irchat

COLLABORATORS			
	TITLE : irchat		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		October 23, 2022	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

irchat

1.1 irchat

```
                  Welcome to the documentation of irchat.el, the GNU Emacs ↩
                     interface to
Internet Relay Chat. irchat.el provides all the necessary IRC features,
combined with the easy extensibility of GNU Emacs.


                Introduction
                What is irchat.el?

                Etiquette
                 How to behave in IRC.

                Overview
                 Why use IRC in GNU Emacs?

                Installing
                 How to get irchat.el up and running?

                Using
                   Using irchat.el

                Commands
                 IRC Commands in irchat.el

                Customizing
                 Customizing irchat.el

                Authors
                 Reporting bugs etc.
```

1.2 irchat/Introduction

```
Introduction
************

   What is irchat.el?

   Irchat.el is a GNU Emacs client for the Internet Relay Chat protocol.
Study the documentation accompanying the server. Make sure to read the
etiquette file. Irchat.el is designed to be easily modifiable if you
know emacs-lisp and understand the meaning of hooks.
Files that are needed to run irchat are: irchat.el, irchat-commands.el
and irchat-handle.el.
This document describes version 2.1 of irchat. It runs with servers up
to and including ircd2.6.1.
```

1.3 irchat/Etiquette

```
Etiquette
*********


/************************************************************************
 *   IRC - Internet Relay Chat, doc/etiquette  *   Copyright (C) 1990,
Lea Viljanen and Ari Husa  *  *   This program is free software; you
can redistribute it and/or modify  *   it under the terms of the GNU
General Public License as published by  *   the Free Software
Foundation; either version 1, or (at your option)  *   any later
version.  *  *   This program is distributed in the hope that it will
be useful,  *   but WITHOUT ANY WARRANTY; without even the implied
warranty of  *   MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
See the  *   GNU General Public License for more details.  *  *   You
should have received a copy of the GNU General Public License  *
along with this program; if not, write to the Free Software  *
Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.  */

   HOW TO BEHAVE ON IRC

   Authors:      Lea Viljanen (LadyBug)  viljanen@kreeta.helsinki.fi
           Ari Husa     (luru)     so-luru@tolsun.oulu.fi

   1) Language

   The most widely understood and spoken language on IRC is English.
However! As IRC is used in many different countries, English is by no
means the only language. If you want to speak some other language than
English (for example with your friends), go to a separate channel and
set the topic (with /topic) to indicate that. For example    /topic
Finnish only! would mean that this channel would be reserved for
Finnish discussion. On the other hand, you should check the topic (with
/list command) before you move to a channel to see if there are any
restrictions about language.    On a channel not restricted by /topic,
please speak a language everybody can understand. If you want to do
otherwise, change channels and set the topic accordingly.
```

2) Hello/Goodbye

It’s not necessary to greet everybody on a channel personally. Usually one "Hello" or equivalent is enough. And don’t expect everybody to greet you back. On a channel with 20 people that would mean one screenful of hellos. It’s sensible not to greet, in order not to be rude to the rest of the channel. If you must say hello, do it with a private /msg. The same applies to goodbyes.

3) Discussion

When you come to a new channel it’s advised you to listen for a while to get an impression of what’s discussed. Please feel free to join in, but do not try to force your topic into the discussion if that doesn’t come naturally.

4) |[]\

IRC has quite a lot of people from Scandinavian countries, the above characters are letters in their alphabet. This has been explained on IRC about a thousand and one times, so read the following, do not ask it on IRC:

is an A with 2 dots over it is an A with a small circle above it | is either an O with 2 dots over it or an O with a dash (/) through it [,], and \ are the preceding three letters in upper case.

There are a lot of people from Japan as well, who use Kanji characters which may look quite exotic as well. As I don’t know Kanji I don’t even try to explain any of the characters.

5) ATTENTION!

Remember, people on IRC form their opinions about you only by your actions, writings and comments on IRC. So think before you type. Do not "dump" to a channel or user (send large amounts of unwanted information). This is likely to get you /kicked off the channel or /killed off from irc. Dumping causes network ’burbs’, connections going down because servers cannot handle the large amount of traffic any more.

1.4 irchat/Overview

Overview
****

Why IRC in Emacs?

1. You can have other Emacs buffers visible and have irchat either in a small buffer or not visible at all. For example, you can read USENET news with GNUS without leaving irc, and be noticed if someone is trying to reach you. Nowadays when computing power is cheap you can have multiple emacses running so this view has lost some of its importance.

2. You get all the GNU Emacs features such as a calculator, all usable while in IRC. And you can run your favourite shell commands and get their output directly to IRC or to your input (*IRC Commands*) buffer for you to send to IRC channel.

3. You can edit other files while talking about their contents on IRC.

4. The commands for editing your text are those familiar from GNU Emacs and if you have customized them, the customized editing keys are automatically available. Irchat uses it's own keymap bound to C-c, so if you have used that key in your own keybindings you have to do some reordering in order to use your own commands.

5. You may easily add features if you find something missing (assuming you know elisp, of course, but this is a good reason to learn. ;)) Actually we are now (01-07.1991) working on a easily modifiable hooks to irchat (something like /ON on 'ircII' - one of the most popular irc clients).

6. GNU Emacs has been ported on many different UNIX machines, so you will not have to do the porting yourself, which might be the case with the traditional clients.

7. If you use IRC on several machines, you don't need to compile an IRC client for all of them. You can just take the small elisp code with you and use it.

8. GNU Emacs tries to handle most terminals, which might not be the case with the traditional clients.

1.5 irchat/Installing

```
Installing irchat.el
********************
```

Try to convince your local emacs administrator to put irchat.el to a globally accessible elisp directory and to add this info to the info tree. No modifications are needed in irchat.el itself, as long as the irc port is 6667 as it usually is and the server is local. If your server uses different irc port or you use non local irc server then you have to set them by setting environment variables IRCPORT and/or IRCSERVER.

You need add line containing following to your '.emacs', that way can start irchat by issuing command M-x irchat.

```
(autoload 'irchat "irchat" "GNUemacs IRC client." t)
```

If you are not happy with the way irchat.el is when shipped, you need to have a small piece of code for yourself for setting some variables that control irchat the way you want them to be. You may place them in your '.emacs' or in an other file to be loaded before starting irchat itself. See the section called "Customizing irchat.el"

for info on the customization possibilities.

1.6 irchat/Using

Using irchat.el
****

In irc-mode you will have two buffers, the command buffer and the dialogue buffer. You will be typing in the command buffer, and the dialogue will appear in the dialogue buffer. The command buffer roughly corresponds to the one-line area in the bottom in the traditional clients.

In the command buffer, the lines you type will be sent to the channel you are on when you press the return or the linefeed key. You may freely edit them or do any regular editing tricks that you may want to do. If the line exceeds the right margin, it will be wrapped around, so you can finish the line and send it when you're ready.

The cursor keys move you around in the buffer. If you want to say something again, just find it in the command buffer, move the cursor on it, and press the return key.

You can use the regular GNU Emacs commands in the command buffer for inserting text from elsewhere, like insert-file (C-x i) for adding other files, C-u M-! for inserting output of shell commands and so forth. Notice that these will NOT be sent to the channel before you go on each line and press return. You can send output of shell command to IRC by executing irchat-Command-exec (C-c !).

The commands for controlling IRC are available in the command buffer. See section "IRC Commands" for info.

The other buffer is the dialogue buffer. The conversation you have had in IRC will be stored there. Notice that this buffer is not saved anywhere unless you explicitly tell emacs to do so. This feature lets you see what has been spoken about on the channel while you have been away, for example. There is a number of commands for the dialogue buffer. There are also a couple of commands for controlling the dialogue buffer from the command buffer, namely C-c C-f, C-c DEL and C-c C-l. C-c C-f (or C-c F) will freeze the dialogue buffer so that you can read the text there and scroll the buffer. C-c DEL will scroll dialogue buffer one windowful back and C-c C-l will refresh the irchat buffers. To scroll dialogue forward one screenful you can use C-SPC or if you have only two buffers visible you can use the default emacs scroll-other-window M-C-v.

Notice that there will be a capital F on the modeline of the dialogue buffer if the dialogue buffer is frozen. Press C-c C-f again to unfreeze it. A capital A signifies that you've marked yourself as being AWAY.

In the dialogue buffer there is a special keymap, which doesn't allow for typing.
The commands there are:

```
   * t for tagging the line

   * o for switching buffers

   * f for freeze

   * DEL for scrolling up (backward)

   * space for scrolling down (forward)

   * $ for end of buffer

   * > for end of buffer

   * < for beginning of buffer

   * m for sending message to your selected channel

   * C-m for sending private message to user

   There is also a mode for debugging irchat, which you can use if you
want to familiarize yourself with the server-client traffic or think
that irchat is handling something wrong. You invoke it with C-c C-d and
end with another C-c C-d.

   CTCP (Client-To-Client-Protocol) is bound on prefix C-c C-c, it has
7 (seven) different commands:
   * v for version

   * u for userinfo

   * h for help

   * c for clientinfo

   * g for generic (ask any CTCP info, not nessessary on all clients)

   * U for setting userinfo, asks it from minibuffer

   * C-u for setting userinfo to contain current line from
     command-buffer
```

1.7 irchat/Commands

```
IRC Commands
************

There are two kinds of commands in IRC, namely client commands and
server commands. For using irchat.el it is useful to understand the
difference.  For example, the client command /CLEAR for clearing the
screen is used in the traditional clients. This command has no effect
outside your screen, and the other IRC-users need not be notified. On
the other hand, the server command /NICK changes your nickname. This
command affects everyone and is sent to everyone on the same channel
```

with you (and to all servers in the network, so don't change nick without need).

In irchat.el, all the server commands are directly available. If you have set irchat-want-traditional simply by typing a slash (/) at the beginning of a line and then typing the server command with the parameters or by typing the server command in *IRC Commands* buffer and then typing C-c c. This line will be sent directly to the server (no /QUOTE needed or allowed). Notice, however, that you need to type the commands in full. So you cannot use abbreviations like /N for /NAMES. Also notice that the /M (message to user) command is really called /PRIVMSG.

For most commands, however, there is a better way. We have written emacs commands for sending the line and bound them to keys. For example, for /LIST you just say C-c l, C-c n for /NICK, C-c C-n for /NAMES and so on.

Some of there commands require parameters and they'll prompt you for them. For example, the C-c n command will ask you for the new nickname you with to use. If you at this point decide that you didn't really want to change your nickname, press C-g (control-g) to cancel.

For some commands you may give an argument but you don't have to. For example, to see the users on channel 42, you type C-u 42 C-c w when just C-c w would show you all users. You may omit the channel number if you mean the current channel, that is C-u C-c w for users on the same channel with you. To ask users on named channels like '+glbf' or '#war' you can use C-u - C-c w, which asks the channel name in minibuffer. C-c n works in a similar fashion.

There exists a special /QUERY-like mode, in which every line you type will be sent to one person via a private message. You begin it with C-c 2 and specifying the person to chat with, and end with another C-c 2 with blank spaces as the name of the person.

There are two other ways for sending private messages, C-c p and C-c m. With C-c m you'll be asked for the name of the person and then for the text. You'll be typing the text in the minibuffer. The other way is to type the text in the command buffer, and to press C-c p to send it. irchat.el prompts you for the name of the person to send it to, the default being the name of the person you sent your last private message to. You can easily chat privately with two persons at a time by being in a C-c 2 with one and C-c p:ing with the other.

When you join a channel, completion is also available. C-c j +mTAB takes you to the channel +My_very_long_channel_name, if there happens to be such a channel. You can also join a channel where some person you know is, just use C-c j mta and you would be on channel where user mta is.

Notice that for C-c m, C-c p, C-c 2, C-c f (/WHOIS), and C-c k (/IGNORE) nickname completion is available. What this means is that you only need to type a few first letters of the nick, and you can use TAB or SPC to fill in the rest. This feature is also available in the command buffer, bound to C-i (TAB), where it completes the word you typed last assuming it is a nickname. This is handy for typing long

```
nicks so that you get the CaSe correctly. For example, to say
"BigCheese: ..." you just need to say "biTAB:", assuming nobody else
(on IRC at the time) has nick that starts with bi.
```

1.8 irchat/Customizing

```
Customizing irchat.el
*********************

   The simplest way to customize is through the defvars, that is,
variables. You can place these either in your `.emacs' or in a special
file. The most important defvars for getting irchat work the way you
want it to are irchat-beep-on-bells and irchat-command-window-on-top,
but there are many others.

   The most used irchat variables follow, setting other variables
requires deeper understanding of irchat and is not recommended.

   * irchat-want-traditional: nil
     Currently only used on determing whether the commands-window is
     placed on bottom and to force regognition of old irc commands that
     start with /.

   * irchat-server: from environment variable IRCSERVER
     Tells what irc server irchat tries to connect.

   * irchat-service: from environment variable IRCPORT or 6667
     Tells in what port the server resides.

   * irchat-nickname: from environment variable IRCNICK or your
     loginname
     What is your nickname in IRC.

   * irchat-startup-channel: nil
     Channel you try to join when starting irchat.

   * irchat-format-string: ">%s<"
     Format used when displaying your private messages to some person,
     person's name is inserted in place of %s.

   * irchat-format-string1 "=%s="
     Format used when displaying private messages to you from some
     person, person's name is inserted in place of %s.

   * irchat-format-string2 "<%s>"
     Format used when displaying messages to your primary channel,
     senders name is inserted in place of %s, sender is on this primary
     channel.

   * irchat-format-string3 "<%s:%s>"
     Format used when displaying messages to some of your secondary
     channel, senders name is inserted in place of first %s, channel
     the message has arrived in place of second %s, sender is on this
     secondary channel.
```

* irchat-format-string4 "(%s)"
  Format used when displaying messages to your primary channel, senders name is inserted in place of %s, sender is NOT on this primary channel.

* irchat-format-string5 "(%s:%s)"
  Format used when displaying messages to some of your secondary channel, senders name is inserted in place of first %s, channel the message has arrived in place of second %s, sender is NOT on this secondary channel.

* irchat-nam-suffix: ""
  What suffix is to be added to every line you send. Not recommended to set, can be very annoying.

* irchat-beep-on-bells: nil
  What to do with arriving bells, if nil don't ring bell (or flash). If value is 'always always ring a bell when it arrives, other values mean that only bell us when bell arrives in private message to us.

Variables for controlling CTCP (Client-To-Client-Protocol) and irchat's own file transmission protocol.

* irchat-client-userinfo: "No userinfo given."
  What is given as userinfo for people requesting it from you.

* irchat-file-accept: nil
  Do we want to accept file transmissions from users.

* irchat-file-confirm-save: t
  Do we want irchat to ask us before it writes the file in our directory.

If you're not happy with the way the keys are set up, you may wish to modify the key bindings. This is most conveniently done by defining them in your irchat-Command-mode-hook.

There are lots of other hooks, one for each server reply and so forth. For every servers message the irchat first tries to run function named irchat-handle- 'message' -hook, if that returns true it won't run internal handler for that message. The definition of these functions is somewhat tricky, you have to define it via 'defun' and then use 'setq' to set its variable-value as this function, see provided example for details. With these, an auto-greeter, for example, would be trivial to write. Here is two examples of using hooks. First we set up irchat-Command-mode-hook to read our abbreviations and require our own hook-code if it is not yet loaded and then we set up autogreeter. BTW, auto-greeters are annoying most of the time.

```
(setq irchat-Command-mode-hook
      '(lambda ()
         (make-variable-buffer-local 'blink-matching-paren)
         (setq blink-matching-paren nil)
         (setq local-abbrev-table irchat-mode-abbrev-table)
         (abbrev-mode 1)
         (require 'irchat-hooks)
```

```
          ))

    (setq irchat-join-hook
          '(lambda (prefix rest)
      (cond
    ;;; do not greet people coming to channel #report
       ((and (not (string= prefix irchat-nickname))
       (string= "#report" rest))
        t)
       ((and (string= "INSERT_YOUR_FAVOURITE_PERSON_HERE" prefix)
       (string= irchat-current-channel rest))
    ;;; send the greeting as private message
        (irchat-Command-message prefix (format "Greetings %s" prefix)))
       (t nil))))
```

1.9 irchat/Authors

```
irchat.history, reporting bugs etc.
***********************************

   The original irchat.el was written by Tor Lillqvist
<tml@hemuli.tik.vtt.fi>, but the irchat.el from those days is much
different from what it is now, irchat.el is highly ... eh ...  dynamic
right now.

   Thanks for developing are to nam <kny@niksula.hut.fi>, cletus
<jtp@niksula.hut.fi> for listening luru's <luru@tolsun.oulu.fi> and
Kaizzu's ideas and implementing them. Then after that Kaizzu
<kmk@cc.tut.fi> and mta <mta@cc.tut.fi> are responsible for developing
irchat to what it is now.

   There is, undoubtedly, lots of bugs and unintended features left in
both the code and the documentation. We welcome any suggestions and
really appreciate nice hacks and bug fixes.

   The bugs and comments should be send to irchat <irchat@cc.tut.fi>
which is mailing list for people interested in irchat. To join that
list send request to <irchat-request@cc.tut.fi>.

   This documentation was Kaizzu's first attempt ever to write
emacs-info -like text, and I probably failed miserably. Sorry.

   P.S. I (mta) am responsible for the last revision of this text, so
you can as well accuse me.
```