

# **JHamTune: User Manual**

**Margaret Leber <k3xs@arrl.net>**

---

# JHamTune: User Manual

by Margaret Leber

Published 30 January 2004

Copyright © 2004 Margaret Leber

This document is a guide for users of the JHamTune system for controlling amateur radio stations.



---

---

---

## Table of Contents

1. Installation .....	1
The Preview Edition .....	1
System Requirements .....	1
Java Runtime .....	1
Java Libraries .....	1
Configuration .....	1
2. Future Directions .....	2
3. Operation .....	3
Main Window .....	3
Menus .....	4
Tuner Display .....	5
Scanner Window .....	5
Sweeper Box .....	7
4. Radio Drivers .....	8
General .....	8
Yaesu FT-847 .....	8
Simulated Radio .....	8
For developers: Writing new JHamTune radio drivers .....	8
Interface JHTRadio .....	8

---

## List of Figures

3.1. Main Window .....	3
3.2. Tuner .....	5
3.3. Scanner .....	5
3.4. Sweeper .....	7

---

---

# Chapter 1. Installation

## The Preview Edition

The Preview Edition of JHamTune (packaged in the file `JHamTuneDemo.jar`) has only one radio driver (Yaesu FT-847) and no means of configuring other drivers or saving driver configuration parameters. This is because at this time there is only one working driver, and the driver configuration code has not yet been written. Until several radio drivers have been developed, the driver model and interface may be in a state of flux.

To launch the Preview Edition, include the name of the serial port to which your FT-847 CAT port is attached as a command line parameter, for example:

```
java -jar JHamTuneDemo.jar COM2 or  
java -jar JHamTuneDemo.jar /dev/ttyS1
```

## System Requirements

### Java Runtime

The current version of JHamTune was developed and tested with Version 1.4.2 of the Sun Microsystems Java 2 Standard Edition System Developer's Kit (J2SE SDK). It requires a compatible Java 2 Runtime Environment. If your computer doesn't have a Java runtime, visit <http://java.com> [<http://java.com>]

### Java Libraries

#### **javax.comm**

Javacomm provides platform-independent access to parallel and serial ports. The Sun Javacomm page: <http://java.sun.com/products/javacomm> [<http://java.sun.com/products/javacomm>] On Linux platforms,

#### **javax.help**

Access to this user manual from the Help menu is provided with JavaHelp. The Sun Java Help page: <http://java.sun.com/products/javahelp> [<http://java.sun.com/products/javahelp>] If the `jh.jar` file is not found during help initialization, a warning message will be logged and the User Manual menu choice will be unavailable.

## Configuration

---

## Chapter 2. Future Directions

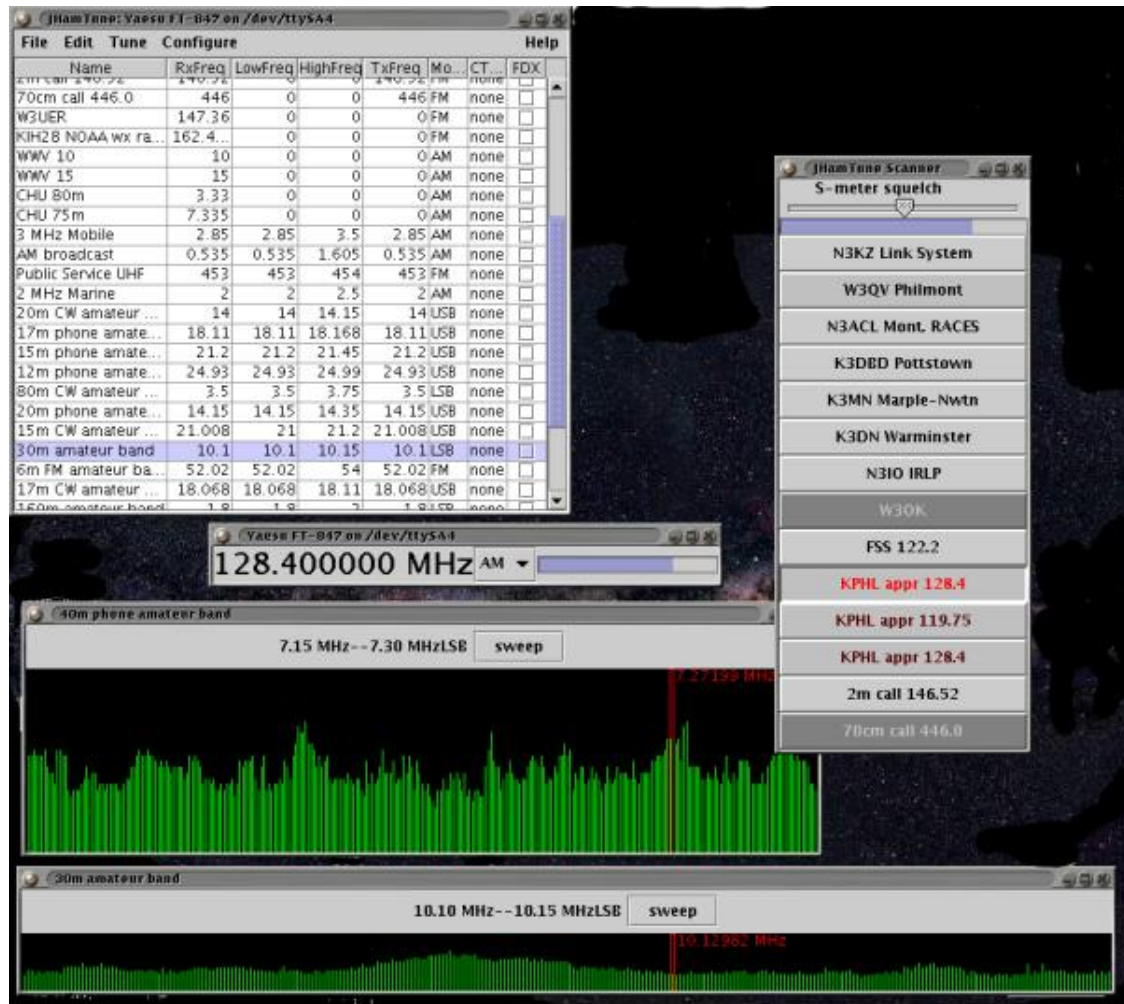
Development of new JHamTune features will be based on directions of interest to the program developers and also on suggestions from users. Some features under consideration are:

- Satellite tracking and pass prediction
- Scheduled contact management and logging, including an interface to the ARRL Logbook of the World™ system.
- Exchangable frequency database information
- Signal recording and analysis
- Direct support for use of digital modes: packet, PSK-31, etc

Let us know at <k3xs@arrl.net> about your own desires, ideas, or interest in participating as a developer.

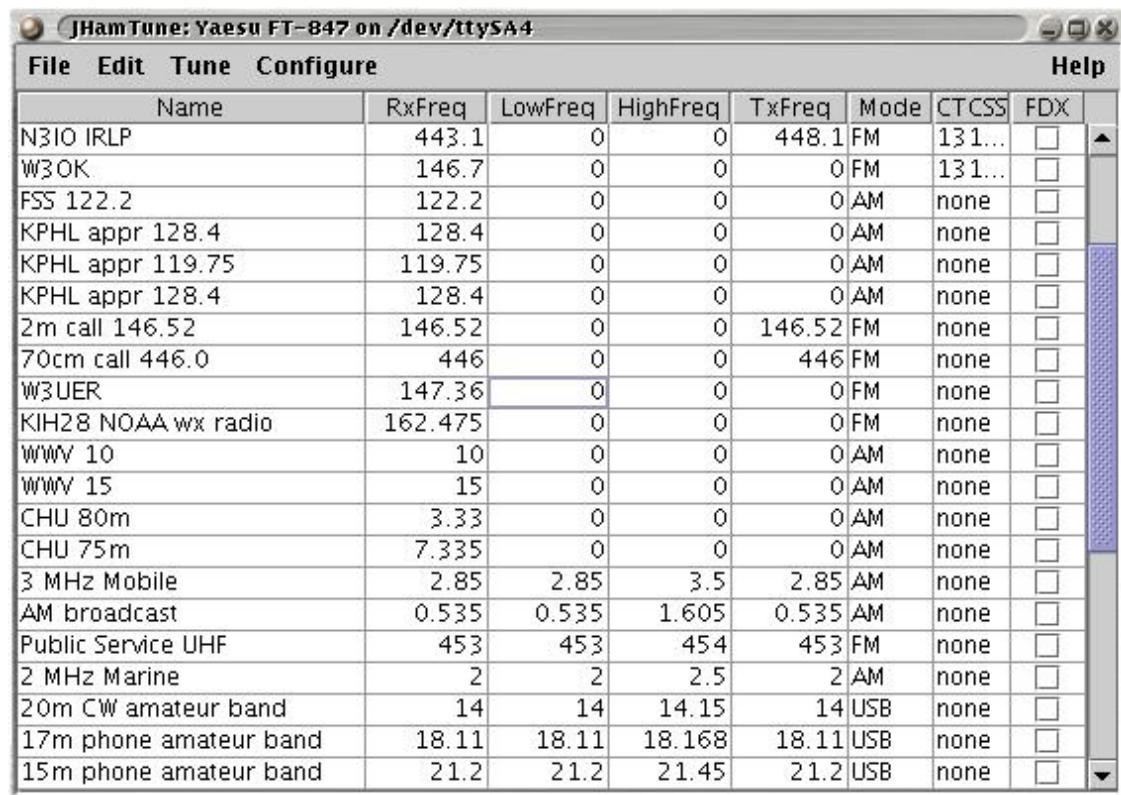


# Chapter 3. Operation



## Main Window

Figure 3.1. Main Window



The screenshot shows the JHamTune application window titled "JHamTune: Yaesu FT-847 on /dev/ttySA4". The window has a menu bar with "File", "Edit", "Tune", "Configure", and "Help". Below the menu bar is a table of tuner settings. The table has columns for Name, RxFreq, LowFreq, HighFreq, TxFreq, Mode, CTCSS, and FDX. The settings include various amateur radio bands and modes, such as N3IO IRLP, W3OK, FSS 122.2, KPHL appr 128.4, 2m call 146.52, 70cm call 446.0, W3UER, KIH28 NOAA wx radio, WWV 10, WWV 15, CHU 80m, CHU 75m, 3 MHz Mobile, AM broadcast, Public Service UHF, 2 MHz Marine, 20m CW amateur band, 17m phone amateur band, and 15m phone amateur band.

Name	RxFreq	LowFreq	HighFreq	TxFreq	Mode	CTCSS	FDX
N3IO IRLP	443.1	0	0	448.1	FM	131...	<input type="checkbox"/>
W3OK	146.7	0	0	0	FM	131...	<input type="checkbox"/>
FSS 122.2	122.2	0	0	0	AM	none	<input type="checkbox"/>
KPHL appr 128.4	128.4	0	0	0	AM	none	<input type="checkbox"/>
KPHL appr 119.75	119.75	0	0	0	AM	none	<input type="checkbox"/>
KPHL appr 128.4	128.4	0	0	0	AM	none	<input type="checkbox"/>
2m call 146.52	146.52	0	0	146.52	FM	none	<input type="checkbox"/>
70cm call 446.0	446	0	0	446	FM	none	<input type="checkbox"/>
W3UER	147.36	0	0	0	FM	none	<input type="checkbox"/>
KIH28 NOAA wx radio	162.475	0	0	0	FM	none	<input type="checkbox"/>
WWV 10	10	0	0	0	AM	none	<input type="checkbox"/>
WWV 15	15	0	0	0	AM	none	<input type="checkbox"/>
CHU 80m	3.33	0	0	0	AM	none	<input type="checkbox"/>
CHU 75m	7.335	0	0	0	AM	none	<input type="checkbox"/>
3 MHz Mobile	2.85	2.85	3.5	2.85	AM	none	<input type="checkbox"/>
AM broadcast	0.535	0.535	1.605	0.535	AM	none	<input type="checkbox"/>
Public Service UHF	453	453	454	453	FM	none	<input type="checkbox"/>
2 MHz Marine	2	2	2.5	2	AM	none	<input type="checkbox"/>
20m CW amateur band	14	14	14.15	14	USB	none	<input type="checkbox"/>
17m phone amateur band	18.11	18.11	18.168	18.11	USB	none	<input type="checkbox"/>
15m phone amateur band	21.2	21.2	21.45	21.2	USB	none	<input type="checkbox"/>

The main window holds a list of tuner settings, and a menu bar.

Each tuner setting contains

- a title
- a receive frequency
- a modulation mode

and optionally

- a transmit frequency for full-duplex or split frequency operation
- a CTCSS (also called "PL tone") channel
- high and low frequencies specifying band limits for band sweeping operation

The list is editable; new entries can be created with the Edit...Insert menu, and deleted with Edit...Delete. Lists can be saved as disk files and reloaded using the file menu.

## Menus

### File

The File menu has submenus for loading and saving frequency data tables. "Get Bands from radio" will load a set of frequency tables giving the basic capabilities of the connected radio; this information is provided by the radio driver.

The Save dialog has a checkbox item labelled "Export in XML format". Checking this box will write an XML-encoded copy of the current frequency list to the output file. This XML-encoded version should be portable across different versions of JHamTune. The Open... function will automatically detect and process the XML-encoded version.

## Edit

The Edit menus allow you to create new or delete entries in the current frequency list. Insert creates a new empty entry, while Snapshot creates a new entry with frequency and mode information copied from the radio's current setting. Delete will delete the selected list entry. To edit individual fields on an entry, click the field with your mouse.

## Tune

The Tune menu can create a Sweeper box from the current row in the frequency list, or create a scanner window from multiple selected frequencies in the list. It can also tune the radio directly to the selected frequency.

## Help

The Help menu lets you display either this usermanual, or an "about" box that gives you version and copyright information. (In the event that the JavaHelp library can't be found during startup, the user manual will not be available, see the section called "javax.help")

# Tuner Display

**Figure 3.2. Tuner**



The Tuner window displays the current frequency and modulation mode of the radio, along with a horizontal bar depicting the signal strength as displayed by the radio's S-meter.

# Scanner Window

**Figure 3.3. Scanner**



The Scanner Window displays a collection of channel buttons representing a selection of frequencies from the Main Window, an S-meter, and a slider that sets a virtual squelch level.

JHamTune listens on each frequency in turn looking for a signal that exceeds the squelch level, and when it finds one it stays tuned to that frequency until either the signal drops or the channel is locked out with a single mouse click. When a channel breaks squelch, the button title turns bright red. When the scan resumes, each time that channel is found to be quiet, the red color is slightly darkened. By observing the relative colors, you can judge which channels have been the most active recently.

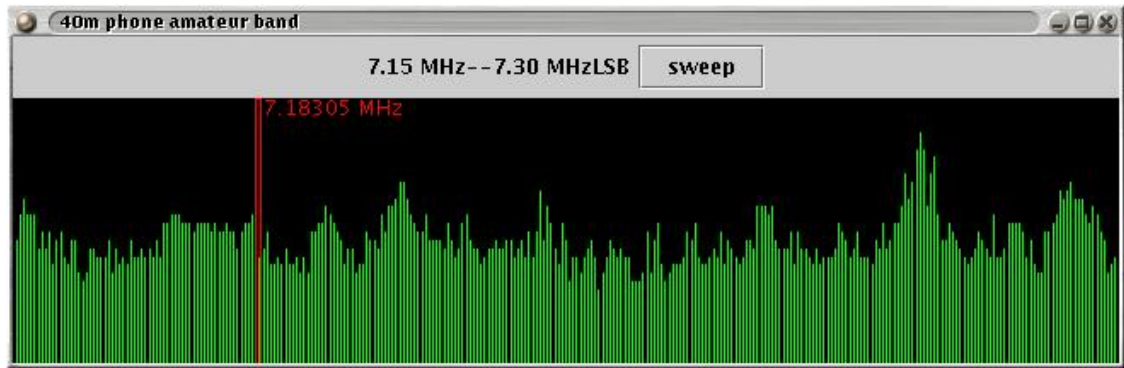
A single mouse click will remove a channel temporarily from the scan, useful when the frequency is busy but not currently of interest. These "locked-out" channels show a darker background.

You can force the scanner to listen to one channel with a double mouse click; the button will turn white, that frequency and mode will be tuned, and scanning will stop. To resume scanning, click on the locked channel once; scanning will resume and the channel will be locked out of the scan. To return the locked-

out channel to the scan click on it again

## Sweeper Box

**Figure 3.4. Sweeper**



The Sweeper Box tunes a radio in narrow steps over a defined frequency band and records the observed signal strength as reported by the radio's signal strength meter. When the sweeping process is complete, individual signal peaks can be tuned by clicking with the mouse on the peak, and a vertical red-line cursor shows the last frequency tuned by this method.

---

# Chapter 4. Radio Drivers

## General

JHamTune is designed around a modular driver scheme. A JHamTune driver is a Java class implementing the `JHTRadio` interface for a particular radio, allowing the same user interface to be used with any supported radio, within the limits of that radio's capabilities.

The original JHamTune driver was designed for my Yaesu FT-847 radio.

## Yaesu FT-847

## Simulated Radio

The `SimulatedRadio` driver does not control an actual radio. It's intended to be used for demonstrating and testing JHamTune when an actual radio is not available. Inputs don't actually do anything, and outputs are pretty much random.

## For developers: Writing new JHamTune radio drivers

JHamTune drivers are responsible for translating method calls on the `JHTRadio` interface into the appropriate signals to the attached radio. Both the API JavaDoc for the `JHTRadio` interface and the source code for the Yaesu FT-847 driver can serve as guides for writers of new driver classes. The CAT command set of the FT-847 appears in the driver source code as comments. Developers of new drivers are invited to contribute them to the JHamTune distribution, contact Maggie at <k3xs@arrl.net>.

## Interface JHTRadio

an interface implemented by drivers for each **JHamTune** supported radio.

## Synopsis

```
package com.k3xs.JHamTune;

public interface JHTRadio {

    // Public Methods
    public void connect();
    public void disconnect();
    public void setFreq(int freq);
    public void setBand(com.k3xs.JHamTune.JHTBand band);
    public int getFreq();
    public void setMode(com.k3xs.JHamTune.JHTMode mode);
    public void setTone(com.k3xs.JHamTune.JHTCTCSS tone);
    public JHTMode getMode();
    public int getSmeter();
    public String getDriverName();
    public String getPortName();
    public String getName();
```

```
    public Collection getRxBands();  
    public Collection getRxBands();  
}
```

**Inheritance Path.** com.k3xs.JHamTune.JHTRadio

## Members

### Method connect()

*Synopsis:* public void **connect**();

#### Description

connect to the radio

### Method disconnect()

*Synopsis:* public void **disconnect**();

#### Description

disconnect automatic control from the radio

### Method getDriverName()

*Synopsis:* public String **getDriverName**();

#### Parameters

*return*      the name for this radio

### Method getFreq()

*Synopsis:* public int **getFreq**();

#### Parameters

*return*      the current frequency in Hz

## Method **getMode()**

*Synopsis:* `public JHTMode getMode();`

### Parameters

*return*      the current modulation mode

## Method **getName()**

*Synopsis:* `public String getName();`

### Parameters

*return*      the name for this instance, usually `driverName + " on " + port`

## Method **getPortName()**

*Synopsis:* `public String getPortName();`

### Parameters

*return*      the name for the port attaching this radio, or null if not connect()ed

## Method **getRxBands()**

*Synopsis:* `public Collection getRxBands();`

### Parameters

*return*      a Collection of receive-only JHTBands



## Method `getRxTxBands()`

*Synopsis:* `public Collection getRxTxBands();`

### Parameters

*return*      a Collection of transceive-capable JHTBands

## Method `getSmeter()`

*Synopsis:* `public int getSmeter();`

### Parameters

*return*      the current signal strength (S-meter reading expressed 0-31)

## Method `setBand(JHTBand)`

*Synopsis:* `public void setBand(com.k3xs.JHamTune.JHTBand band);`

### Description

set mode and band limits from a JHTBand.

## Method `setFreq(int)`

*Synopsis:* `public void setFreq(int freq);`

### Parameters

*freq*                      tune the radio to this frequency in Hz.

## Method `setMode(JHTMode)`

*Synopsis:* `public void setMode(com.k3xs.JHamTune.JHTMode mode);`

## Parameters

mode                      set the radio to this modulation mode.

## Method **setTone(JHTCTCSS)**

*Synopsis:* public void **setTone**(com.k3xs.JHamTune.JHTCTCSS *tone*);

## Parameters

tone                      set this CTCSS/PL tone encoding.