Package 'blastula'

July 19, 2018

Type Package

Title Easily Send HTML Email Messages

Version 0.2.1

Description Compose and send out responsive HTML email messages that render perfectly across a range of email clients and device sizes. Messages are composed using 'Markdown' and a text interpolation system that allows for the injection of evaluated R code within the message body, footer, and subject line. Helper functions let the user insert embedded images, web link buttons, and 'ggplot2' plot objects into the message body. Messages can be sent through an 'SMTP' server or through the 'Mailgun' API service <http://mailgun.com/>.

License MIT + file LICENSE

URL https://github.com/rich-iannone/blastula

BugReports https://github.com/rich-iannone/blastula/issues

Depends R (>= 3.2.1)

Imports commonmark (>= 1.5), downloader (>= 0.4), dplyr (>= 0.7.5), ggplot2 (>= 2.2.1), glue (>= 1.2.0), htmltools (>= 0.3.6), httr (>= 1.3.1), magrittr (>= 1.5), stringr (>= 1.3.1), tidyr (>= 0.8.0)

Suggests covr, testthat

Encoding UTF-8

LazyData true

RoxygenNote 6.0.1

NeedsCompilation no

Author Richard Iannone [aut, cre] (<https://orcid.org/0000-0003-3925-190X>)

Maintainer Richard Iannone <riannone@me.com>

Repository CRAN

Date/Publication 2018-07-19 20:40:15 UTC

R topics documented:

`add_cta_button` *Helper function for adding a CTA button*

Description

Add a call to action button inside the body of the email with this helper function. There are options to specify the button text, the URL, and the button's alignment.

Usage

```
add_cta_button(url, text, align = "center")
```

Arguments

`url`	a URL for the button.
`text`	the text that is placed atop the CTA button.
`align`	the alignment of the button inside the main content area. Options are `center` (the default), `left`, and `right`.

Value

a character object with an HTML fragment that can be placed inside the message body wherever the CTA button should appear.

Examples

```
# Create the button as an HTML fragment
cta_button <-
  add_cta_button(
    url = "http://www.website.net",
    text = "Press This Button")

# Include the button in the email
# message body by simply referencing
# the `cta_button` object
email <-
  compose_email(
    body = "
    Hello!

    Below is a call. It's a call \\
    to action. Press it!

    {cta_button}

    Cheers
    ")
```

`add_ggplot` *Helper function for adding an ggplot*

Description

Add an ggplot plot inside the body of the email with this helper function.

Usage

```
add_ggplot(plot_object, width = 5, height = 5)
```

Arguments

`plot_object`	the ggplot plot object.
`width`	the width of the output plot in inches.
`height`	the height of the output plot in inches.

Value

a character object with an HTML fragment that can be placed inside the message body wherever the plot image should appear.

Examples

```
library(ggplot2)

# Create a ggplot plot
plot <-
  ggplot(
    data = mtcars,
    aes(x = disp, y = hp,
        color = wt, size = mpg)) +
  geom_point()

# Create an HTML fragment that
# contains an the ggplot as an
# embedded plot
plot_html <-
  add_ggplot(
    plot_object = plot,
    height = 5,
    width = 7)

# Include the plot in the email
# message body by simply referencing
# the `plot_html` object
email <-
  compose_email(
  body = "
  Hello!

  Here is a very important plot \\
  that will change the way you \\
  look at cars forever.

  {plot_html}

  So useful, right?
  ") %>% preview_email()
```

---

add_image *Helper function for adding an image*

Description

Add a local image inside the body of the email with this helper function.

Usage

```
add_image(file)
```

Arguments

file	a path to an image file.

Value

a character object with an HTML fragment that can be placed inside the message body wherever the image should appear.

Examples

```
# Create an HTML fragment that
# contains an image
img_file_path <-
  system.file(
    "img",
    "test_image.png",
    package = "blastula")

img_file_html <-
  add_image(
    file = img_file_path)

# Include the image in the email
# message body by simply referencing
# the `img_file_html` object
email <-
  compose_email(
    body = "
    Hello!

    Here is an image:

    {img_file_html}
    ")
```

---

add_readable_time *Helper function for adding a humanized date/time*

Description

Add a nicely-formatted date/time string inside the body of the email with this helper function. This will insert the current date/time/tz based on the caller's locale information at the time of the call. There are options to specify whether the date, time, and time zone parts are to be included.

Usage

```
add_readable_time(use_date = TRUE, use_time = TRUE, use_tz = TRUE)
```

Arguments

use_date	a logical value that indicates whether the current date should be included.
use_time	a logical value that indicates whether the current time should be included.
use_tz	a logical value that indicates whether the locale's time zone should be included.

Value

a character object that can be placed inside any message component message wherever the function is called.

---

blast_first *Setup function to get SMTP mail working effectively*

Description

Run this function first to obtain SMTP mailing functionality, which is important for sending out email.

Usage

```
blast_first()
```

---

compose_email *Create the email message*

Description

Create an email message. String interpolation is possible for the text comprising the email body, footer, and preheader text. This is done by using curly braces to enclose R code chunks. Variables can be specified in the function call (using named arguments with . . .), and any variables not found in . . . will be searched for in the global environment.

Usage

```
compose_email(body = NULL, footer = NULL, .preheader_text = NULL,
  .title = NULL, .envir = parent.frame(), ...)
```

Arguments

body	the main body of text for the email message. Markdown can be used here (along with string interpolation via curly braces and named arguments) to construct the main text.
footer	the footer text for the email message. As with the body, Markdown and string interpolation can be used here.
.preheader_text	text that appears before the subject in some email clients. This must be plaintext.
.title	the title of the email message. This is not the subject but the HTML title text which may appear in limited circumstances.
.envir	allows for setting the environment.
...	expression strings for string interpolation for the body, footer, and preheader_text string data.

Value

an email_message object, which can be used for previewing with the preview_email() function or for sending out actual emails with the send_email_out() function.

Examples

```
# Create a simple email message using
# Markdown formatting
email <-
  compose_email(
  body = "
  Hello!

  ## This a section heading

  We can use Markdown formatting \\
  to **embolden** text or to add \\
  *emphasis*. This is exciting, \\
  right?

  Cheers")

# The email message can always be
# previewed using `preview_email()`
preview_email(email = email)

# We can use string interpolation to
# add in R code or strings assigned
# to variables; variables can be
# obtained from the global workspace
# or from temporary variables in the
# function call
sender_name <- "Mike"
```

```
email <-
  compose_email(
  body = "
  Hello!

  I just wanted to let you \\
  know that the {thing} that \\
  asked me for is ready to \\
  pick up. So, come over and \\
  do that.

  Cheers,

  {sender_name}",
  thing = "report")
```

`create_email_creds_file` *Create a file with email access credentials*

Description

Create a file with access credentials for the purpose of automatically emailing notification messages.

Usage

```
create_email_creds_file(user, password, provider = NULL, host = NULL,
  port = NULL, sender = NULL, use_ssl = TRUE, use_tls = FALSE,
  authenticate = TRUE, creds_file_name = NULL)
```

Arguments

`user`	the username for the email account.
`password`	the password associated with the email account.
`provider`	an optional provider email provider with which an STMP account is available. Options currently include `gmail`, `outlook`, and `office365`. If nothing is provided then values for `host`, `port`, `sender`, `use_ssl`, `use_tls`, and `authenticate` are expected.
`host`	the `host` name.
`port`	the port number.
`sender`	the sender name.
`use_ssl`	an option as to whether to use SSL; supply a `TRUE` or `FALSE` value (`TRUE` is the default value).
`use_tls`	a logical value to indicate whether to use TLS; supply a `TRUE` or `FALSE` value (`FALSE` is the default value).

authenticate — an option as to whether to authenticate; supply a TRUE or FALSE value (TRUE is the default value).

creds_file_name — an option to specify a name for the credentials file. If no name is provided, one will be automatically generated. The autogenerated file will be invisible and have its name constructed in the following way: .bls_<host_name>.

Examples

```
## Not run:
# Create a credentials file to facilitate
# the sending of email messages
create_email_creds_file(
  user = "user_name@gmail.com",
  password = "************",
  provider = "gmail")

## End(Not run)
```

---

get_html_str *Get the HTML content of an email message*

Description

Get the HTML content string from an email_message object as a single-length character vector.

Usage

```
get_html_str(message)
```

Arguments

message — the email message object, as created by the compose_email() function. The object's class is email_message

Value

a character object containing the email message's HTML content.

prepare_test_message *Prepare a email test message object*

Description

Create an email test message object, which is helpful for sending a test message with the `send_email_out()` function.

Usage

```
prepare_test_message()
```

Examples

```
## Not run:
# Create a credentials file to
# send via Gmail (this will be named
# `.bls_smtp_gmail_com`)
create_email_creds_file(
  user = "username@gmail.com",
  password = "**************",
  provider = "gmail",
  sender = "Sender Name")

# Send oneself a test message to
# test these new SMTP settings
send_email_out(
  message = prepare_test_message(),
  sender = "Sender Name",
  subject = "test 2",
  from = "username@gmail.com",
  to = "username@gmail.com",
  creds_file = ".bls_smtp_gmail_com")

## End(Not run)
```

preview_email *Preview an email message*

Description

Preview an HTML email in the Viewer before sending it out.

Usage

```
preview_email(email)
```

Arguments

email	an email_message object.

---

send_by_mailgun *Send an email message through the Mailgun API*

Description

Send an email message via the Mailgun API. This requires an account with Mailgun.

Usage

```
send_by_mailgun(message, subject = NULL, from, recipients, url, api_key)
```

Arguments

message	the email message object, as created by the compose_email() function. The object's class is email_message
subject	the subject of the email.
from	the email address of the sender. This does not have to be the same email that is associated with the account actually sending the message.
recipients	a vector of email addresses.
url	the URL for the sending domain.
api_key	the API key registered to the Mailgun service.

Examples

```
## Not run:
# Create a simple email message using
# Markdown formatting
email <-
  compose_email(
  body = "
  Hello!

  ## This a section heading

  We can use Markdown formatting \\
  to **embolden** text or to add \\
  *emphasis*. This is exciting, \\
  right?

  Cheers")

# Generate a vector of recipients
recipient_list <-
  c("person_1@site.net",
```

```
    "person_2@site.net")

# Send it to multiple people through
# the Mailgun API
email %>%
  send_by_mailgun(
    subject = "Sent through Mailgun",
    from = "The Sender <sender@send.org>",
    recipients = recipient_list,
    url = "<..mailgun_sending_domain..>",
    api = "<..mailgun_api_key..>")

## End(Not run)
```

`send_email_out` *Send an email message*

Description

Send an email message to one or more recipients.

Usage

```
send_email_out(message, from, to, subject = NULL, cc = NULL, bcc = NULL,
  attachments = NULL, attach_mime_types = NULL, attach_encodings = NULL,
  attach_dispositions = NULL, creds_file = NULL, sender = NULL,
  host = NULL, port = NULL, user = NULL, password = NULL,
  use_ssl = TRUE, use_tls = FALSE, authenticate = TRUE, ehlo = FALSE,
  verbose = FALSE, debug = FALSE)
```

Arguments

`message`	the email message object, as created by the `compose_email()` function. The object's class is `email_message`
`from`	the email address of the sender. This does not have to be the same email that is associated with the account actually sending the message.
`to`	a vector of email addresses serving as primary recipients for the message. For secondary recipients, use the `cc` and `bcc` arguments.
`subject`	the subject of the message, which is usually a brief summary of the topic of the message.
`cc`	a vector of email addresses for sending the message as a carbon copy. This list of for those who are to receive a copy of a message addressed primarily to another. The list of recipients in the CC list is visible to all other recipients of the message.
`bcc`	a vector of email addresses for sending the message as blind carbon copies. Any email addresses provided here will receive the message and these email addresses will be concealed from other recipients (including others on the BCC list).

attachments — a vector of paths to files to be attached to the email.

attach_mime_types — an optional vector of mime types to use for each of the attachments specified in attachments. If not provided, mime types will be assigned based on file extensions.

attach_encodings — an optional vector of encoding types to use for each of the attachments specified in attachments. Options are base64, 7bit, 8bit, or none.

attach_dispositions — an optional vector of disposition types for each of the attachments specified in attachments. Options are inline and attachment.

creds_file — an optional path to an email credentials file. This file must be created by the create_email_creds_file() function.

sender — the sender name.

host — the email host.

port — the port associated with the email account.

user — the username associated with the email account.

password — the password associated with the email account.

use_ssl — a logical value to indicate whether to use SSL.

use_tls — a logical value to indicate whether to use TLS.

authenticate — a logical value to indicate whether to use authentication.

ehlo — a logical value to indicate whether to force an EHLO command after connection to the SMTP host.

verbose — a logical value indicating whether verbose messages should be printed to the console during sending of email.

debug — a logical value to indicate whether the mail sending statement should be printed to the console. No emails are sent when debug is set to TRUE.

%>% *The magrittr pipe*

Description

blastula uses the pipe function, %>% to turn function composition into a series of imperative statements.

Index