**Send Email us emailjs**

**Link:** [**https://www.npmjs.com/package/emailjs**](https://www.npmjs.com/package/emailjs)

**emailjs**

send emails, html and attachments (files, streams and strings) from node.js to any smtp server

**INSTALLING**

npm install emailjs

**FEATURES**

* works with SSL and TLS smtp servers
* supports smtp authentication ('PLAIN', 'LOGIN', 'CRAM-MD5', 'XOAUTH2')
* emails are queued and the queue is sent asynchronously
* supports sending html emails and emails with multiple attachments (MIME)
* attachments can be added as strings, streams or file paths
* supports utf-8 headers and body

**REQUIRES**

* auth access to an SMTP Server
* if your service (ex: gmail) uses two-step authentication, use an application specific password

**EXAMPLE USAGE - text only emails**

var email  **=** require("./path/to/emailjs/email"); //require("emailjs/email");

var server  **=** email.server.connect({

   user**:**    "username",  //abc@gmail.com

   password**:**"password",  //abc

   host**:**    "smtp.your-email.com",  //smtp.gmail.com

   ssl**:**     true

});

*// send the message and get a callback with an error or details of the message that was sent*

server.send({

   text**:**    "i hope this works",

   from**:**    "you <username@your-email.com>",

   to**:**      "someone <someone@your-email.com>, another <another@your-email.com>",

   cc**:**      "else <else@your-email.com>",

   subject**:** "testing emailjs"

}, function(err, message) { console.log(err **||** message); });

**EXAMPLE USAGE - html emails and attachments**

var email  **=** require("./path/to/emailjs/email");

var server  **=** email.server.connect({

   user**:** "username",

   password**:**"password",

   host**:** "smtp.your-email.com",

   ssl**:** true

});

var message **=** {

   text**:** "i hope this works",

   from**:** "you <username@your-email.com>",

   to**:** "someone <someone@your-email.com>, another <another@your-email.com>",

   cc**:** "else <else@your-email.com>",

   subject**:** "testing emailjs",

   attachment**:**

   [

      {data**:**"<html>i <i>hope</i> this works!</html>", alternative**:**true},

      {path**:**"path/to/file.zip", type**:**"application/zip", name**:**"renamed.zip"}

   ]

};

*// send the message and get a callback with an error or details of the message that was sent*

server.send(message, function(err, message) { console.log(err **||** message); });

*// you can continue to send more messages with successive calls to 'server.send',*

*// they will be queued on the same smtp connection*

*// or you can create a new server connection with 'email.server.connect'*

*// to asynchronously send individual emails instead of a queue*

**EXAMPLE USAGE - sending through hotmail/outlook**

var email  **=** require("./path/to/emailjs/email");

var server  **=** email.server.connect({

   user**:** "username",

   password**:**"password",

   host**:** "smtp-mail.outlook.com",

   tls**:** {ciphers**:** "SSLv3"}

});

var message **=** {

   text**:** "i hope this works",

   from**:** "you <username@outlook.com>",

   to**:** "someone <someone@your-email.com>, another <another@your-email.com>",

   cc**:** "else <else@your-email.com>",

   subject**:** "testing emailjs",

   attachment**:**

   [

      {data**:**"<html>i <i>hope</i> this works!</html>", alternative**:**true},

      {path**:**"path/to/file.zip", type**:**"application/zip", name**:**"renamed.zip"}

   ]

};

*// send the message and get a callback with an error or details of the message that was sent*

server.send(message, function(err, message) { console.log(err **||** message); });

**EXAMPLE USAGE - attaching and embedding an image**

var email  **=** require("./path/to/emailjs/email");

var server  **=** email.server.connect({

   user**:** "username",

   password**:**"password",

   host**:** "smtp-mail.outlook.com",

   tls**:** {ciphers**:** "SSLv3"}

});

var message **=** {

   text**:** "i hope this works",

   from**:** "you <username@outlook.com>",

   to**:** "someone <someone@your-email.com>, another <another@your-email.com>",

   cc**:** "else <else@your-email.com>",

   subject**:** "testing emailjs",

   attachment**:**

   [

      {data**:** "<html>i <i>hope</i> this works! here is an image: <img src='cid:my-image' width='100' height ='50'> </html>"},

      {path**:**"path/to/file.zip", type**:**"application/zip", name**:**"renamed.zip"},

      {path**:**"path/to/image.jpg", type**:**"image/jpg", headers**:**{"Content-ID"**:**"<my-image>"}}

   ]

};

*// send the message and get a callback with an error or details of the message that was sent*

server.send(message, function(err, message) { console.log(err **||** message); });

**API**

**email.server.connect(options)**

// options is an object with the following keys

options =

{

user // username for logging into smtp

password // password for logging into smtp

host // smtp host

port // smtp port (if null a standard port number will be used)

ssl // boolean or object {key, ca, cert} (if true or object, ssl connection will be made)

tls // boolean or object (if true or object, starttls will be initiated)

timeout // max number of milliseconds to wait for smtp responses (defaults to 5000)

domain // domain to greet smtp with (defaults to os.hostname)

authentication // array of preferred authentication methods ('PLAIN', 'LOGIN', 'CRAM-MD5', 'XOAUTH2')

}

**email.server.send(message, callback)**

// message can be a smtp.Message (as returned by email.message.create)

// or an object identical to the first argument accepted by email.message.create

// callback will be executed with (err, message)

// either when message is sent or an error has occurred

**message**

// headers is an object ('from' and 'to' are required)

// returns a Message object

// you can actually pass more message headers than listed, the below are just the

// most common ones you would want to use

headers =

{

text // text of the email

from // sender of the format (address or name <address> or "name" <address>)

to // recipients (same format as above), multiple recipients are separated by a comma

cc // carbon copied recipients (same format as above)

bcc // blind carbon copied recipients (same format as above)

subject // string subject of the email

attachment // one attachment or array of attachments

}

**email.SMTP.authentication**

associative array of currently supported SMTP authentication mechanisms

**attachment**

// can be called multiple times, each adding a new attachment

// options is an object with the following possible keys:

options =

{

// one of these fields is required

path // string to where the file is located

data // string of the data you want to attach

stream // binary stream that will provide attachment data (make sure it is in the paused state)

// better performance for binary streams is achieved if buffer.length % (76\*6) == 0

// current max size of buffer must be no larger than Message.BUFFERSIZE

// optionally these fields are also accepted

type // string of the file mime type

name // name to give the file as perceived by the recipient

charset // charset to encode attatchment in

method // method to send attachment as (used by calendar invites)

alternative // if true, will be attached inline as an alternative (also defaults type='text/html')

inline // if true, will be attached inline

encoded // set this to true if the data is already base64 encoded, (avoid this if possible)

headers // object containing header=>value pairs for inclusion in this attachment's header

related // an array of attachments that you want to be related to the parent attachment

}

**Authors**

eleith

**Testing**

npm install -d

npm test

**Contributions**

issues and pull requests are welcome

**Keywords**

none