Sending Emails with JavaScript

ON **OCTOBER 15, 2019** / BY **ARTUR HEBDA**

JavaScript is a programming language that you can use for both front-end and back-end development. When the name JavaScript is used in the context of sending emails, Node.js is the first thing that comes to mind. We even blogged about how to send emails with Node.js. In this article, we want to change the perspective from the server-side to the client-side. Let's figure out how you can use JS to send emails from the app that has no back-end.

Can I send emails with JS or not?

You can't send emails using JavaScript code alone due to lack of support for server sockets. For this, you need a server-side language that talks to the SMTP server. You can use JS in conjunction with a server script that will send emails from the browser based on your requests. This is the value we're going to introduce below.

Table of Contents [hide]

Why you might want to send emails with JS

Traditionally, the server-side of a regular app is responsible for sending emails. You will need to set up a server using back-end technology. The client-side sends a request to the server-side, which creates an email and sends it to the SMTP server. If you're curious about what happens with an email after that, read our blog post, SMTP relay.

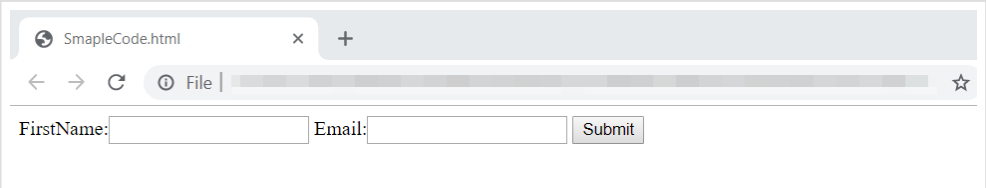
So, why would anyone be willing to go another way and send emails right from the client-side using JavaScript? Such an approach is quite useful for building contact forms or other kinds of user interaction on web apps, which allows your app to send an email without refreshing the page the user is interacting with. Besides, you don't have to mess around with coding a server. This is a strong argument if your web app uses email sending for contact forms only. Below, you will find a few options on how to make your app send emails from the client-side.

`mailto:` for sending form data

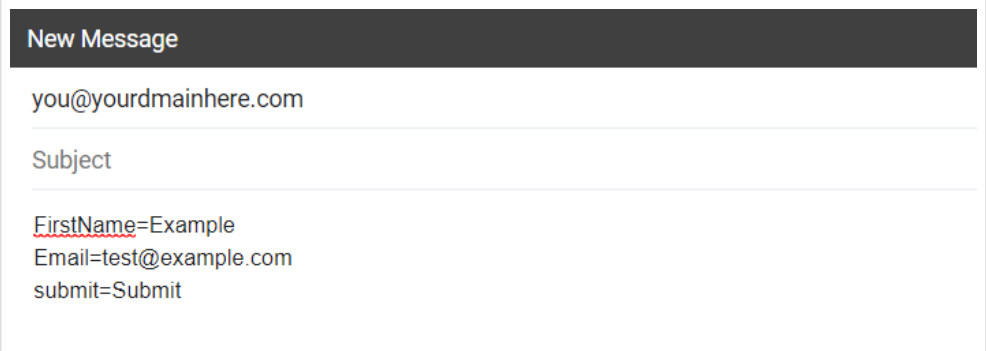
Since you can't send an email directly with JS, you can tell the browser to open a default mail client to do so. Technically, the `mailto:` method does not send email directly from the browser, but it can do the job. Check out how the following code example works:

```
<form action="mailto:you@yourdmainhere.com" method="post" enctype="text/plain
FirstName:<input type="text" name="FirstName">
Email:<input type="text" name="Email">
<input type="submit" name="submit" value="Submit">
</form>
```

When you launch it in the browser, you'll see the following:



Once the data has been submitted, the browser opens a default mail client. In our case, this is Gmail.



The `mailto:` method is a rather easy solution to implement but it has some specific drawbacks:

- You can't control the layout of the data since the data is submitted in the form sent by the browser.
- `mailto:` doesn't protect your email address from being harvested by spambots. Some time ago, this could be mitigated by constructing a link in JS. These days, more and more bots run JS and do not rely on HTML rendered by the server alone.

SmtpJS.com – true email sending from JavaScript

SmtpJS is a free library you can use for sending emails from JavaScript. All you need is an SMTP server and a few manipulations to get things done. We'll use Mailtrap.io as the server because it is an actionable solution for email testing. Below is the flow you should follow:

```
<script src="https://smtpjs.com/v3/smtp.js">
</script>
```

- Create a button that will trigger the JavaScript function

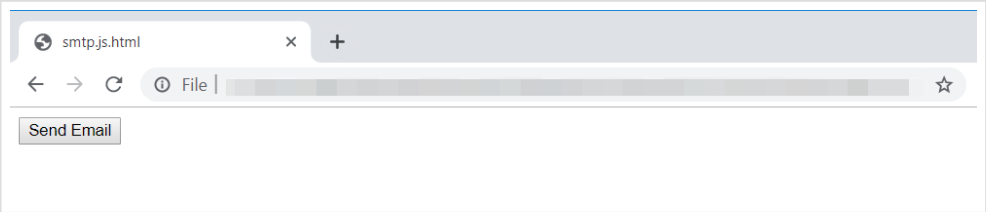
```
<input type="button" value="Send Email" onclick="sendEmail()">
```

- Write the JS function to send emails via SmtpJS.com.

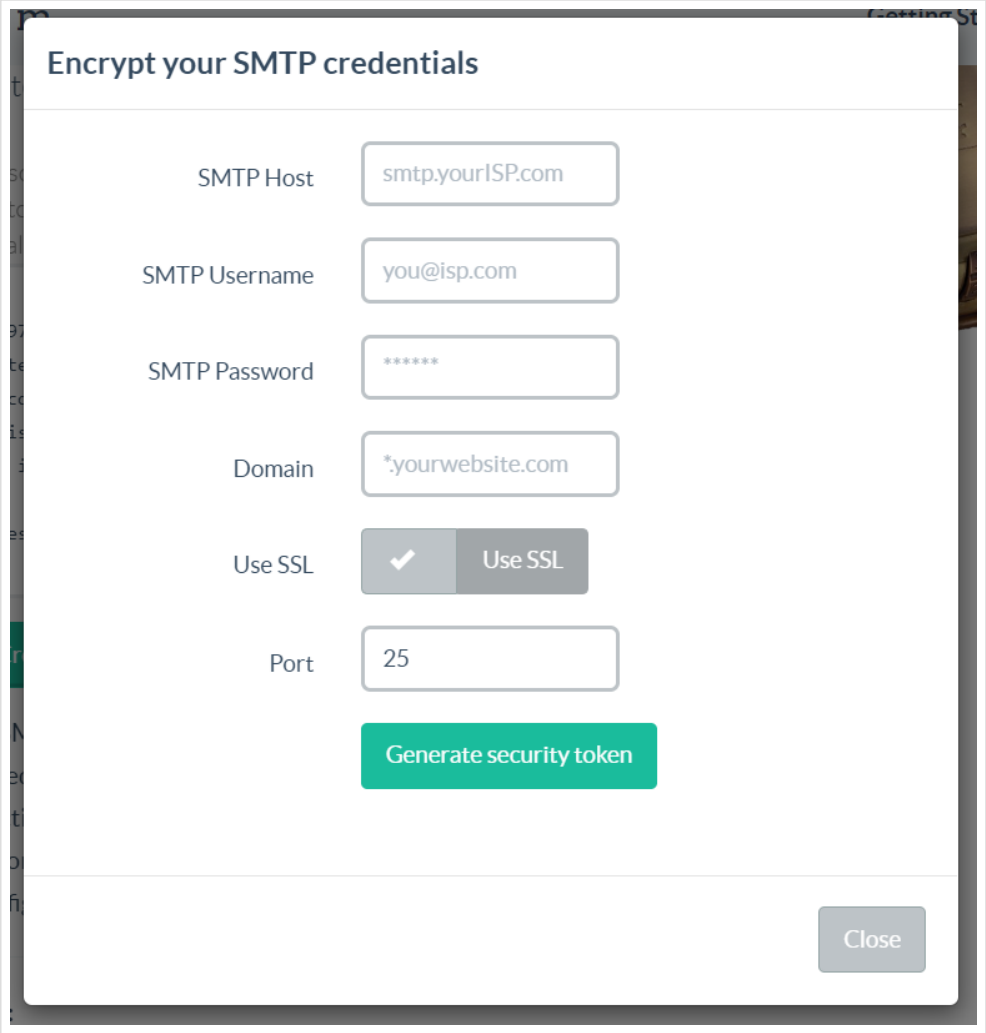
```
function sendEmail() {
Email.send({
    Host : "smtp.mailtrap.io",
    Username : "<Mailtrap username>",
    Password : "<Mailtrap password>",
    To : 'recipient@example.com',
    From : "sender@example.com",
    Subject : "Test email",
    Body : "<html><h2>Header</h2><strong>Bold text</strong><br></br><em>Itali
}).then(
  message => alert(message)
);
}
```

If you have multiple recipients, you can specify an array of email addresses in the `To` : property.

- Run *test.html* in the browser and send your email



The drawback with the code sample above is that your username and password are visible in the client-side script. This can be fixed if you utilize the



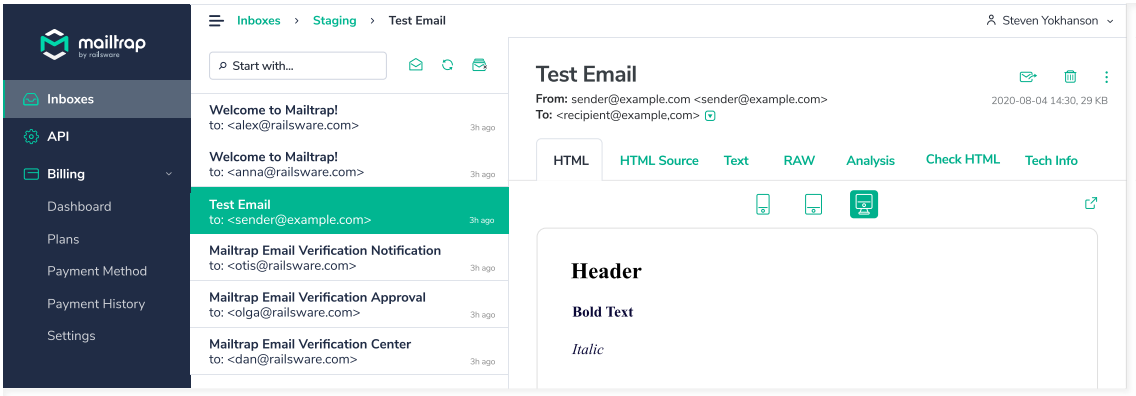
After that, press **Generate security token** and use it then in your JS function instead of SMTP server settings like the following:

```
SecureToken : "<your generated token>"
```

Below you'll find code samples for an HTML email and an email with attachment. Thanks Mailtrap, you can also see how they will look in the recipients inbox. For more about what you can do with this service, read the Mailtrap Getting Started Guide.

Code sample for sending an HTML email

```
<script>
function sendEmail() {
    Email.send({
    SecureToken : "<your generated token>",
    To : 'recipient@example.com',
    From : "sender@example.com",
    Subject : "Test Email",
    Body : "<html><h2>Header</h2><strong>Bold text</strong><br></br><em>Itali
    }).then(
        message => alert("mail sent successfully")
    );
}
</script>
```

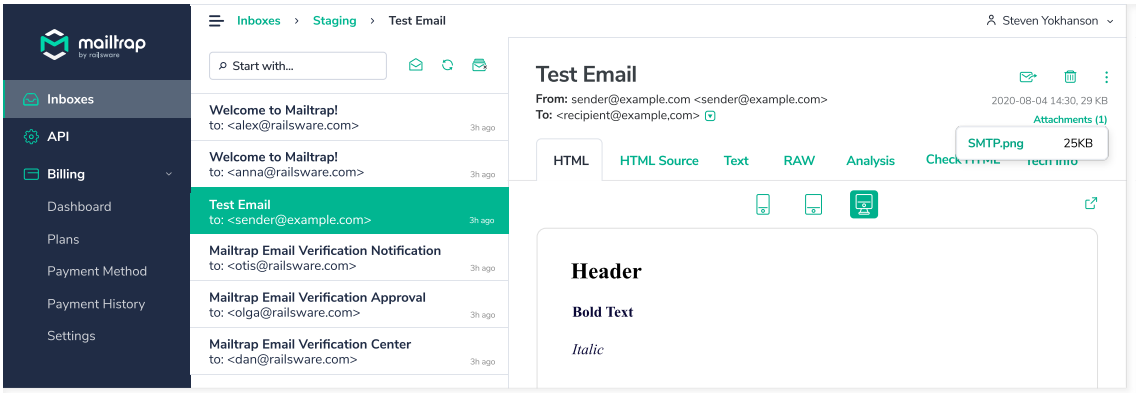


Code sample for sending an email with attachments

For sending an email with an attachment, use the `Attachments` property, as follows:

```
<script src="https://smtpjs.com/v3/smtp.js"></script>
<input type="button" value="Send Email" onclick="sendEmail()">
<script>
function sendEmail() {
```

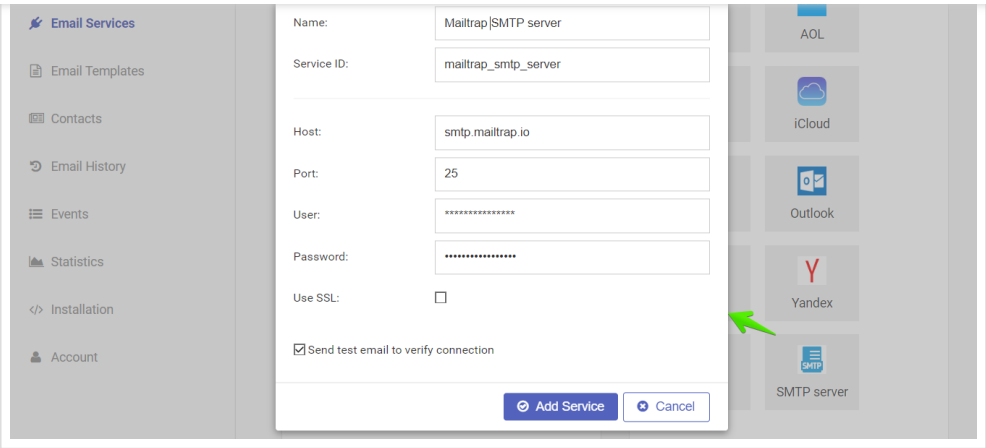
```
    To : 'recipient@example.com',
    From : "sender@example.com",
    Subject : "Test Email",
    Body : "<html><h2>Header</h2><strong>Bold text</strong><br></br><em>Itali
    Attachments : [
    {
        name : "smtp.png",
        path : "https://…/smtp.png"
    }]
    }).then(
        message => alert(message)
    );
}
</script>
```



EmailJS

EmailJS.com was already featured in our blog post, Sending Emails with ReactJS. This service allows you to connect your email service, build an email template and send it from JavaScript without any server code. Let's check out the scope.

- Create an account and choose an email service to connect with. There are the popular transactional services options available, such as Amazon SES or Mailgun, as well as personal services like Gmail or Outlook. You can also add a custom SMTP server. That's what we're going to do since we use Mailtrap.



- Create an email template using the built-in editor. The editor provides plenty of options for content building and other useful features, such as auto-reply, reCAPTCHA verification, and more. It's also necessary to understand the basics of coding your own HTML email template. For this, read our Guide on How to Build HTML Email. Once this is done, click **Save**.

One of the major benefits of EmailJS.com is that the typical email attributes are hidden. The template includes the recipient field and it cannot be overridden from JS, so you send the template you have configured previously.

- Now you need to install EmailJS SDK. This can be done with npm:

```
npm install emailjs-com --save
```

or bower

```
bower install emailjs-com --save
```

If you need to use EmailJS on your website, paste the following code before closing tag:

```
<script type="text/javascript"
        src="https://cdn.jsdelivr.net/npm/emailjs-com@2.4.0/dist/email.min.js
</script>
```

```
        emailjs.init("YOUR_USER_ID"); //use your USER ID
    })();
</script>
```

- The actual email sending can be carried out via two methods: `emailjs.send` or `emailjs.sendForm`. Here are the code examples for both of them:

emailjs.send

```
var templateParams = {
    name: 'James',
    notes: 'Check this out!'
};

emailjs.send('YOUR_SERVICE_ID', 'YOUR_TEMPLATE_ID', templateParams) //use you
    .then(function(response) {
       console.log('SUCCESS!', response.status, response.text);
    }, function(error) {
       console.log('FAILED...', error);
    });
```

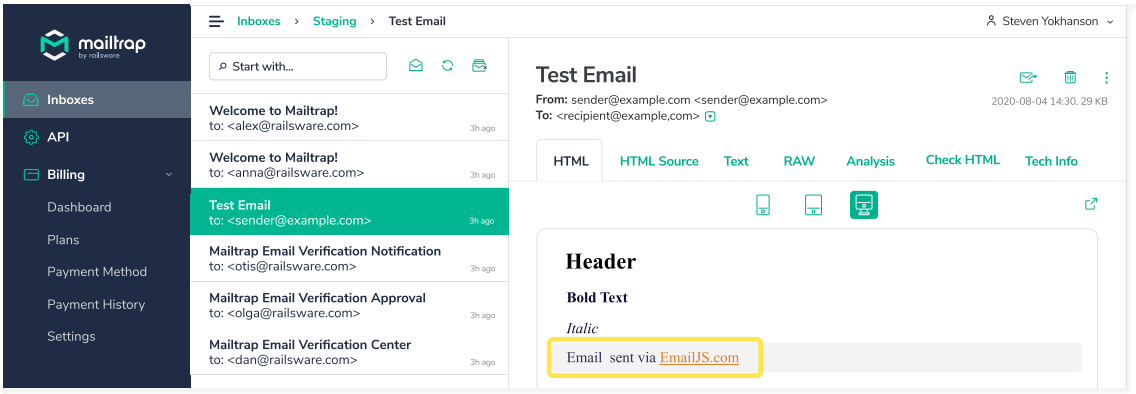
emailjs.sendForm

```
var templateParams = {
    name: 'James',
    notes: 'Check this out!'
};

emailjs.sendForm('YOUR_SERVICE_ID', 'YOUR_TEMPLATE_ID', templateParams) //use
    .then(function(response) {
       console.log('SUCCESS!', response.status, response.text);
    }, function(error) {
       console.log('FAILED...', error);
    });
```

Let's check whether EmailJS.com works by sending an email straight from the browser. We've set up an email service, Mailtrap, and built a simple template. Now, we need to create an HTML file with the following code:

```
<script type="text/javascript"
        src="https://cdn.jsdelivr.net/npm/emailjs-com@2.4.0/dist/email.min.js
</script>
<script type="text/javascript">
   (function(){
      emailjs.init("<USER-ID>"); //Insert your User ID
   })();
</script>
<script>
    var templateParams = {
    name: 'Sender',
    notes: 'Test email'
};

emailjs.send('<email-service-ID>', '<email-template-ID>', templateParams) //I
    .then(function(response) {
       console.log('SUCCESS!', response.status, response.text);
    }, function(error) {
       console.log('FAILED...', error);
    });
</script>
```

Run it in the browser and check out the Mailtrap Demo Inbox. It works!



EmailJS offers a free subscription plan that allows you to send up to 200 emails per month using only two templates. In addition, you'll have a limited list of contacts and email size (up to 50Kb). Higher quotas are available for paid subscriptions: Personal ($5/month), Professional ($15/month), and Business ($50/month).

To wrap up

Although sending emails is a server-side thing, you can refrain from dealing with server-side coding. SmtpJS and EmailJS are two actionable solutions to streamline your front-end and make it able to send emails. EmailJS is better because it hides typical email attributes and you are able to send whatever templates you have configured before. The mailto: method is different but it can also be useful in some cases. If you, however, have changed your mind and are willing to set up back-end for your app, check out one of our dedicated blog posts:

- How to send emails with Python
- How to send emails with Ruby
- How to send emails with PHP
- How to send emails with ASP.NET C#

Add comment

Name*

E-mail*

Subscribe

Submit

railsware

PRODUCTS

Jira Smart Checklist

Notemate.co

Pivotal Booster

EXPERIENCE

Open Source

Railsware Blog

MAILTRAP

Pricing

Changelog

Status

Terms of Service

Privacy Policy

Navigational Information

DPA

CONTACT

email: support[at]mailtrap.io