Email clients info for Linux

Git

These days most developers use `git send-email` instead of regular email clients. The man page for this is quite good. On the receiving end, maintainers use `git am` to apply the patches.

If you are new to `git` then send your first patch to yourself. Save it as raw text including all the headers. Run `git am raw_email.txt` and then review the changelog with `git log`. When that works then send the patch to the appropriate mailing list(s).

General Preferences

Patches for the Linux kernel are submitted via email, preferably as inline text in the body of the email. Some maintainers accept attachments, but then the attachments should have content-type `text/plain`. However, attachments are generally frowned upon because it makes quoting portions of the patch more difficult in the patch review process.

Email clients that are used for Linux kernel patches should send the patch text untouched. For example, they should not modify or delete tabs or spaces, even at the beginning or end of lines.

Don't send patches with `format=flowed`. This can cause unexpected and unwanted line breaks.

Don't let your email client do automatic word wrapping for you. This can also corrupt your patch.

Email clients should not modify the character set encoding of the text. Emailed patches should be in ASCII or UTF-8 encoding only. If you configure your email client to send emails with UTF-8 encoding, you avoid some possible charset problems.

Email clients should generate and maintain References: or In-Reply-To: headers so that mail threading is not broken.

Copy-and-paste (or cut-and-paste) usually does not work for patches because tabs are converted to spaces. Using xclipboard, xclip, and/or xcutsel may work, but it's best to test this for yourself or just avoid copy-and-paste.

Don't use PGP/GPG signatures in mail that contains patches. This breaks many scripts that read and apply the patches. (This should be fixable.)

It's a good idea to send a patch to yourself, save the received message, and successfully apply it with 'patch' before sending patches to Linux mailing lists.

Some email client (MUA) hints

Here are some specific MUA configuration hints for editing and sending patches for the Linux kernel. These are not meant to be complete software package configuration summaries.

Legend:

- TUI = text-based user interface
- GUI = graphical user interface

Alpine (TUI)

Config options:

In the :menuselection:`Sending Preferences` section:

- :menuselection:`Do Not Send Flowed Text` must be `enabled`
- :menuselection:`Strip Whitespace Before Sending` must be `disabled`

When composing the message, the cursor should be placed where the patch should appear, and then pressing :kbd:`CTRL-R` let you specify the patch file to insert into the message.

Claws Mail (GUI)

Works. Some people use this successfully for patches.

To insert a patch use :menuselection:`Message-->Insert` File (:kbd:`CTRL-I`) or an external editor.

If the inserted patch has to be edited in the Claws composition window "Auto wrapping" in :menuselection:`Configuration-->Preferences-->Compose-->Wrapping` should be disabled.

Evolution (GUI)

Some people use this successfully for patches.

When composing mail select: Preformat

from :menuselection:`Format-->Paragraph Style-->Preformatted` (:kbd:`CTRL-7`) or the toolbar

Then use: :menuselection:`Insert-->Text File...` (:kbd:`ALT-N x`) to insert the patch.

You can also `diff -Nru old.c new.c | xclip`, select :menuselection:`Preformat`, then paste with the middle button.

Kmail (GUI)

Some people use Kmail successfully for patches.

The default setting of not composing in HTML is appropriate; do not enable it.

When composing an email, under options, uncheck "word wrap". The only disadvantage is any text you type in the email will not be word-wrapped so you will have to manually word wrap text before the patch. The easiest way around this is to compose your email with word wrap enabled, then save it as a draft. Once you pull it up again from your drafts it is now hard word-wrapped and you can uncheck "word wrap" without losing the existing wrapping.

At the bottom of your email, put the commonly-used patch delimiter before inserting your patch: three hyphens (`---`).

Then from the :menuselection:`Message` menu item, select insert file and choose your patch. As an added bonus you can customise the message creation toolbar menu and put the :menuselection:`insert file` icon there.

Make the composer window wide enough so that no lines wrap. As of KMail 1.13.5 (KDE 4.5.4), KMail will apply word wrapping when sending the email if the lines wrap in the composer window. Having word wrapping disabled in the Options menu isn't enough. Thus, if your patch has very long lines, you must make the composer window very wide before sending the email. See: https://bugs.kde.org/show_bug.cgi?id=174034

You can safely GPG sign attachments, but inlined text is preferred for patches so do not GPG sign them. Signing patches that have been inserted as inlined text will make them tricky to extract from their 7-bit encoding.

If you absolutely must send patches as attachments instead of inlining them as text, right click on the attachment and select properties, and highlight :menuselection:`Suggest automatic display` to make the attachment inlined to make it more viewable.

When saving patches that are sent as inlined text, select the email that contains the patch from the message list pane, right click and select :menuselection:`save as`. You can use the whole email unmodified as a patch if it was properly composed. There is no option currently to save the email when you are actually viewing it in its own window -- there has been a request filed at kmail's bugzilla and hopefully this will be addressed. Emails are saved as read-write for user only so you will have to chmod them to make them group and world readable if you copy them elsewhere.

Lotus Notes (GUI)

Run away from it.

Mutt (TUI)

Plenty of Linux developers use `mutt`, so it must work pretty well.

Mutt doesn't come with an editor, so whatever editor you use should be used in a way that there are no automatic linebreaks. Most editors have an :menuselection:`insert file` option that inserts the contents of a file unaltered.

To use `vim` with mutt:

```
set editor="vi"
```

If using xclip, type the command:

```
:set paste
```

before middle button or shift-insert or use:

```
:r filename
```

if you want to include the patch inline. (a)ttach works fine without `set paste`.

You can also generate patches with `git format-patch` and then use Mutt to send them:

```
$ mutt -H 0001-some-bug-fix.patch
```

Config options:

It should work with default settings. However, it's a good idea to set the `send_charset` to:

```
set send_charset="us-ascii:utf-8"
```

Mutt is highly customizable. Here is a minimum configuration to start using Mutt to send patches through Gmail:

```
# .muttrc
# ================  IMAP ====================
set imap_user = 'yourusername@gmail.com'
set imap_pass = 'yourpassword'
set spoolfile = imaps://imap.gmail.com/INBOX
set folder = imaps://imap.gmail.com/
set record="imaps://imap.gmail.com/[Gmail]/Sent Mail"
set postponed="imaps://imap.gmail.com/[Gmail]/Drafts"
set mbox="imaps://imap.gmail.com/[Gmail]/All Mail"

# ================  SMTP  ====================
set smtp_url = "smtp://username@smtp.gmail.com:587/"
set smtp_pass = $imap_pass
set ssl_force_tls = yes # Require encrypted connection

# ================  Composition  ====================
set editor = `echo \$EDITOR`
```

```
set edit_headers = yes  # See the headers when editing
set charset = UTF-8     # value of $LANG; also fallback for send_charset
# Sender, email address, and sign-off line must match
unset use_domain        # because joe@localhost is just embarrassing
set realname = "YOUR NAME"
set from = "username@gmail.com"
set use_from = yes
```

The Mutt docs have lots more information:

http://dev.mutt.org/trac/wiki/UseCases/Gmail

http://dev.mutt.org/doc/manual.html

Pine (TUI)

Pine has had some whitespace truncation issues in the past, but these should all be fixed now.

Use alpine (pine's successor) if you can.

Config options:

- `quell-flowed-text` is needed for recent versions
- the `no-strip-whitespace-before-send` option is needed

Sylpheed (GUI)

- Works well for inlining text (or using attachments).
- Allows use of an external editor.
- Is slow on large folders.
- Won't do TLS SMTP auth over a non-SSL connection.
- Has a helpful ruler bar in the compose window.
- Adding addresses to address book doesn't understand the display name properly.

Thunderbird (GUI)

Thunderbird is an Outlook clone that likes to mangle text, but there are ways to coerce it into behaving.

- Allow use of an external editor: The easiest thing to do with Thunderbird and patches is to use an "external editor" extension and then just use your favorite `$EDITOR` for reading/merging patches into the body text. To do this, download and install the extension, then add a button for it using :menuselection:`View-->Toolbars-->Customize...` and finally just click on it when in the :menuselection:`Compose` dialog.

  Please note that "external editor" requires that your editor must not fork, or in other words, the editor must not return before closing. You may have to pass additional flags or change the settings of your editor. Most notably if you are using gvim then you must pass the -f option to gvim by putting `/usr/bin/gvim -f` (if the binary is in `/usr/bin`) to the text editor field in :menuselection:`external editor` settings. If you are using some other editor then please read its manual to find out how to do this.

To beat some sense out of the internal editor, do this:

- Edit your Thunderbird config settings so that it won't use `format=flowed`. Go to :menuselection:`edit-->preferences-->advanced-->config editor` to bring up the thunderbird's registry editor.
- Set `mailnews.send_plaintext_flowed` to `false`

- Set `mailnews.wraplength` from `72` to `0`
- :menuselection:`View-->Message Body As-->Plain Text`
- :menuselection:`View-->Character Encoding-->Unicode (UTF-8)`

TkRat (GUI)

Works. Use "Insert file..." or external editor.

Gmail (Web GUI)

Does not work for sending patches.

Gmail web client converts tabs to spaces automatically.

At the same time it wraps lines every 78 chars with CRLF style line breaks although tab2space problem can be solved with external editor.

Another problem is that Gmail will base64-encode any message that has a non-ASCII character. That includes things like European names.