code]	Sim	Texto /8	CEP do endereço do cliente
customer[addresses][][street]	Sim	Texto /120	Nome da rua do cliente
customer[addresses][][number]	Sim	Texto /10	Número do endereço do cliente
customer[addresses][][neighborhood]	Sim	Texto /100	Bairro do endereço do cliente
customer[addresses][][completion]		Texto /100	Complemento do endereço
customer[addresses][][city]	Sim	Texto /120	Cidade do endereço do cliente
customer[addresses][][state]	Sim	Texto /2	Estado do endereço do cliente
customer[birth_date]		Data / 10	Data no formato dd/mm/aaaa
transaction[available_payment_methods]		Texto /20	Veja NOTA 1
transaction[order_number]		Texto /2	Número do pedido
transaction[shipping_type]		Texto /100	Tipo do envio
transaction[shipping_price]		Decimal / 11	Preço de Envio. Formato: 0.00
transaction[price_discount]		Decimal / 11	Valor do desconto. Formato: 0.00
transaction[price_additional]		Decimal / 11	Valor adicional. Formato: 0.00
transaction[url_notification]		Texto /255	URL de Notificação de Status
transaction[free]		Texto /200	Campo Livre
transaction[sub_store]		Texto /20	Sub-Loja
transaction_product[][description]	Sim	Texto /100	Nome do produto
transaction_product[][quantity]	Sim	Número / 3	Quantidade do item do produto
transaction_product[][price_unit]	Sim	Decimal / 11	Valor unitário. Formato: 0.00
transaction_product[][code]		Texto /10	Código do produto
transaction_product[][sku_code]		Texto /50	Código SKU do produto
transaction_product[][extra]		Texto /100	Campo Livre do produto
payment[payment_method_id]	Sim	Texto /2	Forma de Pagamento. Veja ANEXO
payment[split]	Sim	Texto /2	Número de parcelas (01 a 12)
payment[card_name]		Texto /100	Nome impresso no cartão
payment[card_number]		Número /20	Número do cartão
payment[card_expdate_month]		Número /2	Mês de vencimento do cartão
payment[card_expdate_year]		Número / 4	Ano de vencimento do cartão
payment[card_cvv]		Número /3	Código de segurança do cartão

Note que nas informações acima os dados de produtos, telefone e endereço possuem uma característica diferente, tendo um elemento [] dentro de sua formatação, como por ex:

transaction_product[][description].

Isso ocorre justamente para permitir que sejam enviados diversos itens na mesma chamada, sejam vários telefones, endereços ou produtos.

Veja abaixo uma chamada de API de exemplo com 2 produtos e 2 telefones, destacados em vermelho:

```
curl https://api.traycheckout.com.br/v2/transactions/pay_complete -d
  "token_account=SEU_TOKEN_AQUI
  &customer[name]=Nome do Cliente
  &customer[cpf]=12312312312
  &customer[email]=emaildo@cliente.com.br
  &customer[contacts][][type_contact]=M
  &customer[contacts][][number_contact]=11998761234
  &customer[addresses][][type_address]=B
  &customer[addresses][][postal_code]=04001001
  &customer[addresses][][street]= Av Paulista
  &customer[addresses][][number]=1001
  &customer[addresses][][neighborhood]=Centro
  &customer[addresses][][city]=São Paulo
  &customer[addresses][][state]=SP
  &transaction_product[][description]=Notebook Preto
  &transaction_product[][quantity]=1
  &transaction_product[][price_unit]=1999.99
  &transaction_product[][description]=Notebook Branco
  &transaction_product[][quantity]=1
  &transaction_product[][price_unit]=2199.99
  &transaction[order_number]=123456
  &transaction[free]=Texto Interno
  &payment[payment_method_id]=3
  &payment[split]=12
  &payment[card_name]=Nome Impresso no Cartão
  &payment[card_number]=1234123412341234
  &payment[card_expdate_month]=01
  &payment[card_expdate_year]=2016
  &payment[card_cvv]=123"
```

* *Necessário remover as quebras de linha (deixar inline) antes de executar o cURL.*

Resposta da API

A api de Processamento de Transações retorna a resposta em XML. No caso de sucesso, é retornado um nó **<data_response>**. No caso de erro, é retornado um nó **<error_response>**. A primeira parte da resposta identifica se houve erro ou sucesso através do nó **<message_response><message>**.

Exemplo de resposta com sucesso baseando no envio do exemplo acima:

```
<response>
  <message_response>
    <message>success</message>
  </message_response>
  <data_response>
    <transaction>
      <order_number>123456</order_number>
      <free>Texto Interno</free>
      <transaction_id type="integer">999999</transaction_id>
      <status_name>Aprovada </status_name>
      <status_id type="integer">6</status_id>
      <token_transaciton>3a2a84aa09d4ff8320f940070bcdb3fa</transaction_token>
      <payment>
        <price_payment type="decimal">4199.98</price_payment>
        <payment_response>Autorização aprovada</payment_response>
        <url_payment></url_payment>
        <tid>10347871500026BF1001</tid>
        <split type="integer">12</split>
        <payment_method_id type="integer">3</payment_method_id>
        <payment_method_name>Visa</payment_method_name>
        <linha_digitavel nil="true"/>
      </payment>
    </transaction>
  </data_response>
</response>
```

```

    </payment>
  </transaction>
</data_response>
</response>

```

Abaixo um detalhamento de cada nó do XML de resposta:

XML de Resposta	
<response>	Nó principal da resposta
<response> <message_response>	Nó que contém o resultado da resposta
<response> <message_response> <message>	Resposta sobre a solicitação Em caso de sucesso: sucess Em caso de erro: error
<response> <data_response>	Nó que contém os dados da resposta
<response> <data_response> <transaction>	Nó que contém as informações sobre a transação
<response> <data_response> <transaction> <order_number>	Número do pedido enviado pela loja na solicitação
<response> <data_response> <transaction> <free>	Texto livre enviado pela loja na solicitação
<response> <data_response> <transaction> <transaction_id type="integer">	Id da transação
<response> <data_response> <transaction> <status_name>	Descrição do status da transação
<response> <data_response> <transaction> <status_id type="integer">	Id do status da transação
<response> <data_response> <transaction> <token_transaction>	Token da transação. Deve ser armazenado para ser utilizado nas consultas posteriores.
<response> <data_response> <transaction> <payment>	Nó com informações sobre o pagamento
<response> <data_response> <transaction> <payment> <price_payment type="decimal">	Preço pago pelo cliente

<pre><response> <data_response> <transaction> <payment> <payment_response></pre>	Resposta de pagamento da operadora para ser exibida na página de recibo de compra
<pre><response> <data_response> <transaction> <payment> <url_payment></pre>	Url para exibir o boleto ou ambiente externo do banco. Apenas para formas de pagamento que não são fechadas online, como por exemplo o Boleto Bancário e Itau Shopline
<pre><response> <data_response> <transaction> <payment> <tid></pre>	Número único que identifica a transação na operadora
<pre><response> <data_response> <transaction> <payment> <split type="integer"></pre>	Número de parcelas da transação
<pre><response> <data_response> <transaction> <payment> <payment_method_id type="integer"></pre>	Id da forma de pagamento escolhida
<pre><response> <data_response> <transaction> <payment> <payment_method_name></pre>	Nome da forma de pagamento escolhida
<pre><response> <data_response> <transaction> <payment> <linha_digitavel nil="true"/></pre>	Linha digitável para pagamento via Internet Banking. Apenas no caso da escolha por Boleto Bancário

A seguir um exemplo de um retorno com erro:

```
<response>
  <message_response>
    <message>error</message>
  </message_response>
  <error_response>
    <general_errors type="array">
      <general_error>
        <code>003039</code>
        <message>Vendedor inválido ou não encontrado</message>
      </general_error>
    </general_errors>
  </error_response>
</response>
```

E o detalhamento de cada nó do XML de resposta:

<response>	Nó principal da resposta
<response> <message_response>	Nó que contém o resultado da resposta
<response> <message_response> <message>	Resposta sobre a solicitação Em caso de sucesso: sucess Em caso de erro: error
<response> <error_response>	Nó contendo os erros encontrados
<response> <error_response> <general_errors type="array">	Nó contendo os erros encontrados
<response> <error_response> <general_errors type="array"> <general_error>	Nó contendo o detalhamento de um erro
<response> <error_response> <general_errors type="array"> <general_error> <code>	Código do erro
<response> <error_response> <general_errors type="array"> <general_error> <message>	Mensagem do erro

NOTAS

NOTA 1

A informação **available_payment_methods** permite que sejam enviados os códigos de meios de pagamento do TrayCheckout que poderão ser disponibilizados no processo de cobrança de transações que não forem aprovadas pela operadora (cartão) ou que não forem pagas (boleto). Os códigos deverão ser enviados separados por vírgula (,); Essa informação é opcional, e quando não for enviado o TrayCheckout permitirá que o comprador finalize a compra com todas as formas de pagamento disponíveis.

Essa campo é útil quando a loja deseja oferecer uma condição especial para finalização em uma determinada forma, como por exemplo Desconto de 10% no Boleto Bancário. Assim deve-se enviar apenas o código do boleto bancário, e o comprador não conseguirá trocar a forma de pagamento e finalizar a compra com cartão usufruindo-se do desconto no cartão.

6.2 Integração de Carrinho de Compras via API

O TrayCheckout fornece uma ferramenta poderosa para sites que não possuem um carrinho de compras e querem oferecer a possibilidade de visitantes escolherem produtos e finalizarem uma compra.

Esta ferramenta exclusiva também permite que seja possível o seu uso em um ambiente multi-vendedor, onde o site/marketplace pode fornecer um carrinho de compras para todas as lojas simultaneamente, fazendo com que o comprador tenha uma experiência única de compra com o TrayCheckout.

É possível inclusive utilizar um calculador de fretes baseando-se nos Correios ou em uma API própria de cada vendedor.

Para esta integração, deverá ser feito uso da API a seguir:

Endereço da API	
Ambiente de Testes	https://api.sandbox.traycheckout.com.br/edge/tmp_transactions/create
Ambiente de Produção	https://api.traycheckout.com.br/edge/tmp_transactions/create
Protocolo	Rest/HTTP

Os dados que podem ser enviados nessa API são:

Dados de Entrada	Obrigatório	Formato / Tamanho Max	Descrição
token_account	Sim	Texto /15	Token de identificação do vendedor
transaction_product[][code]	Obrigatório	Texto /20	Código do produto
transaction_product[][description]	Obrigatório	Texto /150	Descrição do produto
transaction_product[][quantity]	Obrigatório	Decimal / 11	Quantidade do produto. Formato: 0.00
transaction_product[][price_unit]	Obrigatório	Decimal / 11	Valor do produto. Formato: 0.00
transaction_product[][extra]		Texto /45	Informações extras do produto
transaction_product[][url_img]		Texto /255	URL da imagem do produto
transaction_product[][sku_code]		Texto /20	Código SKU do produto
transaction_product[][weight]		Inteiro / 11	Peso do produto em gramas
token_transaction		Texto /32	Token da transação. Veja NOTA 1
url_seller		Texto /255	URL do botão Voltar a Loja
url_notification		Texto /255	Url para NAS. Veja Capítulo 7
price_discount		Decimal / 11	Valor do desconto da transação
postal_code_seller		Texto /8	Cep de origem. Veja NOTA 2
shipping_type		Texto /15	Forma de envio. Veja NOTA 2
shipping_price		decimal	Valor do frete Veja NOTA 2
sub_store		Texto /20	Sub-Loja. Veja NOTA 3
url_css		Texto /255	URL do CSS customizado. Veja NOTA 4

Exemplo da inserção de 1 produto no carrinho de compras:

```
curl https://api.traycheckout.com.br/edge/tmp_transactions/create -d
  "token_account=TOKEN_DO_VENDEDOR
  &transaction_product[][code]=1
  &transaction_product[][description]=Notebook Preto
  &transaction_product[][quantity]=1
  &transaction_product[][price_unit]=1999.99
  &transaction_product[][extra]=HD 500 GB
  &postal_code_seller =040010001"
```

* *Necessário remover as quebras de linha (deixar inline) antes de executar o cURL.*

Exemplo de resposta com sucesso baseando no envio do exemplo acima:

```
<tmp_transaction>
  <message_response>
    <message>success</message>
  </message_response>
  <data_response>
    <token_transaction>b93b787058cb5cd6b80b6139cbca5342</token_transaction>
    <url_car>https://app.traycheckout.com.br/payment/car/v2</url_car>
    <transaction_products type="array">
      <transaction_product>
        <code>1</code>
        <img nil="true"/>
        <sku_code nil="true"/>
        <description> Notebook Preto </description>
        <extra> HD 500 GB </extra>
        <price_unit type="decimal">3.5</price_unit>
        <quantity type="decimal">1.0</quantity>
        <weight nil="true"/>
      </transaction_product>
    </transaction_products>
  </data_response>
</tmp_transaction>
```

Abaixo um detalhamento de cada nó do XML de resposta:

XML de Resposta	
<tmp_transaction>	Nó principal da resposta
<tmp_transaction> <message_response>	Nó que contém o resultado da resposta
<tmp_transaction> <message_response> <message>	Resposta sobre a solicitação Em caso de sucesso: sucess Em caso de erro: error
<tmp_transaction> <data_response>	Nó que contém os dados da resposta
<tmp_transaction> <data_response> <token_transaction>	Token da transação. Veja NOTA 1
<tmp_transaction> <data_response> <url_car>	Url para o comprador finalizar a compra com os produtos do carrinho
<tmp_transaction> <data_response> <transaction_products type="array">	Nó contendo os produtos que estão no carrinho

<pre><tmp_transaction> <data_response> <transaction_products type="array"> <transaction_product></pre>	Nó contendo os detalhes de um produto
<pre><tmp_transaction> <data_response> <transaction_products type="array"> <transaction_product> <code></pre>	Código do produto
<pre><tmp_transaction> <data_response> <transaction_products type="array"> <transaction_product> <sku_code nil="true"/></pre>	Código do Sku
<pre><tmp_transaction> <data_response> <transaction_products type="array"> <transaction_product> <description></pre>	Descrição do produto
<pre><tmp_transaction> <data_response> <transaction_products type="array"> <transaction_product> <extra></pre>	Informação extra do produto
<pre><tmp_transaction> <data_response> <transaction_products type="array"> <transaction_product> <price_unit type="decimal"></pre>	Preço. Formato 0.00
<pre><tmp_transaction> <data_response> <transaction_products type="array"> <transaction_product> <quantity type="decimal"></pre>	Quantidade. Formato 0.00
<pre><tmp_transaction> <data_response> <transaction_products type="array"> <transaction_product> <weight nil="true"/></pre>	Peso do produto.

NOTAS

NOTA 1

Token da Transação será a forma de identificar um carrinho de compras. Assim, quando ele não for enviado, será criado um novo carrinho. Para que o produto enviado seja adicionado a um carrinho existe, basta enviar o token_transaction desse carrinho criado anteriormente.

NOTA 2

O carrinho do TrayCheckout pode calcular o frete com base nos valores dos correios, através de uma API do vendedor ou ainda apenas exibir valores pré calculados na loja. Para calcular o frete no carrinho, é necessário enviar a informação **postal_code_seller** contendo o CEP de origem da mercadoria.

Caso não for utilizar o cálculo de frete, poderá ainda exibir as informações já calculadas ou simplesmente reforçar que o frete é grátis, através dos campos **shipping_type** e **shipping_price**

Exemplo de opção com Frete grátis:

shipping_type= Frete Grátis

shipping_price= 0.00

NOTA 3

O parâmetro **sub_loja** permite enviar uma informação específica para ser utilizada no portal do cliente, de forma a distinguir transações de um mesmo vendedor que trabalha com linhas diferentes de produtos, ou ainda lojas diferentes com a mesma conta TrayCheckout

NOTA 4

É possível informar uma url para customizar a página do carrinho, alterando as cores e posicionamentos através de css.

7. NAS – Notificação Automática de Status

O TrayCheckout pode comunicar com sua aplicação a cada alteração de status de uma transação, fazendo com que seu sistema acompanhe todo o fluxo de status e esteja sempre atualizado quanto a situação da transação.

É necessário configurar uma url que receberá a chamada, processará os dados recebidos, e em seguida irá obter mais detalhes da transação através de uma API específica. Esse parâmetro está disponível em ambas as integrações (POST ou API) e chama-se **url_notification**.

A notificação será realizada através de um POST, que espera receber o retorno “HTTP 200” de sua aplicação. Caso a página esteja fora do ar ou com algum outro erro e não retorne “HTTP 200”, serão feitas novas tentativas a cada 12 horas durante 3 dias.



O TrayCheckout irá realizar o POST com as seguintes informações:

Post de Notificação Automática de Status

token_transaction	Token da transação
-------------------	--------------------

Após receber esta informação, deverá ser feita a consulta pelas informações completas da transação, através do método de consulta da transação pelo token:

Endereço da API

Ambiente de Testes	https://api.sandbox.traycheckout.com.br/v2/transactions/get_by_token
Ambiente de Produção	https://api.traycheckout.com.br/v2/transactions/get_by_token
Protocolo	Rest/HTTP

Esta API deve receber as seguintes informações:

Dados de Entrada	Obrigatório	Formato / Tamanho Max	Descrição
token_account	Sim	Texto / 15	Token de identificação do vendedor
token_transaction	Sim	Texto / 32	Token de identificação da transação

Abaixo um exemplo de retorno da api de consulta de transação:

```
<response>
  <message_response>
    <message>success</message>
  </message_response>
  <data_response>
    <transaction>
      <order_number>89568398</order_number>
      <free> </free>
      <transaction_id type="integer">64997</transaction_id>
      <status_name>Reprovada</status_name>
      <status_id type="integer">89</status_id>
      <token_transaction>d2c4c6fa6272f4c5f9156185602e9b08</token_transaction>
      <payment>
        <price_payment type="decimal">24009.0</price_payment>
        <payment_response>Autorização Negada</payment_response>
        <url_payment/>
        <tid>1233</tid>
        <split type="integer">1</split>
        <payment_method_id type="integer">4</payment_method_id>
        <payment_method_name>Mastercard</payment_method_name>
        <linha_digitavel nil="true"/>
      </payment>
      <customer>
        <name>Nome do Cliente</name>
        <cpf>12312312332</cpf>
        <email>nomedo@cliente.com.br</email>
        <addresses type="array">
          <address>
            <street>Av Paulista</street>
            <number>123</number>
            <neighborhood>Jardim Paulistano</neighborhood>
            <postal_code>04001000</postal_code>
            <completion>Sala 1020 </completion>
            <city>São Paulo</city>
            <state>SP</state>
          </address>
        </addresses>
        <contacts type="array">
          <contact>
            <value>1434065858</value>
            <type_contact>H</type_contact>
          </contact>
        </contacts>
      </customer>
    </transaction>
  </data_response>
</response>
```

Abaixo um detalhamento de cada nó do XML de resposta:

XML de Resposta	
<response>	Nó principal da resposta
<response> <message_response>	Nó que contém o resultado da resposta
<response> <message_response> <message>	Resposta sobre a solicitação Em caso de sucesso: sucess Em caso de erro: error
<response> <data_response>	Nó que contém os dados da resposta
<response> <data_response> <transaction>	Nó que contém as informações sobre a transação
<response> <data_response> <transaction> <order_number>	Número do pedido enviado pela loja na solicitação
<response> <data_response> <transaction> <free>	Texto livre enviado pela loja na solicitação
<response> <data_response> <transaction> <transaction_id type="integer">	Id da transação
<response> <data_response> <transaction> <status_name>	Descrição do status da transação
<response> <data_response> <transaction> <status_id type="integer">	Id do status da transação
<response> <data_response> <transaction> <token_transaction>	Token da transação. Deve ser armazenado para ser utilizado nas consultas posteriores.
<response> <data_response> <transaction> <payment>	Nó com informações sobre o pagamento
<response> <data_response> <transaction> <payment> <price_payment type="decimal">	Preço pago pelo cliente
<response> <data_response> <transaction> <payment> <payment_response>	Resposta de pagamento da operadora para ser exibida na página de recibo de compra

<pre><response> <data_response> <transaction> <payment> <url_payment></pre>	Url para exibir o boleto ou ambiente externo do banco. Apenas para formas de pagamento que não são fechadas online, como por exemplo o Boleto Bancário e Itau Shoptline
<pre><response> <data_response> <transaction> <payment> <tid></pre>	Número único que identifica a transação na operadora
<pre><response> <data_response> <transaction> <payment> <split type="integer"></pre>	Número de parcelas da transação
<pre><response> <data_response> <transaction> <payment> <payment_method_id type="integer"></pre>	Id da forma de pagamento escolhida
<pre><response> <data_response> <transaction> <payment> <payment_method_name></pre>	Nome da forma de pagamento escolhida
<pre><response> <data_response> <transaction> <payment> <linha_digitavel nil="true"/></pre>	Linha digitável para pagamento via Internet Banking. Apenas no caso da escolha por Boleto Bancário
<pre><response> <data_response> <transaction> <customer></pre>	Nó com informações do comprador
<pre><response> <data_response> <transaction> <customer> <name></pre>	Nome do comprador
<pre><response> <data_response> <transaction> <customer> <cpf></pre>	Cpf do comprador
<pre><response> <data_response> <transaction> <customer> <email></pre>	Email do comprador
<pre><response> <data_response> <transaction> <customer> <addresses type="array"></pre>	Nó com array de endereços utilizados

<pre> <response> <data_response> <transaction> <customer> <addresses type="array"> <address> </pre>	Nó de um endereço
<pre> <response> <data_response> <transaction> <customer> <addresses type="array"> <address> <street> </pre>	Logradouro do Comprador
<pre> <response> <data_response> <transaction> <customer> <addresses type="array"> <address> <number> </pre>	Número
<pre> <response> <data_response> <transaction> <customer> <addresses type="array"> <address> <neighborhood> </pre>	Bairro
<pre> <response> <data_response> <transaction> <customer> <addresses type="array"> <address> <postal_code> </pre>	Cep
<pre> <response> <data_response> <transaction> <customer> <addresses type="array"> <address> <completion> </pre>	Complemento
<pre> <response> <data_response> <transaction> <customer> <addresses type="array"> <address> <city> </pre>	Cidade
<pre> <response> <data_response> <transaction> <customer> <addresses type="array"> <address> <state> </pre>	Estado

<pre> <response> <data_response> <transaction> <customer> <contacts type="array"> </pre>	Array com telefones utilizados
<pre> <response> <data_response> <transaction> <customer> <contacts type="array"> <contact> </pre>	Nó com telephone
<pre> <response> <data_response> <transaction> <customer> <contacts type="array"> <contact> <value> </pre>	Número do Telefone
<pre> <response> <data_response> <transaction> <customer> <contacts type="array"> <contact> <type_contact> </pre>	Tipo do contato: H - Residencial (Home) M - Celular (Mobile) W - Trabalho (Work)

8. Módulos para Softwares de Terceiros

O TrayCheckout possui módulos prontos para alguns sistemas de terceiros. Esses módulos são desenvolvidos internamente ou através de parceiros, e possuem diversos testes para garantir a sua compatibilidade com as versões abaixo indicadas.

Abaixo são disponibilizados os links para os manuais dos módulos disponíveis:

- Módulo Magento: <https://static.traycheckout.com.br/pdf/TrayCheckout-Modulos-Magento-v1.0.pdf>
- Módulo PrestaShop: <https://static.traycheckout.com.br/pdf/TrayCheckout-Modulos-Prestashop-v1.0.pdf>

9. Tabelas Auxiliares

Algumas das APIs do TrayCheckout retornam códigos específicos para alguns “tipos” de dados. Abaixo listamos os códigos disponíveis.

Tipos de Conta	
1	Pessoal
3	Empresarial

Tipos de Endereço	
B	Cobrança
D	Entrega

Sexo	
F	Feminino
M	Masculino

Estado Civil	
S	Solteiro
M	Casado
A	Separado
D	Divorciado
W	Viúvo

Status da Transação	
4	Aguardando Pagamento
5	Em Processamento
6	Aprovada
7	Cancelada
24	Em Contestação
87	Em Monitoramento
88	Em Recuperação
89	Reprovada

Formas de Pagamento	
2	Diners
3	Visa
4	Mastercard
5	American Express

6	Boleto Bancário
7	Itaú Shopline
14	Peela
15	Discover
16	Elo

Contato	
H	Residencial
M	Celular
W	Trabalho