INP111 Homework 01 Week #7 (2022-10-27)

Date: 2022-10-27
Due Date: 2022-11-17

Minimized IRC Server (Daemon)

This homework is the extension of our previous lab, the chat server. Instead of working with a proprietary protocol, we aim to implement a server compliant with real-world clients running the IRC protocol. The IRC protocol is standardized in RFC 1459 (https://www.rfc-editor.org/rfc/rfc1459.html). Although it has been updated several times, we choose to follow the simplest (oldest) version to implement this homework.

> A complete IRC server can interconnect with other servers. To minimize the cost of implementing this homework, you do not need to handle interconnections with other servers.

The specification of this homework is much more complicated than our previous lab. Please read it carefully and ensure that your implementation works appropriately with our selected console-based IRC client, weechat (https://weechat.org/). The weechat client can be installed in Debian/Ubuntu Linux using the command `sudo apt install weechat` and in Mac OS using the command `brew install weechat` (You have to install `homebrew` (https://brew.sh/) package manager first).

Brief Descriptions of the Specification

You have to read RFC 1459 (https://www.rfc-editor.org/rfc/rfc1459.html) for the details of commands and response/error messages. Here we simply provide brief descriptions of the required commands and response/error messages you must implement in this homework.

Commands (received from a Client)

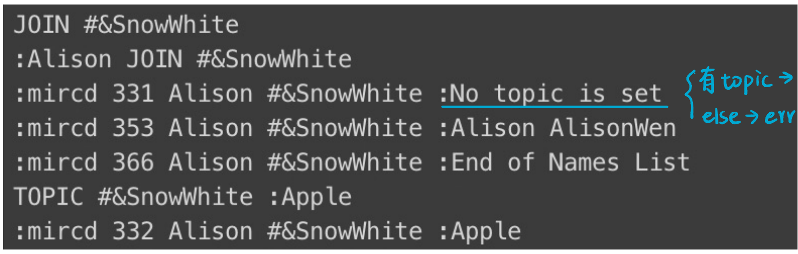
The required commands are listed below. Note that command arguments enclosed by brackets are optional.

從 client 傳入的指令.
↓

```
1 NICK <nickname>
2 USER <username> <hostname> <servername> <realname>
3 PING <server1> [<server2>]
4 LIST [<channel>]
5 JOIN <channel>
6 TOPIC <channel> [<topic>]
7 NAMES [<channel>]
8 PART <channel>
9 USERS
10 PRIVMSG <channel> <message>
11 QUIT
```

private msg

pair <nickname , pair< pss , pss>>



```
JOIN #&SnowWhite
:Alison JOIN #&SnowWhite
:mircd 331 Alison #&SnowWhite :No topic is set
:mircd 353 Alison #&SnowWhite :Alison AlisonWen
:mircd 366 Alison #&SnowWhite :End of Names List
TOPIC #&SnowWhite :Apple
:mircd 332 Alison #&SnowWhite :Apple
```

{有 topic → topic
else → errmsg.

channel <string, <string ,set<string>>

You can find the details of each command in Section 4 of RFC 1459 (https://www.rfc-editor.org/rfc/rfc1459.html).

> PING conmmand changed from PING <message> to PING <server1> [<server2>]
>
> To simplify the implementation, you can always repond PONG to the PING message **without** checking the availability of the target server. Respond **ERR_NOORIGIN** if no server is given.

Additional remarks for the commands listed above are summarized as follows. These remarks might make your implementation simpler.

- A `<nickname>` cannot contain spaces.
- A `<channel>` must be prefixed with a pond (#) symbol, e.g., `#channel`.
- The last parameter must be prefixed with a colon (':').
- By default, there is no response from `NICK` and `USER` commands unless there are errors. But the server must send the message of the day (motd) to a client once both commands are accepted.
- `JOIN` a channel is always a success. If the channel does not exist, your server should automatically create the channel.
- For successful `JOIN` and `PART` commands, the server must respond `:<nickname> <the command & params received from the server>` first, followed by the rest of the responses.
- `PRIVMSG` can be only used to send messages into a channel in our implementation. You do not have to implement sending a message to a user privately.

Server Messages (sent from the Server)

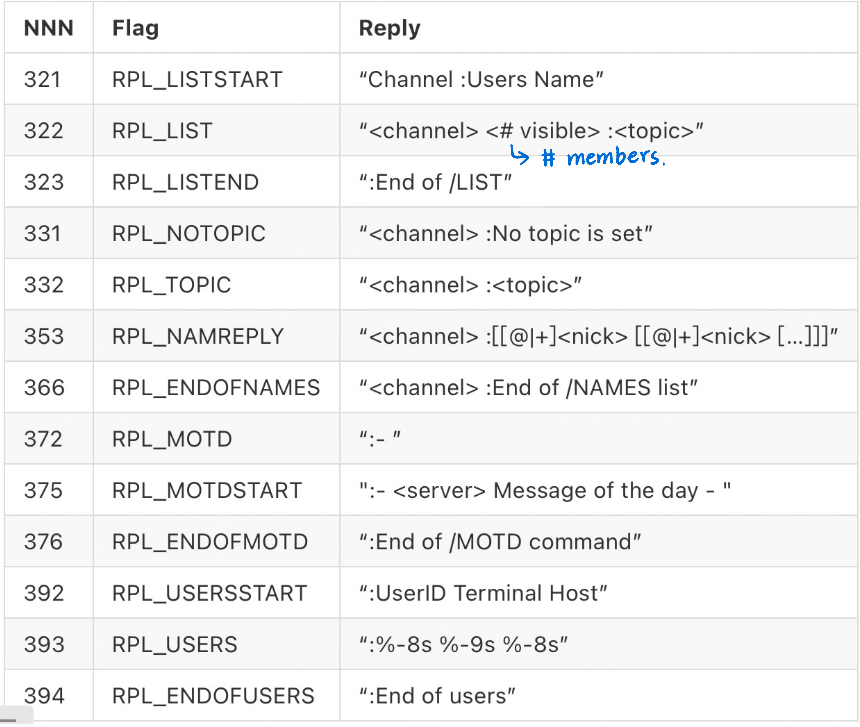
The general form of a server responded message is:

從 server 要 write 出去的 string.

```
:<prefix> NNN [<nickname>] [<params> ... ]
```

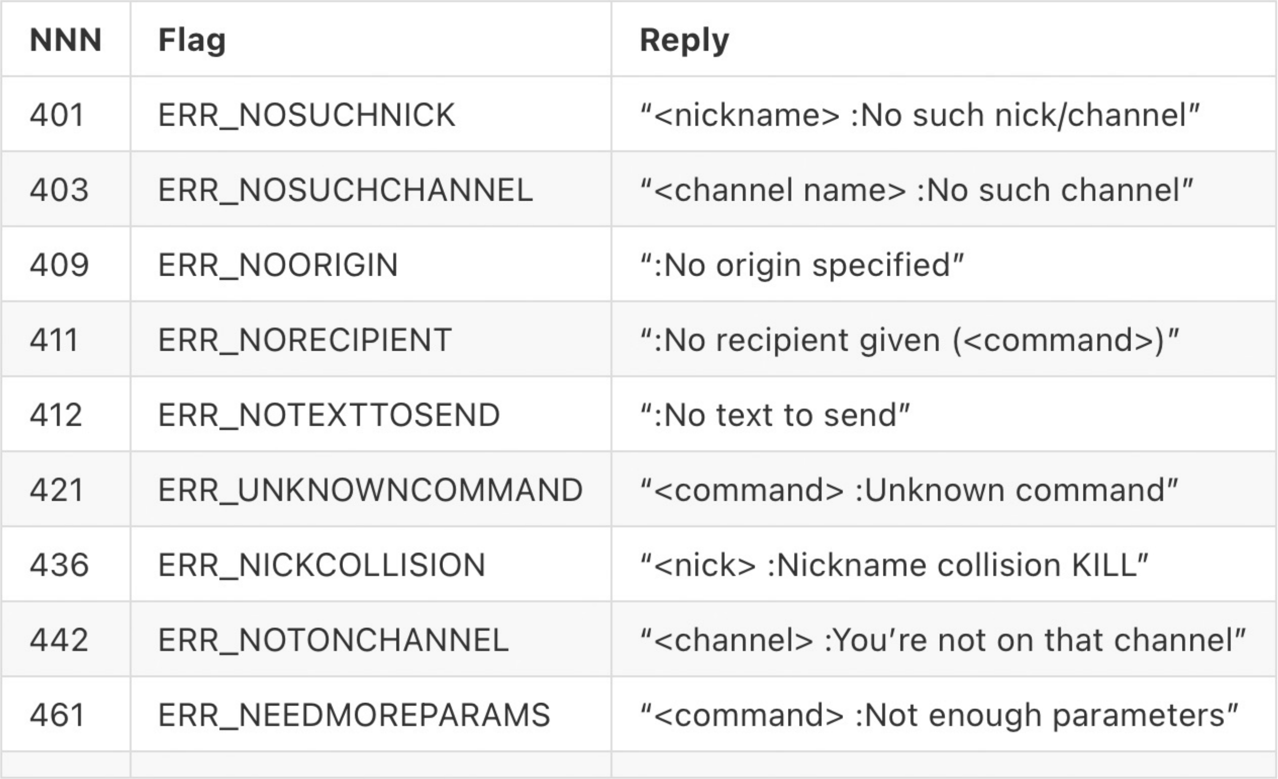
server name

The required response and error messages are listed below. You can find the details of each response/error message in Section 6 of RFC 1459 (https://www.rfc-editor.org/rfc/rfc1459.html).



NNN	Flag	Reply
321	RPL_LISTSTART	"Channel :Users Name"
322	RPL_LIST	"<channel> <# visible> :<topic>" ↳ # members.
323	RPL_LISTEND	":End of /LIST"
331	RPL_NOTOPIC	"<channel> :No topic is set"
332	RPL_TOPIC	"<channel> :<topic>"
353	RPL_NAMREPLY	"<channel> :[[@|+]<nick> [[@|+]<nick> [...]]]"
366	RPL_ENDOFNAMES	"<channel> :End of /NAMES list"
372	RPL_MOTD	":- "
375	RPL_MOTDSTART	":- <server> Message of the day - "
376	RPL_ENDOFMOTD	":End of /MOTD command"
392	RPL_USERSSTART	":UserID Terminal Host"
393	RPL_USERS	":%-8s %-9s %-8s"
394	RPL_ENDOFUSERS	":End of users"

```
(321) RPL_LISTSTART list start
(322) RPL_LIST list
(323) RPL_LISTEND list end   JOIN.
(331) RPL_NOTOPIC no topic
(332) RPL_TOPIC topic
(353) RPL_NAMREPLY
(366) RPL_ENDOFNAMES end of names
(372) RPL_MOTD
(375) RPL_MOTDSTART
(376) RPL_ENDOFMOTD
(392) RPL_USERSSTART users start.
(393) RPL_USERS users.
(394) RPL_ENDOFUSERS end of users

(401) ERR_NOSUCHNICK no such nick.
(403) ERR_NOSUCHCHANNEL no such channel
(411) ERR_NORECIPIENT no recipient
(412) ERR_NOTEXTTOSEND no text to send.
(421) ERR_UNKNOWNCOMMAND unknown command.
(431) ERR_NONICKNAMEGIVEN no nickname given.
(436) ERR_NICKCOLLISION nick collision.
(442) ERR_NOTONCHANNEL not on channel.
(461) ERR_NEEDMOREPARAMS need more parameters. "<command> :Not enough parameters"
(409) ERR_NOORIGIN. no origin specified.
Removed Error Codes
-----------------------
(451) ERR NOTREGISTERED
```



NNN	Flag	Reply
401	ERR_NOSUCHNICK	"<nickname> :No such nick/channel"
403	ERR_NOSUCHCHANNEL	"<channel name> :No such channel"
409	ERR_NOORIGIN	":No origin specified"
411	ERR_NORECIPIENT	":No recipient given (<command>)"
412	ERR_NOTEXTTOSEND	":No text to send"
421	ERR_UNKNOWNCOMMAND	"<command> :Unknown command"
436	ERR_NICKCOLLISION	"<nick> :Nickname collision KILL"
442	ERR_NOTONCHANNEL	"<channel> :You're not on that channel"
461	ERR_NEEDMOREPARAMS	"<command> :Not enough parameters"

Additional remarks for the general form of the server responded messages are summarized as follows. These remarks might make your implementation simpler.

- The prefix can be a fixed string, e.g., `mircd` or your preferred server name.
- `NNN` is the response/error code composed of three digits.
- `<nickname>` is required when it is known for an associated connection.
- The number of `<params>` depends on the corresponding response/error code.
- The last parameter must be prefixed with a colon (':').
- To deliver a `PRIVMSG` message received from `<userX>`, the received message from `<userX>` is prepended with the prefix `:<userX>` and then delivered to all users in the channel. The resulted message should look like `:<userX> PRIVMSG #<channel> :<message>`.

  From to channel.

The IRC Client

We use the console-based IRC client, weechat (https://weechat.org/), to test your implementation. You may use it to play with our sample implementation running at `inp111.zoolab.org` port `10004`. The relevant weechat commands are summarized as follows.

Note that weechat, by default, stores user configurations and data in `~/.config/weechat` and `~/.local/share/weechat` and disallows multiple instances to run simultaneously. To run multiple weechat instances on the same machine, you must use the `-d` option to specify a dedicated directory for each running instance.

- List available servers: `/server`

- Add a server: `/server add <servername> <hostname>/<port>`
  For example, `/server add mircd inp111.zoolab.org/10004` adds a server named `mircd` running at `inp111.zoolab.org` port `10004`.

- Set user nickname for a server: `/set irc.server.<servername>.nicks "<nickname>"`
  For example, `/set irc.server.mircd.nicks "user1"`.

- Connect to a server: `/connect <servername>`
  Connect to an added server named `<servername>`. For example, `/connect mircd`.

- List users on the server: `/users`

- List channels on the server: `/list`

- Join a channel: `/join <#channel>`

- Leave a channel: `/part`

- Close a buffer and leave a channel (or a server): `/close`

- Set channel topic: `/topic <topic>`. You can only do this when you are in a channel.

- Terminate weechat: `/quit`

- Send messages in a channel: Simply type the message you want to send, and all the users in the channel should receive the message.

Demonstration

1. [15 pts] Connect to the server and receive the message of the day (motd).

You may use the following commands to emulate two IRC users connecting to the same server. Suppose the server runs at `localhost` port `10004`. The command for *User1* and *User2* should be run in two terminals, respectively.

- User1

```
weechat -d ./user1 -r '/server add mircd localhost/10004; /set irc.
```

- User2

```
weechat -d ./user2 -r '/server add mircd localhost/10004; /set irc.
```

If you want to clear the settings for the users, remove the directories `user1` and `user2` in the current working directory before running the above commands.

2. [15 pts] List users on the server: weechat command `/users`

3. [15 pts] List available channels: weechat command `/list`

4. [15 pts] Join a channel successfully: weechat command `/join #chan1`

5. [15 pts] Get and set channel topic: weechat command `/topic hello, world!`

6. [15 pts] Send messages to a channel. Users can simply type messages in a channel.

7. [10 pts] Correct handle error condition.

To simplify the demo process, you may run the following commands iteratively for the two users after they have connected to the server.

- User1

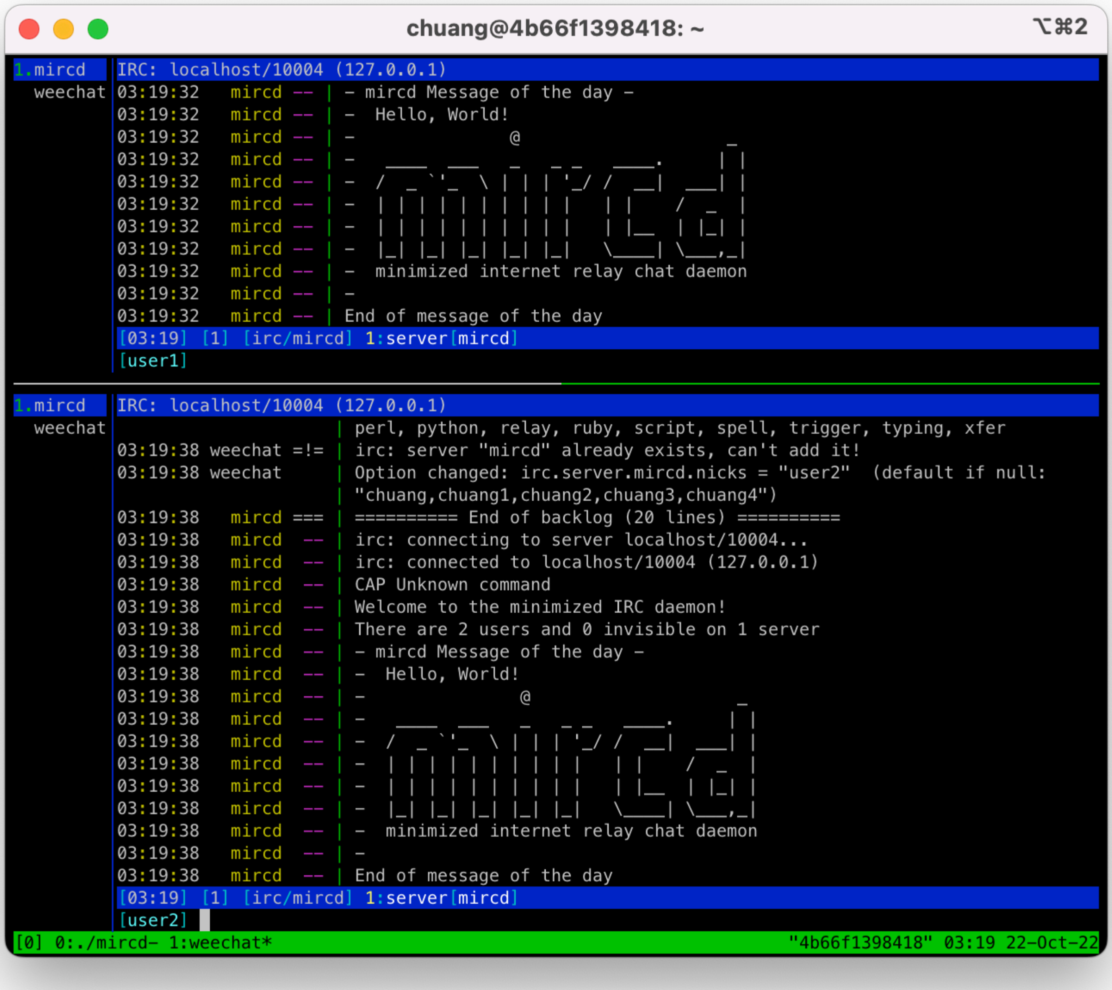
```
/join #chal1
/topic hello, world!
<wait for user2 to join, and then send some messages>
/part
/close
/quit
```

- User2

```
/users
/list
/join #chal1
/topic
<send some messages>
/part
/close
/quit
```

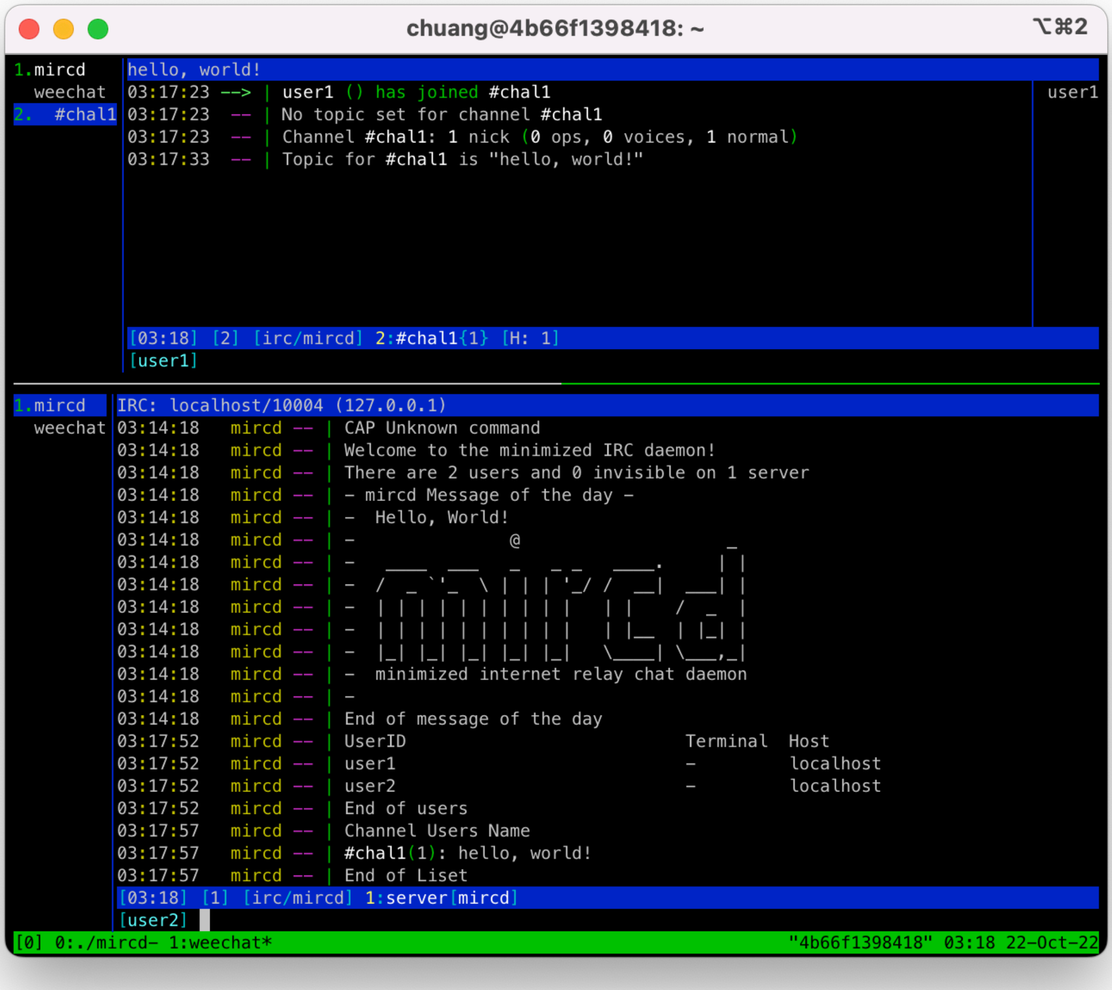
Here is a sample pcap file (https://inp111.zoolab.org/hw01/mircd.pcap) for you to inspect the messages exchanged between a server and two clients. A few screenshots are also available here for your reference.

- User1 and user2 connect to the server [top: user1; bottom: user2]



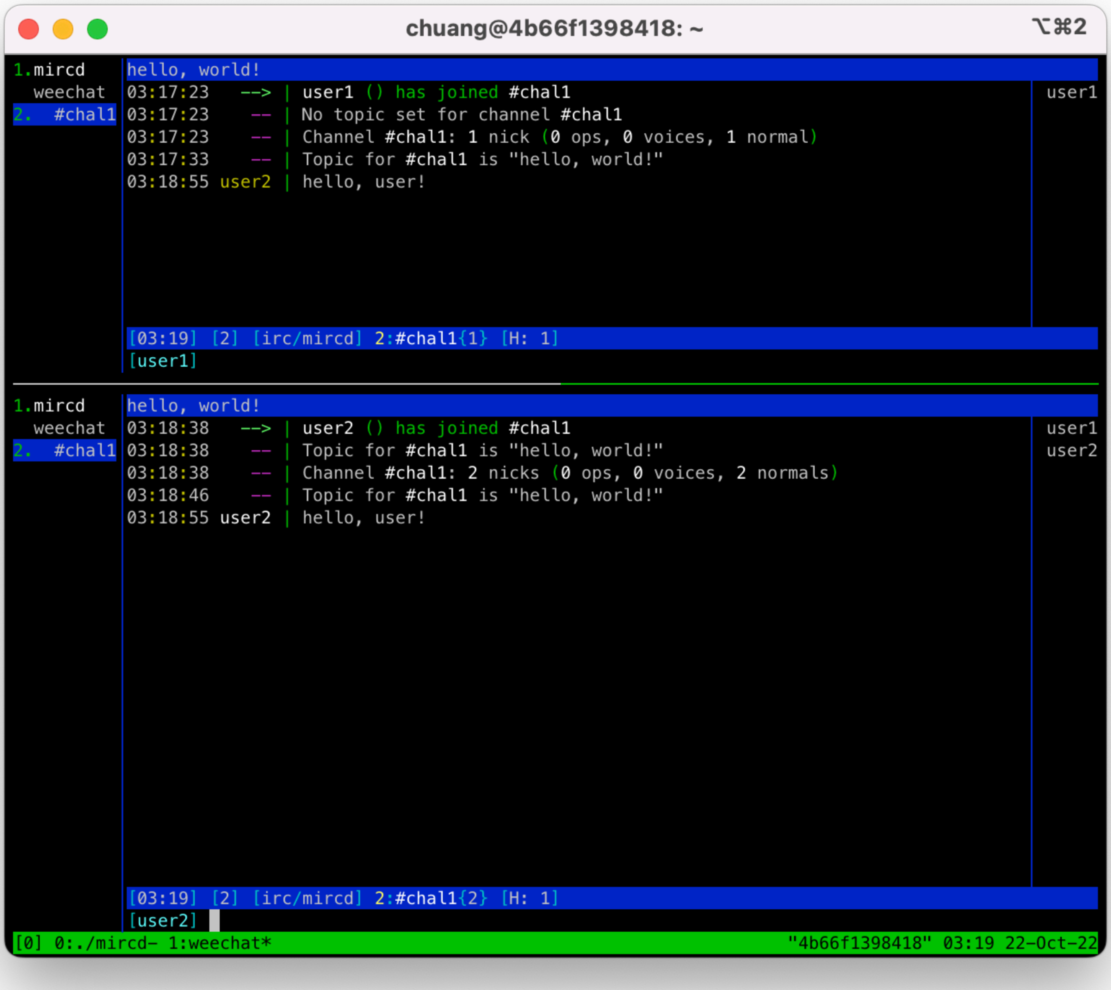
```
1.mircd  IRC: localhost/10004 (127.0.0.1)
 weechat 03:19:32   mircd -- | - mircd Message of the day -
         03:19:32   mircd -- | -  Hello, World!
         03:19:32   mircd -- | -                @                _
         03:19:32   mircd -- | -   ____ ___   _   _ _   ____.    | |
         03:19:32   mircd -- | -  /  _ `'_  \ | | | '_/ /  __|  ___| |
         03:19:32   mircd -- | -  | | | | | | | | | |   | |    /  _  |
         03:19:32   mircd -- | -  | | | | | | | | | |   | |__  | |_| |
         03:19:32   mircd -- | -  |_| |_| |_| |_| |_|   \____| \___,_|
         03:19:32   mircd -- | -  minimized internet relay chat daemon
         03:19:32   mircd -- | -
         03:19:32   mircd -- | End of message of the day
         [03:19] [1] [irc/mircd] 1:server[mircd]
         [user1]

1.mircd  IRC: localhost/10004 (127.0.0.1)
 weechat                      | perl, python, relay, ruby, script, spell, trigger, typing, xfer
         03:19:38 weechat =!= | irc: server "mircd" already exists, can't add it!
         03:19:38 weechat     | Option changed: irc.server.mircd.nicks = "user2"  (default if null:
                              | "chuang,chuang1,chuang2,chuang3,chuang4")
         03:19:38   mircd === | ========== End of backlog (20 lines) ==========
         03:19:38   mircd  -- | irc: connecting to server localhost/10004...
         03:19:38   mircd  -- | irc: connected to localhost/10004 (127.0.0.1)
         03:19:38   mircd  -- | CAP Unknown command
         03:19:38   mircd  -- | Welcome to the minimized IRC daemon!
         03:19:38   mircd  -- | There are 2 users and 0 invisible on 1 server
         03:19:38   mircd  -- | - mircd Message of the day -
         03:19:38   mircd  -- | -  Hello, World!
         03:19:38   mircd  -- | -                @                _
         03:19:38   mircd  -- | -   ____ ___   _   _ _   ____.    | |
         03:19:38   mircd  -- | -  /  _ `'_  \ | | | '_/ /  __|  ___| |
         03:19:38   mircd  -- | -  | | | | | | | | | |   | |    /  _  |
         03:19:38   mircd  -- | -  | | | | | | | | | |   | |__  | |_| |
         03:19:38   mircd  -- | -  |_| |_| |_| |_| |_|   \____| \___,_|
         03:19:38   mircd  -- | -  minimized internet relay chat daemon
         03:19:38   mircd  -- | -
         03:19:38   mircd  -- | End of message of the day
         [03:19] [1] [irc/mircd] 1:server[mircd]
         [user2]
[0] 0:./mircd- 1:weechat*                                "4b66f1398418" 03:19 22-Oct-22
```

- User1 created a channel `#chal1` , and user2 listed the channel(s) [top: user1; bottom: user2].



```
1.mircd    hello, world!
  weechat  03:17:23 --> | user1 () has joined #chal1                                 user1
2.  #chal1 03:17:23  -- | No topic set for channel #chal1
           03:17:23  -- | Channel #chal1: 1 nick (0 ops, 0 voices, 1 normal)
           03:17:33  -- | Topic for #chal1 is "hello, world!"





           [03:18] [2] [irc/mircd] 2:#chal1{1} [H: 1]
           [user1]

1.mircd   IRC: localhost/10004 (127.0.0.1)
  weechat 03:14:18   mircd -- | CAP Unknown command
          03:14:18   mircd -- | Welcome to the minimized IRC daemon!
          03:14:18   mircd -- | There are 2 users and 0 invisible on 1 server
          03:14:18   mircd -- | - mircd Message of the day -
          03:14:18   mircd -- | -  Hello, World!
          03:14:18   mircd -- | -               @                    _
          03:14:18   mircd -- | -   ____  ___   _   _ _   ____.     | |
          03:14:18   mircd -- | -  /  _ `'_  \ | | |'_/  /  __|  ___| |
          03:14:18   mircd -- | -  | | | | | | | | | |   | |    /  _  |
          03:14:18   mircd -- | -  | | | | | | | | | |   | |__  | |_| |
          03:14:18   mircd -- | -  |_| |_| |_| |_| |_|   \____| \___,_|
          03:14:18   mircd -- | -  minimized internet relay chat daemon
          03:14:18   mircd -- | -
          03:14:18   mircd -- | End of message of the day
          03:17:52   mircd -- | UserID                            Terminal  Host
          03:17:52   mircd -- | user1                             -         localhost
          03:17:52   mircd -- | user2                             -         localhost
          03:17:52   mircd -- | End of users
          03:17:57   mircd -- | Channel Users Name
          03:17:57   mircd -- | #chal1(1): hello, world!
          03:17:57   mircd -- | End of Liset
          [03:18] [1] [irc/mircd] 1:server[mircd]
          [user2]
[0] 0:./mircd- 1:weechat*                                            "4b66f1398418" 03:18 22-Oct-22
```

- User2 joined channel `#chal1` and sent a message [top: user1; bottom: user2].



```
1.mircd    hello, world!
  weechat  03:17:23   --> | user1 () has joined #chal1                        user1
2.  #chal1 03:17:23    -- | No topic set for channel #chal1
           03:17:23    -- | Channel #chal1: 1 nick (0 ops, 0 voices, 1 normal)
           03:17:33    -- | Topic for #chal1 is "hello, world!"
           03:18:55 user2 | hello, user!

           [03:19] [2] [irc/mircd] 2:#chal1{1} [H: 1]
           [user1]

1.mircd    hello, world!
  weechat  03:18:38   --> | user2 () has joined #chal1                        user1
2.  #chal1 03:18:38    -- | Topic for #chal1 is "hello, world!"               user2
           03:18:38    -- | Channel #chal1: 2 nicks (0 ops, 0 voices, 2 normals)
           03:18:46    -- | Topic for #chal1 is "hello, world!"
           03:18:55 user2 | hello, user!

           [03:19] [2] [irc/mircd] 2:#chal1{2} [H: 1]
           [user2]
[0] 0:./mircd- 1:weechat*                          "4b66f1398418" 03:19 22-Oct-22
```

Error test_case

In this part, we will give some test_case let you to check your error handle.

We use `nc` command to connect server to test error, you need to use NICK and USER command after `nc` , since only after both NICK and USER have been received from a client does a user become registered.

```
nc localhost 10004            //connect to your server

NICK Kilac
USER Kilac localhost test kilac
    <<success connect>>
JOIN #hehe            //These arguments can set by yourself
```

1. (401) ERR_NOSUCHNICK

```
PRIVMSG #cool hello            // cool doesn't exist
:mircd 401 Kilac #cool :No such nick/channel
```

2. (403) ERR_NOSUCHCHANNEL

```
PART #cool            // cool doesn't exist
:mircd 403 Kilac #cool :No such channel
```

3. (411) ERR_NORECIPIENT

```
PRIVMSG
:mircd 411 Kilac :No recipient given (PRIVMSG)
```

4. (412) ERR_NOTEXTTOSEND

```
PRIVMSG #hehe
:mircd 412 Kilac :No text to send
```

5. (421) ERR_UNKNOWNCOMMAND

```
TEST
:mircd 421 Kilac TEST :Unknown command
```

6. (431) ERR_NONICKNAMEGIVEN

```
NICK
:mircd 431 :No nickname given
```

7. (436) ERR_NICKCOLLISION

```
NICK Kilac             // Kilac already used by another one
:mircd 436 Kilac :Nickname collision KILL
```

8. (442) ERR_NOTONCHANNEL

```
TOPIC #hehe
:mircd 442 Kilac #hehe :You are not on that channel

PART #hehe
:mircd 442 Kilac #hehe :You are not on that channel
```

9. (461) ERR_NEEDMOREPARAMS

```
USER
:mircd 461 Kilac USER :Not enought parameters
```

> Next week demo we may use another test_case to test your server !!!