

JS8Net User Guide

Author: Lawrence Byng

Date: July 2022

Version 1.0

Table of Contents

JS8Net User Guide.....	1
What is JS8Net?.....	2
Why was JS8Net developed?.....	2
So how do I run a net with JS8Net?.....	3
Starting the Application.....	3
Sending a QST.....	4
Asking for Checkins.....	5
Processing checkins.....	5
Opening the net.....	5
Sending the Roster.....	6
Handing over to the first station.....	6
Recap.....	6
Directed Nets, NCS responding to a station.....	6
Offset Plans.....	7
Station Awards.....	8
Dynamic Content Macro Language.....	9
Click Sequencing thru Macro Messages.....	9
Time Related Macros.....	9
Net Field Macros.....	10
Net State Macros.....	10
Checkbox Macros.....	11
Roster Macro.....	12
Checkin Macros.....	13
Call Sign Macros.....	13
Offsets Plan Macro.....	13
Station Signal Macros.....	13
Last Station Macros.....	14
Next Station macros.....	14
Award Macros.....	14
Edition Field Macros.....	14
select one of the message formats by random.....	15
Command line options.....	16
Advanced Features.....	19
Distributing an offset plan.....	19
Modifying Pre-defined messages.....	19
Editing the built in macros.....	20
Simulation mode.....	21
Updating Frequency from a QSY message.....	21
Examples.....	21

What is JS8Net?

JS8 Net is a python application developed specifically for hosting JS8 ham radio nets typically on HF. The application provides many enhanced features that are simply not achievable using a keyboard-only style of operating. An overview of some of the main features is as follows:

- Roster to display participating station call signs, operator name, net status and offset
- Point and click style of net operation.
- Can be run as net control station view or as participant station view
- Convenient notepad area to pre-type 'report' and 'announcement' type communications prior to the net
- Auto save feature
- Built in Dynamic Content Macro language allowing construction of wide variety of point and click messages. The Macros provide access to the relevant fields and functions of JS8Net.
- Customization feature allowing customization of all point and click messages
- Built in state machine and text mode parser to track progress of the net. This allows participant stations to view upto date net information as if connected via the internet.
- No internet connection required. Uses text mode communication over JS8 only.
- Real time tracking of SNR, bad frames and time delta for each station.
- Offset plan distribution functionality built in.
- Use of the JS8Net application is optional. Stations using JS8Call only as well as stations using JS8Call + JS8Net are both able to participate in the net.

Why was JS8Net developed?

I developed JS8 net primarily as a tool for facilitating running a JS8 net. After having run several JS8 nets using a keyboard only approach, it seemed to me there was an opportunity for a software solution to help run the net. Using a keyboard only approach requires lots of fast typing and results in lots of typos and frequent delays while responses are constructed and previewed prior to sending to the net. The whole process seemed to me to be less than optimal so this was the initial impetus to build the app.

As a result you will see there is large emphasis on allowing the majority of net control features to be handled as simply point and click operations with the minimum of typing. There is however still the flexibility that allows net control full control over message content and does not preclude any of the keyboard only mode type actions for constructing and sending messages.

Providing an enhanced visual experience for net participants was also a consideration. Features such as a visual roster, indication of who has taken their turn and who is next as well as details

about the net itself such as start time, frequency and group name. Additionally, the ability to communicate this information out to the various participant stations as if they were connected via internet but without the use of any internet connections was also high on the list.

In order to provide this participant station functionality, the application needs to be able to track the progress of the net and parse the messages from other stations and net control. This requires a fairly elaborate state machine and parsing logic. At the core of JS8-NET you will find a state machine that handles the various transitions between different net states and different participant states as the net progresses. In addition JS8Net uses a novel approach of parsing the various messages and attempting to follow along based on the content of the various text messages going back and forth. This is completely experimental and unique but so far it appears to work quite well.

So how do I run a net with JS8Net?

First if you are one of those people than can type everything perfectly at 100 words per minute and prefer to use keyboard only with no additional software when using digital modes then this is not for you so please go ahead and close this manual and application and go ahead and continue with what you were doing.

If however you are like me and your typing is less than perfect, and you need to correct typo then review the message for accuracy and have to squint a little when reviewing the message, then please read on. Yes there is a small learning curve when you first start to use the application but in my view it is worth it. Having used the application extensively to run JS8 nets for some time, I find it is lots of fun to use the application and I would not consider running the net any other way currently.

Probably the best way is to give a start to finish example from the perspective of net control running the net, along with an explanation.

Starting the Application

First start JS8Call. Make sure you have enabled the external TCP connections interface in JS8Call.

Now start the JS8Net application in net control mode as follows...

```
$ python ./js8_net_client.py --interface=netcontrol
```

This will use the default save file for saving and loading any data used by the application. The default data file provided on github is a good starting point and should be used to start off with.

Fill out the top three rows of the screen with information about your JS8 net. All data is required and will be used in subsequent steps.

If there is any data on the roster this will be from the last time the application was run. Click the 'Clear' button to clear the roster data. Now the only data on the roster will be your call sign and <NCS> showing that you are net control station.

If the name field shows as '-', click on your call sign on the roster then in the name entry field put your name then click update. Now the roster should display your name along side the call sign. The name will be saved in the save data, so once the name is set this will be remembered by the application indefinitely.

You can go ahead and add any call signs manually to the roster now. You can also edit the names of each participant in the same manner. These will also be saved in the save data file.

It is a good idea to initially enter the names manually so that the application has some data saved so that when the auto check in feature is used, the application can automatically add the station and the name to the roster automatically.

Sending a QST

First make sure 'Flash Buttons' is checked. This will flash the buttons that are relevant as a guide as the net progresses.

Second, set the mode for passing the data to JS8 call as 'Post to JS8Call only'. This mode will set the text in JS8Call and allow it to be previewed prior to sending. A second mouse click is required to actually send the message. This is done using the 'Go' button to the right of the preview window at the bottom of the screen. Posting first then sending is a good place to start and allows greater control over sending the messages. The other option is 'Post to JS8Call + Send'. This option is for advanced users only. It will automatically send the message after about 2.5 seconds allowing true one click operation of many features.

Prior to the net, typically 30 minutes prior and 15 minutes prior it may be useful to send out a QST to all stations as a reminder. If you look at the lower portion of the screen, you will see two rows of 7 buttons each. The top left of these buttons is marked 'QST'. Go ahead and click this button. The preview window in JS8NET will show the QST message based on the net control info that you entered in the starting the application step. Now click the 'QST' button a second time. You will see that the message preview window provides a different format but largely the same content to the message. Most buttons have several different formats available. When you have previewed the message and are happy with it, make sure the TX button at the top right of JS8Call is enabled then go ahead and click the 'Go' button (flashing red/blue) at the lower right of the preview window in JS8NET. JS8Call will begin sending the message.

Congratulations! You have now gone thru the initial small learning curve on how to operate JS8Net. This is about 90% of the process of using the application. The rest follows this same pattern but relates to the different stages of the net.

The QST message also sends out much information about the net that can be decoded by any participant station that is running the JS8Net software. Participant stations can start up the program as a participant station then clear all of the fields and wait for the QST. When the participant station hears the QST message it will populate most of the fields automatically. At

this point there should be sufficient information for the participant station to decide to join the net and construct the appropriate messages. The remaining fields will be populated when the net starts.

Asking for Checkins

Now click the 'Ckin?' button. Prior to the net time that you specified earlier, the messages will be pre-check messages. After the net starts these will change to regular check in messages. Once you have selected the format required go ahead and click the flashing 'Go' button. JS8Call will send the message

Processing checkins

Make sure that 'Auto Check-in' checkbox is checked at the upper right of the screen and wait for a station to appear on frequency. When the station send a JS8 message their call sign will appear on the roster automatically. If you prefer to add them manually then simply uncheck the 'Auto Check-in' checkbox and proceed to enter the station in the 3 fields above the roster. Initially any stations checked in automatically will appear with a status of <HEARD> or <CHECKIN>. If you know this is a station that wants to be checked in to the net go ahead and click the 'Got u' button. This will provide several messages for confirming those stations that just appeared on the roster. Now go ahead and click the flashing 'Go' button. You will see that the message is sent and also the status of the stations in the roster is updated to <STANDBY>. This means they are ready and standing by for the net. If you are not sure if a station wants to be added or not you can enter free-form text in the 'Other' window and then click the 'Go' button that is to the right of the 'Other' window. If this station responds that he is not participating you can set the status to <IGNORE> or <SWL> or you can simply select that station on the roster then click delete to remove it from the roster.

Opening the net

When the start time rolls around, you will see the Timer at the top right of the screen change to "ON AIR" This is your cue to start the net. Go ahead and click the 'Open' button. Again there are a couple of format options choose the one you want then click the 'Go' button. After the message is sent you can click the checkin button to ask if there are any more checkins. Note that the checkin messages now are regular checkin message rather than pre-check messages. If there are additional station to add go ahead and add them in the same way as before in the checkins section.

Any stations that started the JS8Net application late and who wish to join the net can listen to the opening message. The JS8Net application will automatically decode the information and populate the screens of the participant stations.

Sending the Roster

Now click on the 'Roster' button then click the flashing 'Go' button. All of the stations participating in the net will be set to <STANDBY> status and the roster sent out to all of the participant stations. Any participant station that is also running JS8Net will see the roster appear on their screens that reflects the same roster as is on net control screen.

Handing over to the first station

Click on the 'First' button. You will see that the first station on the roster is now queued up by updating its status to <NEXT>. Now go ahead and click the flashing 'Go' button. This will send the message. After a short delay, you will also notice the countdown timer in the middle of the screen starts counting down. This is a guide to time the response of the station. If the station responds, then the status on the roster will be updated to <TALKING>. If the station does not respond in a certain amount of time such as that amount of time from the countdown timer, then you can send a reminder to the station. You can do this by clicking on the 'Nudge' button and then click 'Go'. You will see the countdown timer start again. This timer is as a reference point only. If you decide that the station is MIA then go ahead and skip the station by pressing the 'Skip' button then the 'Go' button. This will update the roster to <SKIP> and queue up the next station on the roster and update the status to <NEXT>.

Recap

Lets pause for a minute to review where we are and what we have achieved so far. So up to this point, net control has sent out a whole sequence of messages including QST, PreCheck request, Checking acknowledgement, Net Open, Sending of the full roster and handing over to the first participant all without typing a single letter on the keyboard at all. All we have needed to do so far is click this button then click go, click that button twice then click go, click this button then click go etc etc. One of the design goals was to minimize typing and this is a perfect example to illustrate this.

And on the other side of the equation, each participant station who also runs the JS8Net application has been able to receive all of the information for the net, populate all of the fields on the main window and populate the roster and join the net all without typing a single character at the keyboard either.

Directed Nets, NCS responding to a station

Once the first station responds, for a directed net that is the queue for net control to construct and send a response. This is where the 'Next' button comes in. The next button and the 'End' button have some very neat features under the cover. Let me explain...

Once the first station has finished his response and handed back to net control go ahead and click the 'next button'. You will see a message appear in the preview window and you will also see the

roster pupdate to reflect that the first station is <DONE> and to queue up the next station by making it as <NEXT>

Now before you click the 'Go', go ahead and check some of the checkboxes above the preview window such as 'UR Welcome', 'Tks', 'SNR', and the one just to the left of 'Great Evening'. Now click the next button again. Note how the preview window has been updated with an expanded text version of this information included an SNR report. Now click the 'Next' button again, you should see a full SNR report including bad frames and time delta. Now type some text into the 'other' window such as 'awesome signal from your new vertical antenna and good luck with the contest'. Now click the 'Next' button one more time. You will see that the message has been constructed automatically.

If you wish to give the first station a second opportunity to communicate with the net, instead, go ahead and click the checkbox to the left of 'More'. You will notice that the response goes back to the first station but does not hand off to the next station just yet. When you have the format of message you wish to use go ahead and click 'Go' to send the message. Once the station responds back a second time, Click the appropriate checkboxes including the one next to the 'Great Evening' checkbox then select the format you want by clicking on the 'next' button several times then click 'Go'.. This will reply back to the first station and hand off the the next on the roster.

So all of the net control aspects were fully automated and the only text we needed to type was the specific text of the additional part of the message to send to the first station.

If the station does not respond, you can hit 'Nudge' and then 'Skip' same as before.

Continue thru the roster until all stations had an opportunity to speak.

Offset Plans

JS8Net has built in a pre-set offsets plan. This is a list of offsets that can be used by all stations participating on the net to ensure that stations are distributed as evenly as possible. The idea behind this is to help minimize collisions if two stations happened to hit transmit at the same time. If the stations have non-overlapping signals then they can both be decoded simultaneously.

In order to use this feature effectively, net control and all of the stations must be running the JS8Net application. After the roster is sent to all stations is probably the best time to send out the offsets plan. This will assign each station an offset based on their relative position in the roster. The macro to send out the offsets plan to all of the stations can be found under the 'Extras' button on the main screen. Once the new offsets plan is sent out, each participant station will now have an offset set per the plan. This can be seen in the 'offset' field on each of the stations. Each participant station can still override this offset but that will reduce the effectiveness.

If you wish to set your own offsets plan that is different from the pre-set plan provided, This can be done by specifying your new offsets plan on the command line when the application is run.

The offsets plan is a comma separated list. The first offset refers to net control only. The second offset thru the end of the list of offsets is used to set each of the participant stations. If there are more stations than there are offsets on the list, then the list will wrap around and double up, triple up etc stations on the offsets provided until all stations have been assigned an offset. For this

reason it is better to provide an longer offsets list to handle a large number of stations and to not use all offsets on the list rather than have a short list and to double up stations on the same offset just because of taking shortcuts in the pre-planning steps for the net.

Station Awards

Included with the program are a couple of extras! These are meant as a fun but also as usefull feedback to the participant stations. These focus on three key areas of weak signal communication; SNR, dropped frames and Time Delta.

There are two pre-coded awards:

- 1) The station that has the lowest Signal to noise ratio while still maintaining perfect communication. This is the 'weak signal' award
- 2) The station with the time sync that is a closest match for net control. This is the 'Time Sync' award

The messages for these awards can be found on the 'Awards' button. The stations are computed automatically when the button is clicked by using the most up to date information from the roster. If there are multiple winning stations then these will all be listed in the award.

Dynamic Content Macro Language

Each of the main buttons in JS8 net have behind them a set of messages that make extensive use of JS8Net dynamic content macros. Included in JS8Net is large number of these Dynamic content macros that tie into all aspects of the application.

The macros can be used to construct text messages based on the latest up-to-the-second information in JS8Net. Here is an example of a dynamic content macro in use in an actual message. This example is from the participant station's 'QRT' button

```
%GROUPNAME %SEQ I AM GOING QRT TKS
```

```
%SEQ GOING QRT ALOHA ALL %ENDSEQ
```

This is one of the more straight forward examples. Each of the dynamic content macros begins with a % sign. %GROUPNAME for example inserts the current group name from the main window. %SEQ and %ENDSEQ allow for a sequence of different message formats that can be cycled thru by clicking the 'QRT' button multiple times. The text between the first %SEQ and the next %SEQ or %ENDSEQ is shown when the 'QRT' button is clicked the first time. Next time the 'QRT' button is clicked, the text changes to that between the second %SEQ and the %ENDSEQ macros. So this results in the following...

on the first click the message show in the preview window is:

```
@MYGROUP I AM GOING QRT TKS
```

and on the second click the message the preview window changes to:

```
@MYGROUP GOING QRT ALOHA ALL
```

on the third click the message cycles around again and show the first message and so on. So it is a very convenient way to pre-write several different message formats that can be clicked thru in sequence during the net and sent as a message.

Click Sequencing thru Macro Messages

```
%SEQ  %ENDSEQ
```

Allows multiple different formats of message to be created for click sequencing.

```
%SEQ text-1 %SEQ text-2 %SEQ text-n %ENDSEQ
```

Time Related Macros

```
'%ZULUTIME
```

in line replace with the current zulu time

%LOCALTIME

in line replace with the current local time

Net Field Macros

%NETTYPE

in line replace with the value from the 'net type' field. This will be wither 'Directed' or 'Round Table'

%NETSTARTTIME

in line replace with the value of the net srt time field

%NETFRE

in line replace of the net frequency field

%NUMROUNDS

in line replace of the number of rounds field

%CURRENTROUND

inline replace for the current round field

%GROUPNAME

in line replace for the group field

Net State Macros

%IFNETSTARTED %ENDIFNETSTARTED

This macro will include the text between the bookend macros only if the net is started. This is determined by comparing current zulu time to the net start time.

%IFNNETSTARTED %ENDIFNNETSTARTED

This macro will include the text between the bookend macros only if the net is **not** started. This is determined by comparing current zulu time to the net start time.

`%IFPREV %ENDIFPREV`

This macro will include the text between the bookend macros only if the 'Previous' field is not empty.

`%IFNPREV %ENDIFNPREV`

This macro will include the text between the bookend macros only if the 'Previous' field *is* empty i.e. if there is not a previous station

`%IFNEXT %ENDIFNEXT`

This macro will include the text between the bookend macros only if the 'Next' field is not empty.

`%IFNNEXT %ENDIFNNEXT`

This macro will include the text between the bookend macros only if the 'Next' field *is* empty i.e. if there is not a Next station

Checkbox Macros

The set of standard messages included with the application use these on the 'Next' and 'End' buttons. These however can be used on any of the buttons by using the customization –edit command line option to override the set of standard messages.

`%IFSNR %ENDIFSNR ""`

This macro will include the text between the bookend macros only if the 'SNR' checkbox is checked.

`%IFGOODEVE %ENDIFGOODEVE`

This macro will include the text between the bookend macros only if the good evening checkbox is checked.

`%GOODEVE`

This inline replacement macro replaces the text with the value of the combo_aloha combo box.

`%IFOTHER %ENDIFOTHER`

This macro will include the text between the bookend macros only if the good ‘other’ checkbox is checked.

%OTHERMSG

This inline replacement macro replaces the text with the text that was typed into the ‘Other’ multi line entry field.

%IFFURTHER %ENDIFFURTHER

This macro will include the text between the bookend macros only if the ‘further’ checkbox is checked.

%IFNFURTHER %ENDIFNFURTHER

This macro will include the text between the bookend macros only if the ‘further’ checkbox is **not** checked.

%IFTKS %ENDIFTKS

This macro will include the text between the bookend macros only if the ‘TKS’ checkbox is checked.

%IFNTKS %ENDIFNTKS

This macro will include the text between the bookend macros only if the ‘TKS’ checkbox is **not** checked.

%TKSMMSG

This inline replacement macro replaces the text with the value of the combo_tks combo box.

%IFURW %ENDIFURW

This macro will include the text between the bookend macros only if the ‘Ur Welcome’ checkbox is checked.

Roster Macro

%ROSTER

This inline replacement macro replaces the text with the full roster including all call signs and names.

Checkin Macros

`%IFCHECKIN %CHECKINCALL %ENDIFCHECKIN`

This macro is used to construct the automatic reply to any station that checks into the net. The default message is:

`%IFCHECKIN OK I HAVE ADDED %CHECKINCALL TO THE ROSTER. GE TKS FOR CHECKING IN. %ENDIFCHECKIN`

Call Sign Macros

`%NCSNAME`

This returns the name of the NCS station

`%NCSCALL`

This return the call sign of the NCS station

`%NETNAME`

This returns the name of the net from the main window

Offsets Plan Macro

`%OFFSETSPLAN`

This macro does an inline replace with the current offsets list.

Station Signal Macros

`%LBADFRAMES`

This macro returns the number of bad frames that have been counted so far for the last transmitting station.

`%LTIMDELTA`

This macro returns the time delta for the last transmitting station.

Last Station Macros

`%LC`

This macro returns the call sign of the last transmitting station.

`%LN`

This macro returns the name of the last transmitting station.

`%LSR`

This macro returns the signal to noise ratio number for the last transmitting station.

Next Station macros

`%NC`

This macro returns the call sign of the next queued up station.

`%NN`

This macro returns the name of the next queued up station.

Award Macros

`%WEAKSIGNALAWARD`

This does an inline replace with the calculation of the station that wins the weak signal award.

`%SYNCAWARD`

This does an inline replace with the calculation of the station that wins the time sync award.

Edition Field Macros

`%PROF`

This macro returns the value of the 'Edition' field on the main page. This is typically set to the day of the week and is useful in constructing the opening message to the net.

`%PROF(...)` `%ENDPROF`

This macro is used to conditionally include the text between the bookends if the text in the brackets matches the content of the 'Edition' field on the main page.

Example:

%PROF(Tuesday) It is Tuesday! %ENDPROF

This will include the text 'It is Tuesday' if the contents of the edition field is equal to 'Tuesday'

select one of the message formats by random

%CHOICE %CHOICE %ENDCHOICE

This macro will conditionally include one of the text phrases specified based on randmo. The phrases are between the bookends %CHOICE / %CHOICE and %CHOICE / %ENDCHOICE.

Example:

%CHOICE Good evening %CHOICE Hello and welcome %CHOICE greetings %ENDCHOICE

in the above example, the included phrase will be chosen at random from the three phrases provided.

Command line options

--interface= , --i=

default = Net Control

This option will run the program in one of its two modes net control or participant:

examples:

Participant Station

```
$ python ./js8_net_client.py --i=participant
```

Net Control Station

```
$ python ./js8_net_client.py --interface=netcontrol
```

Please note that participant stations should not use the net control option if they plan to participate as a participant. Net control option is for net control only.

--net_file= , --n=

default = js8net_save_data.txt

```
$ python ./js8_net_client.py --interface=netcontrol --net_file='js8net_save_data.txt'
```

This option allows the user to specify alternate data files for the application. The js8net_save_data.txt is the default file and should be used as the starting point. If you wish to create your own alternative files, first copy this file to your new data file then start the program pointing to the new data file.

--js8call= , --j=

default = 127.0.0.1:2442

```
$ python ./js8_net_client.py --interface=netcontrol --js8call='127.0.0.1:2442'
```

This option allows you to specify a different ip and or port number for js8 call. These must point to the computer ip that has the JS8Call application running and the port that is used for the TCP connection.

--group= , --g=


```
$ python ./js8_net_client.py -interface=participant --group='@HINET'
```

This option is primarily for participant station and allows the group field to be pre-specified from the command line. This is preferable if you know the group that will be used for the net that you plan to participate in.

--frequency= , --f

```
$ python ./js8_net_client.py -interface=participant --frequency='fromjs8call'
```

This option allows the application to use the same frequency that is specified in JS8Call. When the JS8Call frequency is changed then the frequency field in the application will also change.

```
$ python ./js8_net_client.py -interface=participant -frequency='7.080'
```

This option will set the frequency field in the application to the frequency specified in MHz.

--main_offset= , --m=

default 1000

```
$ python ./js8_net_client.py -interface=participant --main_offset=1000
```

This sets the offset used for transmitting messages to the value specified

```
$ python ./js8_net_client.py -interface=participant --main_offset='from_file'
```

This set the value of the offset field to the value save in the save file

```
$ python ./js8_net_client.py -interface=participant --main_offset='from_plan'
```

This sets the value of the offset field to the value from the offsets plan

--boundary= , --b=

default 700

```
$ python ./js8_net_client.py -interface=netcontrol --boundary=800
```

This option sets the boundary offset between the main offset and the sidebar offset. Messages using an offset above this number will appear in the main window. Message using an offset below this value will appear in the sidebar window.

--client_read_details , --r

default = False

```
$ python ./js8_net_client.py -interface=participant --client_read_details
```

This option is for participant stations to read the details of the main window from the save data file.

--profile= , --p=

Default = day of week

```
$ python ./js8_net_client.py -interface=netcontrol --profile='Tango'
```

This will override the value of the 'Edition' field on the main page.

--counter= , --c=

default = 200 and visible

```
$ python ./js8_net_client.py -interface=netcontrol --counter='off'
```

This will disable and hide the countdown timer on the main page.

```
$ python ./js8_net_client.py -interface=netcontrol --counter='300'
```

This will set the 'countdown' / count-up timer to count to 300

--delay= , --d=

default = 25

```
$ python ./js8_net_client.py -interface=netcontrol --net_file='js8net_save_data.txt'
```

This option changes the delay when using the post + send option for true one-click operation. The delay is to give you time to review the message or to click on another message format prior to the

message being sent. Each time you click on a new format, the delay timer starts again. Once it reaches the value specified, send activates automatically when the post + send option is used.

--visual= , --v

default =

'background:LightGray,main:SeaGreen1,side:LightBlue1,flash1:red,flash2:blue'

```
$ python ./js8_net_client.py -interface=netcontrol --  
visual='background:LightGray,main:SeaGreen1,side:LightBlue1,flash1:red,flash2:blue'
```

This option allws the screen colors to be modified.

Advanced Features

Distributing an offset plan

--offsets= , --o=

default = '1337,700,870,1140,1210,750,920,1190,1260,800,970,1240'

```
$ python ./js8_net_client.py -interface=netcontrol --offsets='1337,400,500,600,700'
```

A new offsets plan can be specified on the command line as in the above example. Typically the offsets plan is sent to the sations after the roster is sent out to them. The message to send the offsets plan can be found on the 'Extras' button

Modifying Pre-defined messages

--combo_tks= , --t=

default ='Report,Good Report,Great Report,Signal Report,Great Question,Good Idea,Good Comment'

```
$ python ./js8_net_client.py -interface=netcontrol --combo_tks='comments,heads  
up,info'
```

This option allows the customization of the combo drop down linked to the 'Tks' checkbox which is used by net control. The options in the list are provided as a comma separated list as in

the above example. These values are used by the Macro language. The current value of this field can be accessed from the macro language by using the %TKSMSG macro.

The default macros provided with the application (Next and End macros) use this value in the following context:

```
%IFTKS Thanks %LN for the %TKSMSG. %ENDIFTKS '
```

If you wish to change to values of the combo dropdown as well as the context then you will also need to edit the pre-defined macros so that the combination of the edited macro + the combo text makes grammatical sense. This can be done using the –edit feature. Please also refer to previous section about Dynamic Content Macros.

--combo_aloha= , --a=

default ='Great Evening,Good Evening,Great Rest of the Day,Good morning'

```
$ python ./js8_net_client.py -interface=netcontrol --combo_aloha='great morning,nice day,great rest of the day'
```

This option allows the customization of the combo drop down linked to the ‘good evening’ checkbox on the far right of the screen as used by net control. The options in the list are provided as a comma separated list as in the above example. These values are used by the Macro language. The current value of this field can be accessed from the macro language by using the %GOODEVE macro.

The default macros provided with the application (Next and End macros) use this value in the following context:

```
%IFGOODEVE have a %GOODEVE %LN. %ENDIFGOODEVE '
```

If you wish to change to values of the combo dropdown as well as the context then you will also need to edit the pre-defined macros so that the combination of the edited macro + the combo text makes grammatical sense. This can be done using the –edit feature. Please also refer to previous section about Dynamic Content Macros.

Editing the built in macros

--edit , e

```
$ python ./js8_net_client.py -interface=netcontrol --edit
```

This will put the application into edit mode so that the macros can be edited. Only macros relating to the current view are able to be edited and these are different for netcontrol and participant.

Simulation mode

--simulate , --s

```
$ python ./js8_net_client.py -interface=netcontrol --simulate
```

This is an advanced option and is used only for testing of new messages. It allows a simulated message to be sent to the application as if it had been received from another station. The command line option will show a 'simulate' checkbox on the main windows. The checkbox must be clicked to activate simulate mode. Do not use this option unless you have mastered all other aspects of the application.

Updating Frequency from a QSY message

--update_freq_on_qsy , --u

default = False

```
$ python ./js8_net_client.py -interface=netcontrol --update_freq_on_qsy
```

This option will automatically update the value of the frequency field upon receipt of a QSY message from net control. The recommendation is to keep this option set to off unless your station automatically activates re-tuning when the frequency is changed.

Examples

Running the application as net control and specifying data file for saving application data:

```
$ python ./js8_net_client.py --interface=netcontrol --net_file=js8net_save_data.txt
```

Running the application as net control with customized colors:

```
$ python ./js8_net_client.py --interface=netcontrol --net_file=js8net_save_data.txt  
--visual='background:yellow,main:green,side:blue,flash1:purple,flash2:white'
```

Running the application as net client or participant. In this example the program reads the data from the data file, acquires the frequency to be used from JS8Call and overrides the group name from the command line parameter.

```
$ python ./js8_net_client.py --interface=participant --net_file=js8net_save_data.txt  
--simulate --client_read_details --frequency='fromjs8call' --group='@HINET'
```

Running the application as net client or participant. In this example the program uses the simulate mode to allow testing of modified edit data

```
$ python ./js8_net_client.py --interface=participant --net_file=js8net_save_data.txt  
--simulate --frequency='fromjs8call' --group='@HINET'
```