Stripe

 Your Account

- Your Dashboard
- Account Settings

- Dashboard
- Documentation
- Help & Support

Documentation

- Getting Started
- Embedded Form
- Custom Forms
- Mobile Apps
- Charging Cards
- Sending Transfers
- Your Account

References

- Stripe.js
- Checkout
- Webhooks
- Fraud Protection
- Testing
- Examples
- API Upgrades
- API Libraries
- File Upload Guide
- Full API Reference

Subscriptions

- Overview
- Getting Started
- Integration Guide

Connect

- Overview
- Getting Started
- Integrating OAuth
- Collecting Fees
- Shared Customers

- Reference

Disputes

- Overview
- Submitting Evidence
- Dispute Types
- FAQ

FAQ

- Getting Paid
- SSL
- Security

More

- Contact
- Global Users
- Gallery
- Integrations

Checkout

Checkout is the best payment flow, on web and mobile. Checkout builds on top of Stripe.js to provide your users with a streamlined, mobile-ready payment experience that is constantly improving. Learn more about Checkout

Features

- Custom logo and text
- One-click payment through Remember Me
- Desktop, tablet, and mobile support

Demo

Try the demo below with this test card number:
`4242 4242 4242 4242`

Pay with Card

Integration

You can integrate Checkout in as little as a single line of client-side code. As we release new Stripe features, we'll automatically roll them out to your existing Checkout integration, so that you will always be using our latest technology without needing to change a thing.

Checkout supports two different integrations:

- **Simple**: The *simple* integration provides a blue "Pay with card" button and submits your payment form with a Stripe token in a hidden input field.
- **Custom**: The *custom* integration lets you create a custom button and passes a Stripe token to a JavaScript callback.

SimpleCustom

The *simple* integration uses a `<script>` tag inside your payment `<form>` to render the blue Checkout button. When a user clicks the button and completes payment, we will submit your form with a `stripeToken` along with any other `<input>`s in your form.

```
<form action="/charge" method="POST">
  <script
    src="https://checkout.stripe.com/checkout.js" class="stripe-button"
    data-key="pk_test_4KY2NoAENBbzSQF8BRey8TcY"
    data-image="/square-image.png"
    data-name="Demo Site"
    data-description="2 widgets ($20.00)"
    data-amount="2000">
  </script>
</form>
```

It's worth noting that Checkout doesn't actually create charges—it only creates tokens. You can use those tokens to create the actual charge on your server. Alternatively you can save the card for charging later, or sign the user up for recurring charges.

Received Parameters

These parameters will be submitted to your form's action endpoint, along with any other `<input>` fields in the form, once the checkout is complete.

Parameter	Description
`stripeToken`	The ID of the token you need to create a charge or a customer.
`stripeEmail`	The email address the user entered during the Checkout process.

The *custom* integration allow you to use your own custom button with our JavaScript API. This permits any HTML element or JavaScript event to start a Checkout payment.

When your page loads, you should create a handler object using `StripeCheckout.configure()`. You can call `open()` on the handler in response to any event. If you need to abort the Checkout process—for example, when navigation occurs in a single-page application—you can call `close()` on the handler. The `key` parameter, and all callback parameters (e.g. `token`), should be passed to `configure()`. Any other options can be passed to either `configure()` or `open()`.

Here is an example of a custom Checkout integration using jQuery.

```
<script src="https://checkout.stripe.com/checkout.js"></script>

<button id="customButton">Purchase</button>

<script>
```

```
  var handler = StripeCheckout.configure({
    key: 'pk_test_4KY2NoAENBbzSQF8BRey8TcY',
    image: '/square-image.png',
    token: function(token) {
      // Use the token to create the charge with a server-side script.
      // You can access the token ID with `token.id`
    }
  });

  $('#customButton').on('click', function(e) {
    // Open Checkout with further options
    handler.open({
      name: 'Demo Site',
      description: '2 widgets ($20.00)',
      amount: 2000
    });
    e.preventDefault();
  });

  // Close Checkout on page navigation
  $(window).on('popstate', function() {
    handler.close();
  });
</script>
```

Configuration Options

Required

Option	Description
`data-key`	Your publishable key (test or live).
`token`*Only available with the custom integration*	The callback to invoke when the Checkout process is complete. `function(token)` `token` is the token object created. `token.id` can be used to create a charge or customer. `token.email` contains the email address entered by the user.

Highly recommended

Option	Description
`data-image`	A relative URL pointing to a square image of your brand or product. The recommended minimum size is 128x128px.
`data-name`	The name of your company or website.
`data-description`	A description of the product or service being purchased.
`data-amount`	The amount (in cents) that's shown to the user. Note that you will still have to explicitly include it when you create a charge using the Stripe API.

Optional

Option	Description

`data-currency`	The currency of the amount (3-letter ISO code). The default is USD.
`data-panel-label`*panelLabel*	The label of the payment button in the Checkout form (e.g. “Subscribe”, “Pay {{amount}}”, etc.). If you include `{{amount}}`, it will be replaced by the provided amount. Otherwise, the amount will be appended to the end of your label.
`data-zip-code`*zipCode*	Specify whether Checkout should validate the billing ZIP code (true or false). The default is false.
`data-email`	If you already know the email address of your user, you can provide it to Checkout to be pre-filled.
`data-label`*Only available with the simple integration*	The text to be shown on the default blue button.
`data-allow-remember-me`*allowRememberMe*	Specify whether to include the option to "Remember Me" for future purchases (true or false). The default is true.
`opened`*Only available with the custom integration*	`function()` The callback to invoke when Checkout is opened (not supported in IE6 and IE7).
`closed`*Only available with the custom integration*	`function()` The callback to invoke when Checkout is closed (not supported in IE6 and IE7).

More Information

Is Checkout verifying if credit cards are valid?

Yes, Checkout verifies card details with the credit card networks to ensure they are valid. For additional protection, you can opt to have Checkout collect the billing ZIP code, and make sure that ZIP code validation is turned on for your account.

What browsers does Checkout support?

Checkout strives to support all known browsers. If you have an issue with Checkout on a specific browser, please contact us so we can improve its support.

How do I prevent the Checkout popup from being blocked?

You need to call `handler.open` when the user clicks on an element on the page and not in a callback. This indicates to the browser that the user is explicitly requested the popup. Otherwise, mobile devices and some versions of Internet Explorer will block the popup and prevent users from checking out.

```
// This will work
$("#button").on("click", function() {
  handler.open({
    image: '/square-image.png',
    name: 'Demo Site',
    description: '2 widgets ($20.00)',
    amount: 2000
  });
});
```

```
// This will not work
$("#failbutton").on("click", function() {
  $.ajax({
    url: "/",
  }).done(function() {
    handler.open({
      image: '/square-image.png',
      name: 'Demo Site',
      description: '2 widgets ($20.00)',
      amount: 2000
    });
  });
});
```

HTTPS For Your Site

All submissions of payment info using Checkout are made via a secure HTTPS connection. However, in order to protect yourself from certain forms of man-in-the-middle attacks, we suggest that you also serve the page containing the payment form with HTTPS as well. This means that any page that a Checkout form may exist on should start with `https://` rather than just `http://`.

If you are not familiar with the process of buying SSL certificates and integrating them with your server to enable a secure HTTPS connection, please visit our Help Page for SSL.

Guides

We have Checkout integration guides for several platforms and languages to get you up and running as quickly as possible: Sinatra, Rails, Flask and PHP

Questions?

We're always happy to help with code or other questions you might have! Check out our answers to common questions or chat live with other developers in #stripe on freenode.

United States

Select your Country

- Australia
- Canada
- United Kingdom
- Ireland
- United States
- Austria
- Belgium
- Denmark
- Finland
- France
- Germany

- Italy
- Luxembourg
- Netherlands
- Norway
- Spain
- Sweden
- Switzerland