

# FreeCast User's Guide

---

# **FreeCast User's Guide**

Published Thursday, June 29 2006  
Copyright © 2005-2006 Alban Peignier

---

---

---

---

---

## Table of Contents

Foreword .....	xi
1. What's FreeCast .....	1
Architecture overview .....	1
Classic broadcast .....	1
Peer-to-peer broadcast .....	1
Key concepts .....	1
2. Getting started .....	3
FreeCast Manager introduction .....	3
Your first FreeCast network .....	3
General GUI layout .....	3
Actions .....	4
Setup Dialog .....	5
Receiver configuration .....	6
Default configuration .....	6
Troubleshooting .....	6
Network configuration .....	7
Support team .....	7
3. Installation .....	9
Binary distribution .....	9
Content .....	9
Getting started .....	9
Debian package distribution .....	10
Content .....	11
Getting started .....	11
4. Tracker usage .....	13
Description .....	13
Network accessibility .....	13
Connection Assistant Service .....	13
Configuration .....	13
General layout .....	13
Connector .....	13
ConnectionAssistant .....	14
Configuration sample .....	14
Command line usage .....	14
5. Node usage .....	17
Description .....	17
Peer provider .....	17
Sender .....	17
Reference .....	17
Receiver .....	18
Stream validation .....	18
Connection Assistant .....	19
Configuration .....	20
General layout .....	20
Peer Provider .....	20
Sender .....	20
Receiver .....	21
Connection Assistant .....	23
GUI Configuration .....	23
Player .....	23
Configuration samples .....	24
Audio FreeCast network .....	24
Video FreeCast network .....	25
Command line usage .....	26
Swing usage .....	26
Main dialog .....	26
6. Configuration .....	29

Common configuration elements .....	29
Network elements .....	29
7. JavaWebStart™ deployment .....	31
Overview .....	31
JavaWebStart™ in few words .....	31
Benefits for FreeCast deployment .....	32
Stream description file .....	32
Stream description option list .....	32
Advanced examples .....	33
Start page .....	34
8. Source samples .....	35
Source clients .....	35
Ices .....	35
OddCast .....	36
StreamTranscoder .....	36
Source servers .....	36
IceCast .....	36
Flumotion .....	36
I. Reference Pages .....	37
freecast .....	39
A. GNU General Public License .....	41
Preamble .....	41
TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION .....	42
Section 0 .....	42
Section 1 .....	42
Section 2 .....	42
Section 3 .....	43
Section 4 .....	43
Section 5 .....	43
Section 6 .....	44
Section 7 .....	44
Section 8 .....	44
Section 9 .....	44
Section 10 .....	45
NO WARRANTY Section 11 .....	45
Section 12 .....	45
How to Apply These Terms to Your New Programs .....	45
Index .....	47

---

## List of Tables

3.1. Binary distribution content .....	9
3.2. Debian distribution content .....	11
7.1. Stream description .....	32

---

---

## List of Examples

4.1. Tracker default configuration .....	14
5.1. Audio root node configuration sample .....	24
5.2. Audio listener node configuration sample .....	25
5.3. Start a FreeCast network with the Audio configuration sample .....	25
5.4. Video root node configuration sample .....	25
5.5. Video listener node configuration sample .....	26
5.6. Start a FreeCast network with the Audio configuration sample .....	26
6.1. Host configurations .....	29
6.2. Port configurations .....	29
7.1. Minimalist stream description .....	32
7.2. Streaming description with a node configuration file .....	33
7.3. A possible node configuration file .....	33
7.4. Use the start page in a web page .....	34
8.1. Ices2 sample configuration .....	35

---

---

---

# Foreword

This document is mainly dedicated to the broadcasters who want to deploy their own FreeCast network. It should be interesting for everyone who want to know more about FreeCast.

This document is available on the FreeCast website at <http://www.freecast.org/userguide>. A PDF version [<http://www.freecast.org/userguide/userguide.pdf>] is available too. This release is always associated to the current FreeCast release.

Your comments are welcome, please contact the FreeCast support team: <http://www.freecast.org/support>.

The latest release of this document is available during its writing into the development project documentation at <http://www.kolaka.org/freecast/docs/userguide>.

---

---

# Chapter 1. What's FreeCast



FreeCast streams content by using a peer-to-peer network model. It allows to broadcast a stream to a large number of listeners from a simple DSL connection. FreeCast offers the possibility to broadcast over the Internet without a strict limit of listener count. You can share with the largest number without the economic cost of a large bandwidth.

Visit the FreeCast website: <http://www.freecast.org/>

## Architecture overview

To understand how to use and deploy a FreeCast network, you need to understand a few key concepts. A FreeCast network is able to broadcast a stream, which can contain audio and/or video contents. The stream won't be modified by the FreeCast broadcasting. The way and tools used to create and play the stream are near the same than in a classic broadcast.

### Classic broadcast

Before explaining the the concept of peer-to-peer broadcast, let's look at what is a classic broadcast. In a classic broadcast, there are two roles: server and client. The server has the information to be streamed available and sends them directly to each client. That makes things simple for the client, which easily connects and retrieves information with the URL of the server. When the server bandwidth is wide enough, this classic broadcast performs well. The quality of service depends mostly of the Internet access quality of the client.

Things become more difficult for the server as the client count grows. The server must send the same information to each client, and all these data must be sent with the single Internet connection of the server. As such, the needed infrastructure becomes quickly expansive. As an example, if you want to accept up to 50 listeners with a musical stream (typically encoded at 128kbits), you need an Internet connection with an upload bandwidth larger than 6.25 Mbits. As a comparison, a classic DSL access provides an upload bandwidth of 0.5Mbits ...

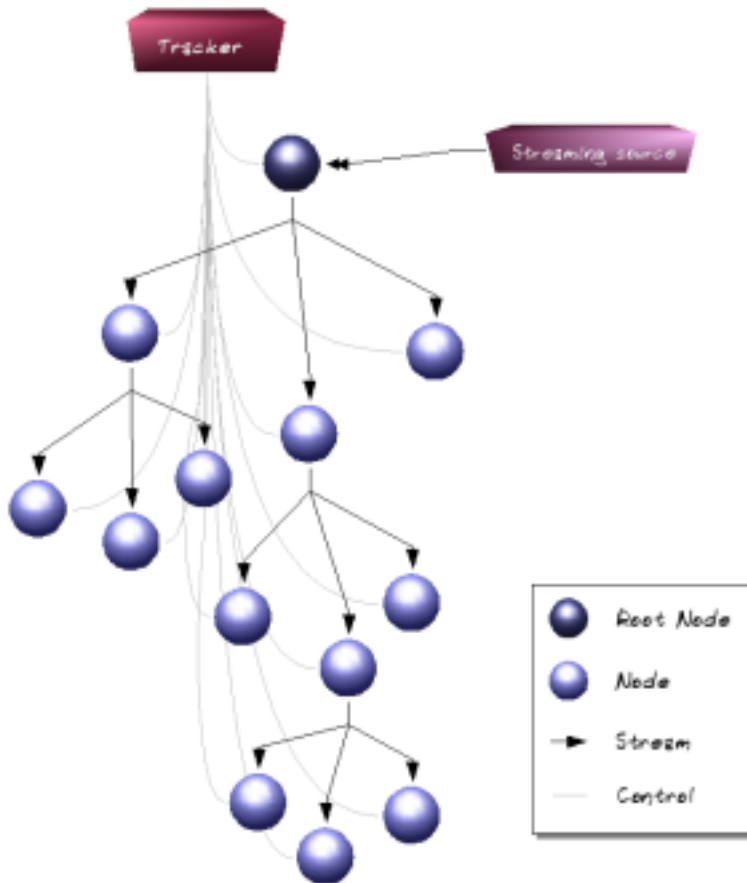
### Peer-to-peer broadcast

The peer-to-peer broadcast is based on a simple question: *why clients which are retrieving (and listening to) the stream should stay passive ?* Nowadays, most of the stream listeners have a DSL access. Instead of concentrating all clients on the server, it would be better if the clients which are listening to the stream relay it to a few other clients. This is what is achieved with a peer-to-peer broadcast.

This collaborative approach has evident benefits. From the broadcaster side, the server bandwidth is less loaded and the client count is not limited. For instance, a broadcast can be started from a simple DSL access. The streaming infrastructure cost can be reduced in a significant manner. From the client side, benefits may be indirect but not null, because if the peer-to-peer model makes things easier for the broadcaster, it allows the clients to find more streams, more choices, less advertising. A collaborative approach can be initiated between broadcaster and clients: clients can support the broadcaster by relaying the stream between their listenings.

### Key concepts

A FreeCast network is composed of a *tracker*, a *root node* and *nodes*.



## Node

In a FreeCast network, the stream transits via one or more nodes. Each node can receive, use (play) and send stream information (packets). All the nodes of a given FreeCast network register themselves to the same tracker.

## Root Node

The root node creates the FreeCast stream (a sequence of packets) from a classic Ogg stream. The following nodes receive directly or indirectly the packets created by the root node.

## Tracker

The tracker has an overview of the network. A FreeCast node can only participate to a FreeCast network once it is registered to the tracker of this network. All the nodes of the network send information to the tracker periodically. The nodes use the tracker to obtain references to other nodes.

---

# Chapter 2. Getting started

## FreeCast Manager introduction



FreeCast Manager is a graphical application which allows to start a FreeCast network. It provides:

- a FreeCast tracker
- a FreeCast root node
- an embedded HTTP server

All these components are configured automatically when the FreeCast Manager starts.

FreeCast Manager can be started via JavaWebStart. In this case, it will be installed and upgraded automatically via Internet.

## Your first FreeCast network

1. You need to have a Java Runtime Environment 1.4 or 1.5 installed, see <http://www.java.com>
2. Start [<http://download.freecast.org/jws/libfreecast/freecast-manager.jnlp>] FreeCast Manager via JavaWebStart
3. When loaded, the FreeCast Manager starts, configures the FreeCast network and displays the GUI.

## General GUI layout



The FreeCast Manager GUI regroups the activity indicators of the embedded tracker and root node.

## Tracker

The node count is displayed in realtime. At startup, a single node is counted: it's the embedded root node.

## Root Node

For the root node, the connected node count is displayed. The first listener nodes will be displayed in this count, the following nodes will be connected to these first nodes.

## Actions



The action menu can be displayed by right-clicking on the GUI. This menu allows the following actions:

### Browse network homepage

Starts your browser to visit the web homepage of the FreeCast network. This web homepage is hosted by the embedded web server provided by the FreeCast Manager. This page will be used by listeners to easily start their nodes.

Don't publish the URL displayed by your browser, use the Email network homepage action instead.

### Email network homepage

Creates a base email to inform your first listeners. The created email contains the URL to be used to visit the homepage of your FreeCast network.

Subject: My FreeCast Network

Visit the homepage of my FreeCast network: `http://84.67.103.24:8080/`

Complete and send this email to your first listeners. They just have to visit the given URL and click on the start button.



#### Different URLs and some don't work

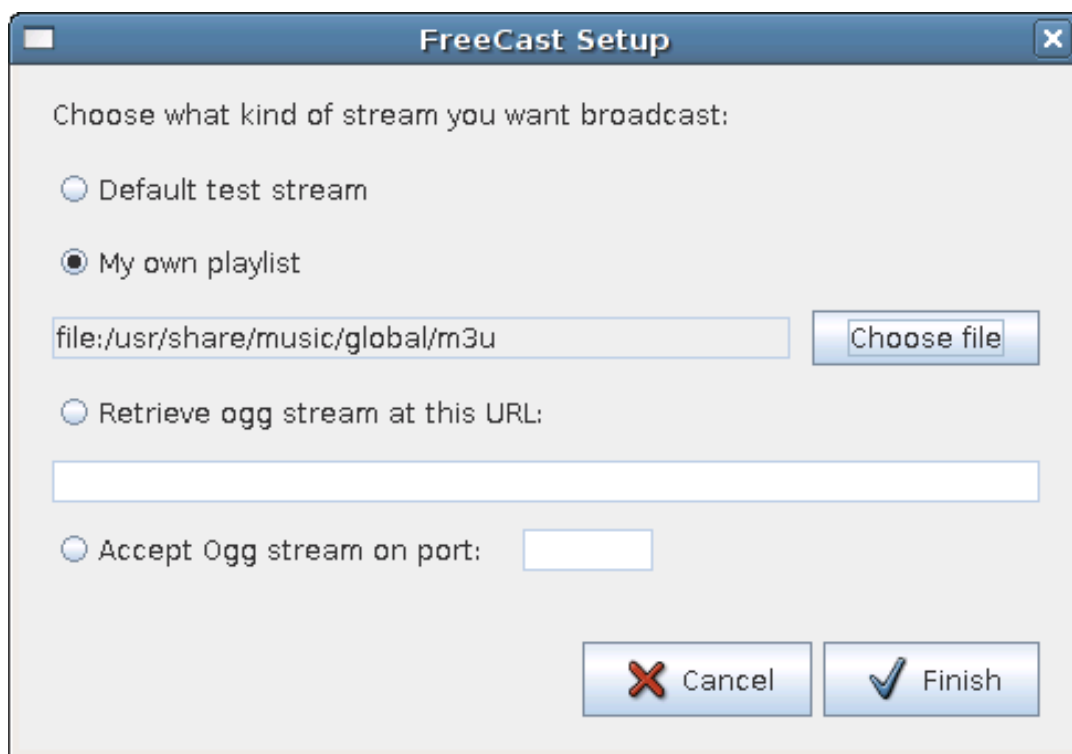
According to your network configuration, the FreeCast Manager can use several URLs to designate the homepage of your FreeCast network. The main cause is that you can have to use a different URL than your listeners to access to your own computer. The FreeCast manager respects this constraint. Take care to use the URL specified in the email to invite your listeners.

### Configure FreeCast

Select this menu to start the Setup Dialog, see the section called "Setup Dialog".

## Setup Dialog

The Setup Dialog allows to configure the FreeCast network supported by FreeCast Manager.



## Receiver configuration

You can select the content you want broadcast.

### Default test stream

Select this content to perform your first tests.

### My own playlist

Select a M3U playlist with the ogg, wav or mp3 files to be broadcast.

### Retrieve the Ogg stream

Specify the URL where the FreeCast root node can retrieve the Ogg stream.

### Receive the Ogg stream

Specify the port where the FreeCast root node will receive the Ogg stream sent by a third application.

## Default configuration

The FreeCast Manager will configure for you:

- a tracker on the port 1665
- a HTTP server which listens on the port 80 or on the first free ports between 8080 and 8100

## Troubleshooting

## Network configuration

*symptoms: listeners can access to the embedded HTTP server, but the start page fails*

As plug-and-play networks are still more a concept than a reality, you can have to configurate your network (router, firewall, etc ...) in order to make the FreeCast Manager services visible to your listeners (generally on the Internet). By default, you need to open and/or forward the TCP ports 80 and 1665.

## Support team

If you experience any problem or if you have any question about FreeCast and FreeCast Manager, contact the FreeCast support team. Visit <http://www.freecast.org/support>.

---

---

---

# Chapter 3. Installation

There are several ways to install FreeCast on a system.

## Binary distribution

1. install a Java Runtime Environment 1.4 or 1.5, see <http://www.java.com>
2. download the latest release from <http://download.freecast.org>
3. extract the FreeCast files from the downloaded archive

## Content

**Table 3.1. Binary distribution content**

Filename	Description
bin	contains Unix and Windows scripts, see Reference Pages
etc	contains configuration files, see the section called "Configuration"
lib	contains all needed Java libraries files
docs/userguide	contains this user guide
docs/examples/audio	contains configuration sample files for an audio stream
docs/examples/video	contains configuration sample files for a video stream
docs/examples/tracker	contains configuration sample files for tracker
docs/examples/jws	contains configuration sample files for a JavaWebStart™ deployment, see Chapter 7, <i>JavaWebStart™ deployment</i>

## Getting started

Once the FreeCast binaries are installed, you can start your first FreeCast network. This is a small procedure to start a FreeCast tracker, a root node and a first listener.

Start a command line and go to the directory where the FreeCast binaries are installed.

### Start a FreeCast tracker

The FreeCast tracker doesn't require specific configuration. Just start the **freecast-tracker**:

```
[host:path/to/freecast] bin/freecast-tracker
create log dir: host:path/to/freecast/log
INFO [Main]: version 20060629
INFO [ConfigurationFactory]: Trying to load configuration defaults-tracker
INFO [Main]: start a HttpTracker on port /0.0.0.0:1665
INFO [ConnectionAssistantServer]: start ConnectionAssistant server on /0.0
```

## Start a FreeCast root node

When the tracker is running, that's the moment to start the root node. In this example, we will use a default playlist provided by the FreeCast website as source. This stream will be transmitted by your FreeCast network.

```
[host:path/to/freecast] bin/freecast -config docs/examples/audio/freecast-node-root
INFO [Main]: version 20060629
INFO [ConfigurationFactory]: Trying to load configuration defaults-node.xml
INFO [NodeConfigurator]: set the public reference to [/x.x.x.x:30932{}], /192.168.0.2:30932
WARN [NodeConfigurator]: the playlist receiver uses a static bandwidth controller
INFO [DefaultNodeService]: received node identifier #32BF31F4C21882D9
INFO [ConnectionAssistantClient]: connect to ConnectionAssistant at cas.freecast
INFO [ConnectionAssistantServiceStub]: new session
```

The default node configuration uses the tracker on the same host (localhost). The command line specifies the use of a sample configuration file. It will use a default playlist provided by the FreeCast website.

The tracker should have logged the following messages when the root node registers:

```
INFO [Tracker$Auditor]: new registered node [/x.x.x.x:30932{}], /192.168.0.2:30932
INFO [Tracker$Auditor]: 1 connected nodes
```

## Start a first FreeCast listener node

A FreeCast network without a listener isn't really a network .. so let's start a first FreeCast listener node.

```
[host:path/to/freecast] bin/freecast-swing -config docs/examples/audio/freecast-node-listener
INFO [Main]: version 20060629
INFO [NodeConfigurator]: set the public reference to [/x.x.x.x:30680{}], /192.168.0.2:30680
INFO [DefaultNodeService]: received node identifier #32C5AE480DADE3F1
INFO [ConnectionAssistantClient]: connect to ConnectionAssistant at cas.freecast
INFO [ConnectionAssistantServiceStub]: new session
```

The listener node connects to the tracker, receives the references to other peers, then tries to connect to one of them (the root node in your case). When the first streamed data are received, the audio player starts to play sound.

The tracker should have logged the following messages when the listener node registers:

```
INFO [Tracker$Auditor]: new registered node [/x.x.x.x:30932{}], /192.168.0.2:30932
INFO [Tracker$Auditor]: 2 connected nodes
```

The root node should have logged the following message when the listener node connects to it:

```
INFO /192.168.0.2:30932 [PeerController$Auditor]: accept connection with DefaultPeer
INFO /192.168.0.2:30932 [PeerController$Auditor]: 1 peers connected
```

## Debian package distribution

You need to have a Java Runtime Environment 1.4 or 1.5 installed, see Java under Debian [[http://wiki.tryphon.org/Java\\_under\\_Debian](http://wiki.tryphon.org/Java_under_Debian)]. Other dependencies are available via packages present into the Debian releases.

The FreeCast debian package is available on the Tryphon Debian archiver [<http://debian.tryphon.org>]. Follow the archiver configuration instructions and install the FreeCast package:

```
[host:~] sudo apt-get install freecast
Reading Package Lists... Done
Building Dependency Tree... Done
The following extra packages will be installed:
  libcommons-cli-java libcommons-collections3-java libcommons-lang-java libcommons-logging3-java
The following NEW packages will be installed:
  freecast libcommons-cli-java libcommons-collections3-java libcommons-lang-java
0 upgraded, 5 newly installed, 0 to remove and 1 not upgraded.
Need to get 3016kB of archives.
After unpacking 3618kB of additional disk space will be used.
Do you want to continue? [Y/n]
```

## Content

**Table 3.2. Debian distribution content**

Filename	Description
/etc/freecast	contains configuration files, see the section called “Configuration”
/var/log/freecast	contains log files as by freecast daemons
/etc/init.d/freecast-tracker	the init.d script to control the FreeCast tracker daemon
/etc/init.d/freecast	the init.d script to control the FreeCast node
/etc/default/freecast-tracker	default values for the FreeCast tracker daemon
/etc/default/freecast	default values for the FreeCast node
/usr/share/doc/freecast/userguide	contains this user guide
/usr/share/doc/freecast/examples/audio	contains configuration sample files for an audio stream
/usr/share/doc/freecast/examples/video	contains configuration sample files for a video stream
/usr/share/doc/freecast/examples/tracker	contains configuration sample files for the tracker
/usr/share/doc/freecast/examples/jws	contains configuration sample files for a JavaWebStart™ deployment, see Chapter 7, <i>JavaWebStart™ deployment</i>

The FreeCast debian package respects the Debian policy [<http://www.debian.org/doc/debian-policy/>]. So FreeCast binaries are available in /usr/bin and in the default PATH, see the Reference Pages.

## Getting started

If you want to deploy your own FreeCast network, you just need to provide a root node configuration.

```
[host:~] sudo cp /usr/share/doc/freecast/examples/audio/freecast-node-root.xm
[host:~] sudo /etc/init.d/freecast-tracker start
[host:~] sudo /etc/init.d/freecast start
```

## Getting started

---

You can follow the FreeCast daemons activity via the log files into `/var/log/freecast`:

```
[host:~] tail -f /var/log/freecast/freecast.log /var/log/freecast/freecast-tracker
```

## Relay node

If you just want to deploy a FreeCast relay node, you need to disable the FreeCast tracker daemon (see `/etc/default/freecast-tracker`) and to provide a node configuration with the tracker to be used.

---

# Chapter 4. Tracker usage

## Description

The main component which can be configured into the tracker is the `connector` that dialogs with every node of the FreeCast network. A `HTTP connector` implementation is available (the `RMI` implementation is obsolete now).

The second component is the `connection-assistant`, see the section called “Connection Assistant Service”.

As soon as the FreeCast tracker is started, nodes can be started.

The information about the FreeCast network managed by the tracker are lost if the tracker reboots. The FreeCast network will experience troubles in this case.

## Network accessibility

The tracker must be accessible for all the nodes that want to participate to the FreeCast network. The `connector` configuration must match the network configuration and the constraints of the computer where the tracker is hosted. If the tracker on which the `connector` is listening requests on the `listen` address `0.0.0.0:1665` (the default value) and if the nodes are on Internet, any firewall or router between the tracker host and Internet must be configured to let pass through or *forward connections* to the `connector` port.



### Track unreachable

If the FreeCast nodes "behind" a firewall, a router or such equipment encounter connection errors, you should verify the network configuration.

## Connection Assistant Service

The connection assistant service provided by the FreeCast tracker assists nodes to connect between them through NATs. Every node connected to the same service can request help to connect to another one. The connection assistant service will notify the other node in realtime that a NAT traversal session is starting.

By default the connection assistant service is started on the TCP port 1666. The default node configuration uses a default service available on `freecast.org` servers. To use your own service, you must make it reachable from Internet (may be with the required forward) and change the node configuration (see the section called “Connection Assistant”).

## Configuration

### General layout

```
<?xml version="1.0"?>
<freecast>
  <tracker>
    Connector
    ConnectionAssistant
  </tracker>
</freecast>
```

### Connector

## HTTP Connector

```
<connector>
  <class>http</class>
  <listenaddress>
    <host>0.0.0.0</host>
    <port>1665</port>
  </listenaddress>
</connector>
```

**listenaddress.** specifies where the connector (and so the tracker) is listening to requests sent by the nodes. By default, the connector is listening on every available IP address (0.0.0.0) and the TCP port 1665.

## ConnectionAssistant

```
<connection-assistant>
  <listenaddress>
    <port>1666</port>
  </listenaddress>
</connection-assistant>
```

**listenaddress.** specifies where the connection assistant is listening to requests sent by the nodes to request assistance to traverse NATs. By default, the connection assistant is listening on every available IP address (0.0.0.0) and the TCP port 1666.

## Configuration sample

This is the default configuration of the FreeCast tracker.

### Example 4.1. Tracker default configuration

```
<freecast>
  <tracker>
    <connector>
      <class>http</class>
      <listenaddress>
        <host>0.0.0.0</host>
        <port>1665</port>
      </listenaddress>
    </connector>
    <connection-assistant>
      <listenaddress>
        <host>0.0.0.0</host>
        <port>1666</port>
      </listenaddress>
    </connection-assistant>
  </tracker>
</freecast>
```

## Command line usage

The FreeCast tracker can be started in command line via the command **freecast-tracker** (see freecast(1)).

```
[host:path/to/freecast] bin/freecast-tracker
create log dir: host:path/to/freecast/log
INFO [Main]: version 20060629
```

## Command line usage

---

```
INFO [ConfigurationFactory]: Trying to load configuration defaults-tracker
INFO [Main]: start a HttpTracker on port /0.0.0.0:1665
INFO [ConnectionAssistantServer]: start ConnectionAssistant server on /0.0
```

---

---

---

# Chapter 5. Node usage

## Description

To use and configure the FreeCast node, you need to understand a part of the internal components.

## Peer provider

The `peerprovider` component manages the connection between the node and the tracker. It must be configured with the tracker's TCP address.

## Sender

The `sender` component listens to requests from other nodes (peers), manages the opened connections and controls the sending of data. The `udp_sender` implementation uses a TCP socket by default. The `none_sender` implementation disables the node relaying (.. discouraged).

## UDP sender

The `udp_sender` implementation waits for UDP connections initiated by other nodes. Once a connection is opened with another node (peer), the sender forwards the received streamed data to this peer.

Via the `connection assistant`, the `udp_sender` is reachable even behind a NAT (NAT traversal).

## Socket sender

The `socket_sender` is obsolete now. Use an `udp_sender`.

## Reference

The node reference is used to localize it over the network. The default reference specifies the IP address and UDP port where the node waits for connections from other nodes (see the section called "UDP sender").

## stun reference

By using `stun` reference, the FreeCast node determinates its public address (and natted UDP port) via a STUN [<http://en.wikipedia.org/wiki/STUN>] request.

The `stun` reference configuration allows to specify the STUN server to be used.

Node local addresses will be determinated automatically to make the node reachable in the local network(s).



### NAT traversal

A node must use a `stun` reference to be reachable behind a NAT and make NAT traversal possible.

## inet reference

The `reference` can be specified manually via an `inet` reference by giving host and port.

Take care by using an `inet` reference, if the specified information is wrong, the node won't be

reachable.

## multiple reference

You can specify several references. It allows the node to be available from several IP addresses. For instance, you can specify a reference with the public IP address, and another with a local network address. Your node will be available both on Internet and on your local network.

## auto reference

The `auto` reference is obsolete, use `stun` reference instead. See the section called “`stun` reference”.

## Receiver

The `receiver` component manages the stream reception. According to the implementation, the stream can be received from other nodes (peers) or created by the node (root node) from a classic Ogg stream.

### Peer receiver

The `peer` receiver component receives the stream from other nodes (peers). It doesn't require any configuration. It will use the tracker to find other nodes.

### Shoutclient receiver

The `shoutclient` receiver component opens a connection on a shout server (like icecast or fluedo) to retrieve an audio or video ogg stream.

The `shoutclient` receiver configuration defines the URL where the stream can be received.

### Shoutserver receiver

The `shoutserver` receiver component acts like a shout server. It can receive an audio or video ogg stream sent by a tool like ices2 or oddcast.

The `shoutserver` receiver configuration defines the listen address used to wait for the shout connection.

### Playlist receiver

The `playlist` receiver component uses a M3U [<http://en.wikipedia.org/wiki/M3u>] playlist which lists the ogg files to be read (a file per line).

The `playlist` receiver configuration contains the URL where the playlist can be found.

### Encoder-Playlist receiver

The `encoder-playlist` receiver component uses a M3U [<http://en.wikipedia.org/wiki/M3u>] playlist which lists the files to be read (a file per line). By default, wav, ogg and mp3 files are supported.

The Ogg stream is created via a native library which depends of the runtime platform. For the moment, the library is available for linux x86 and windows platforms.

The `encoder-playlist` receiver configuration contains the URL where the playlist can be found and stream encoding parameters like wanted Ogg quality, channel count and sample rate.

## Stream validation

The `checksummer` and `validator` components are used to validate the relayed nodes into each listener node. The `checksummer` is deployed into the root node in order to add a verifiable information to each packet (checksum). The `validator` is used by each node listener to verify the checksum of each packet. According to the chosen implementation, it allows to verify that:

- the network transfer doesn't corrupt the packet
- the authentication is created by the root node

Every packet that is not validated by the listener node will be skipped. It won't be played and relayed.



### Root and listener compatibility

Remember that the `validator` configuration of the listener node must be compatible with the `checksummer` configuration of the root node. Otherwise, all packets will seem invalid to misconfigured listeners.

## Digest checksummer and validator

Deployed into the root node, the `digest checksummer` adds a checksum to each packet (using the SHA algorithm).

Deployed into the listener nodes, the `digest validator` verifies that the packet data still matches this checksum.

No specific configuration is needed

## Signature checksummer and validator

Deployed into the root node with a given private key, the `signature checksummer` signs each packet with an encrypted checksum (using an RSA private key with the SHA algorithm).

Deployed into the listener nodes with the associated public key, the `signature validator` verifies that the packet data still matches this digital signature.

To create a key pair, use the `freecast-keygenerator` tool:

```
[host:~] freecast-keygenerator
Generate a RSA 1024 keypair
Write private key to freecast-private.key
Write public key to freecast-public.key
Take care to the private key file security
```

Then, you can use the two created files. Take care to protect the private key file.

## None checksummer and validator

These implementations allow to disable the checksum creation or validator into the nodes. No specific configuration is needed.

## Connection Assistant

The FreeCast nodes can connect through NATs by using the connection assistance service (which is started into the FreeCast tracker).

By default, FreeCast nodes use the connection assistance service available at [cas.freecast.org](http://cas.freecast.org) on port 1666. The node configuration allows to specify the wanted connection assistant. You can use the connection assistant service provided by your tracker. See the section called "Connection Assistant

Service”.

## Configuration

### General layout

```
<?xml version="1.0"?>
<freecast>
  <node>
    PeerProvider
    Sender
    Receiver

    <players>
      Player
      ...
    </players>

    ConnectionAssistant
  </node>

  GUI Configuration
</freecast>
```

### Peer Provider

```
<peerprovider>
  <trackeraddress>
    <host>freecast.acme-radio.org</host>
    <port>1665</port>
  </trackeraddress>
</peerprovider>
```

**trackeraddress** the TCP address where the tracker listens to node requests. This configuration must match the tracker configuration. See the section called “Target TCP address”.

### Sender

```
<sender>
  <class>udp</class>
  <listenaddress>
    <host>0.0.0.0</host>
    <port>30000~31000</port>
  </listenaddress>

  Reference
</sender>
```

**listenaddress** the UDP address where the node listens to requests of other nodes. See the section called “Listen TCP address”.

### Reference

#### Stun Reference

```
<reference>
```

```
<class>stun</class>
<host>stun.xten.net</host>
<port>3478</port>
</reference>
```

### Inet Reference

```
<reference>
  <class>inet</class>
  <host>0.0.0.0</host>
  <port>1664</port>
</reference>
```

### Multiple Reference

```
<reference>
  <class>multiple</class>
  InetReference
  InetReference
  ...
</reference>
```

## Receiver

### Peer Receiver

```
<receiver>
  <class>peer</class>
  Checksummer
</receiver>
```

### ShoutClient Receiver

```
<receiver>
  <class>shoutclient</class>
  <url>http://stream.acme-radio.org/relay.ogg</url>
  Validator
</receiver>
```

### ShoutServer Receiver

```
<receiver>
  <class>shoutserver</class>
  <listenaddress>
    <host>0.0.0.0</host>
    <port>8001</port>
  </listenaddress>
  Validator
</receiver>
```

`listenaddress` the TCP address where the shoutcast server listens to requests. The listen address will be used into the configuration of the source client. See the section called "Listen TCP address".

### Encoder-Playlist Receiver

## Receiver

---

```
<receiver>
  <class>encoder-playlist</class>
  <url>file:///etc/freecast/playlist.m3u</url>
  <channels>2</channels>
  <quality>0</quality>
  <samplerate>44100</samplerate>
  Validator
</receiver>
```

- url** the URL where the playlist can be found. This playlist will be loaded and each specified file will be played. If you use relative file names into the playlist, the base url of the playlist will be used as a prefix. Files specified via HTTP URLs will be cached locally.
- quality** the wanted quality of the encoded Ogg Vorbis stream. Read the dedicated Vorbis FAQ quality section [<http://www.vorbis.com/faq/#quality>]. Notice that only quality between 0 and 10 are supported.
- channels** the channel count of the encoded stream. Set 1 for a mono stream, 2 for a stereo stream.
- samplerate** specifies the sample rate of the encoded stream. Set 44100 for higher quality, 22050 for medium quality.

## Playlist Receiver

```
<receiver>
  <class>playlist</class>
  <url>file:///etc/freecast/playlist.m3u</url>
  <bandwidth>35</bandwidth>
  Validator
</receiver>
```

- url** the URL where the playlist can be found. This playlist will be loaded and each specified file will be played. If you use relative file names into the playlist, the base url of the playlist will be used as a prefix. Files specified via HTTP URLs will be cached locally.
- bandwidth** due to a limitation of the playlist receiver, you need to specify the bitrate of read files. You should take care to use files with a static bitrate.

## Checksummer

### Digest Checksummer

```
<checksummer>
  <class>digest</class>
</checksummer>
```

### Signature Checksummer

```
<checksummer>
  <class>signature</class>
  <privatekey>file:///etc/freecast/private.key</privatekey>
</checksummer>
```

`privatekey` the URL where the private key can be found.

## Validator

### Digest Validator

```
<validator>
  <class>digest</class>
</validator>
```

### Signature Validator

```
<validator>
  <class>signature</class>
  <publickey>http://stream.acme-radio.org/public.key</publickey>
</validator>
```

`publickey` the URL where the public key can be found.

## Connection Assistant

```
<connection-assistant>
  <host>cas.freecast.org</host>
  <port>1666</port>
</connection-assistant>
```

`host` and `port` the host and port of the connection assistant service to be used by the node.

## GUI Configuration

```
<gui>
  <background>#ffffff</background>
  <logo>http://stream.acme-radio.org/freecast/logo.png</logo>
  <tray>http://stream.acme-radio.org/freecast/icon-24.png</tray>

  <player>
    <started>http://stream.acme-radio.org/freecast/player-started.png</start
    <stopped>http://stream.acme-radio.org/freecast/player-stopped.png</stopp
  </player>
</gui>
```

`background` the background color used in the window

`logo` the logo displayed in the window

`tray` the tray icon

`player` the two icons used when the player is playing or stopped

## Player

### Audio Player

```
<player>
  <class>audio</class>
</player>
```

### Video Player

```
<player>
  <class>video</class>
  <audio>true</audio>
  <framerate>30</framerate>
</player>
```

audio            must be set to false if the Ogg Theora stream doesn't contain an audio part

framerate        specifies a video frame rate of the Ogg Theora stream

### Http Player

```
<player>
  <class>http</class>
  <listenaddress>
    <host>0.0.0.0</host>
    <port>8000</port>
  </listenaddress>
</player>
```

listenaddress    the TCP address where external players can connect to retrieve the stream. See the section called "Listen TCP address".

## Configuration samples

The configuration sample files are part of the FreeCast distributions. See Chapter 3, *Installation*.

### Audio FreeCast network

#### Example 5.1. Audio root node configuration sample

```
<!--
  FreeCast configuration file
  visit http://www.freecast.org/userguide/node-configuration.html
-->
<freecast>
  <node>
    <peerprovider>
      <!-- specifies where the tracker lives -->
      <trackeraddress>
        <host>localhost</host>
      </trackeraddress>
    </peerprovider>
    <receiver>
      <!-- just a test playlist, use your own stream ;o) -->
      <class>playlist</class>
      <url>http://download.freecast.org/jws/default/audio.m3u</url>
      <bandwidth>70</bandwidth>
    </receiver>
```

```
</node>
</freecast>
```

### Example 5.2. Audio listener node configuration sample

```
<!--
  FreeCast configuration file
  visit http://www.freecast.org/userguide/node-configuration.html
-->
<freecast>
  <node>
    <peerprovider>
      <!-- specifies where the tracker lives -->
      <trackeraddress>
        <host>localhost</host>
      </trackeraddress>
    </peerprovider>
    <players>
      <player>
        <!-- play the stream in the sound card -->
        <class>audio</class>
      </player>
    </players>
  </node>
</freecast>
```

### Example 5.3. Start a FreeCast network with the Audio configuration sample

```
[host:~] ./bin/freecast-tracker &
[host:~] ./bin/freecast -config docs/examples/audio/freecast-node-root.xml &
[host:~] ./bin/freecast-swing -config docs/examples/audio/freecast-node-list
```

## Video FreeCast network

### Example 5.4. Video root node configuration sample

```
<!--
  FreeCast configuration file
  visit http://www.freecast.org/userguide/node-configuration.html
-->
<freecast>
  <node>
    <peerprovider>
      <!-- specifies where the tracker lives -->
      <trackeraddress>
        <host>localhost</host>
      </trackeraddress>
    </peerprovider>
    <receiver>
      <!-- the nodes connects to the specified source stream server -->
      <class>shoutclient</class>
      <!-- Just a test stream, visit http://www.fluendo.com -->
      <url>http://stream.fluendo.com:8800</url>
    </receiver>
```

```
</node>
</freecast>
```

### Example 5.5. Video listener node configuration sample

```
<!--
  FreeCast configuration file
  visit http://www.freecast.org/userguide/node-configuration.html
-->
<freecast>
  <node>
    <peerprovider>
      <!-- specifies where the tracker lives -->
      <trackeraddress>
        <host>localhost</host>
      </trackeraddress>
    </peerprovider>
    <players>
      <player>
        <!-- play the stream into the embedded video plyare -->
        <class>video</class>
        <audio>true</audio>
      </player>
    </players>
  </node>
</freecast>
```

### Example 5.6. Start a FreeCast network with the Audio configuration sample

```
[host:~] ./bin/freecast-tracker &
[host:~] ./bin/freecast -config docs/examples/video/freecast-node-root.xml &
[host:~] ./bin/freecast-swing -config docs/examples/video/freecast-node-listener.x
```

## Command line usage

The CLI FreeCast node can be started in command line via the command **freecast** (see freecast(1)).

## Swing usage

The Swing FreeCast node can be started via the command **freecast-swing** (see freecast(1)).

This way to start FreeCast is reserved to advanced users. Most of listeners will use the JavaWebStart™ deployment to start FreeCast (see Chapter 7, *JavaWebStart™ deployment*).

The Swing FreeCast node is a Java™ application (and not an applet). No browser is required to start it.

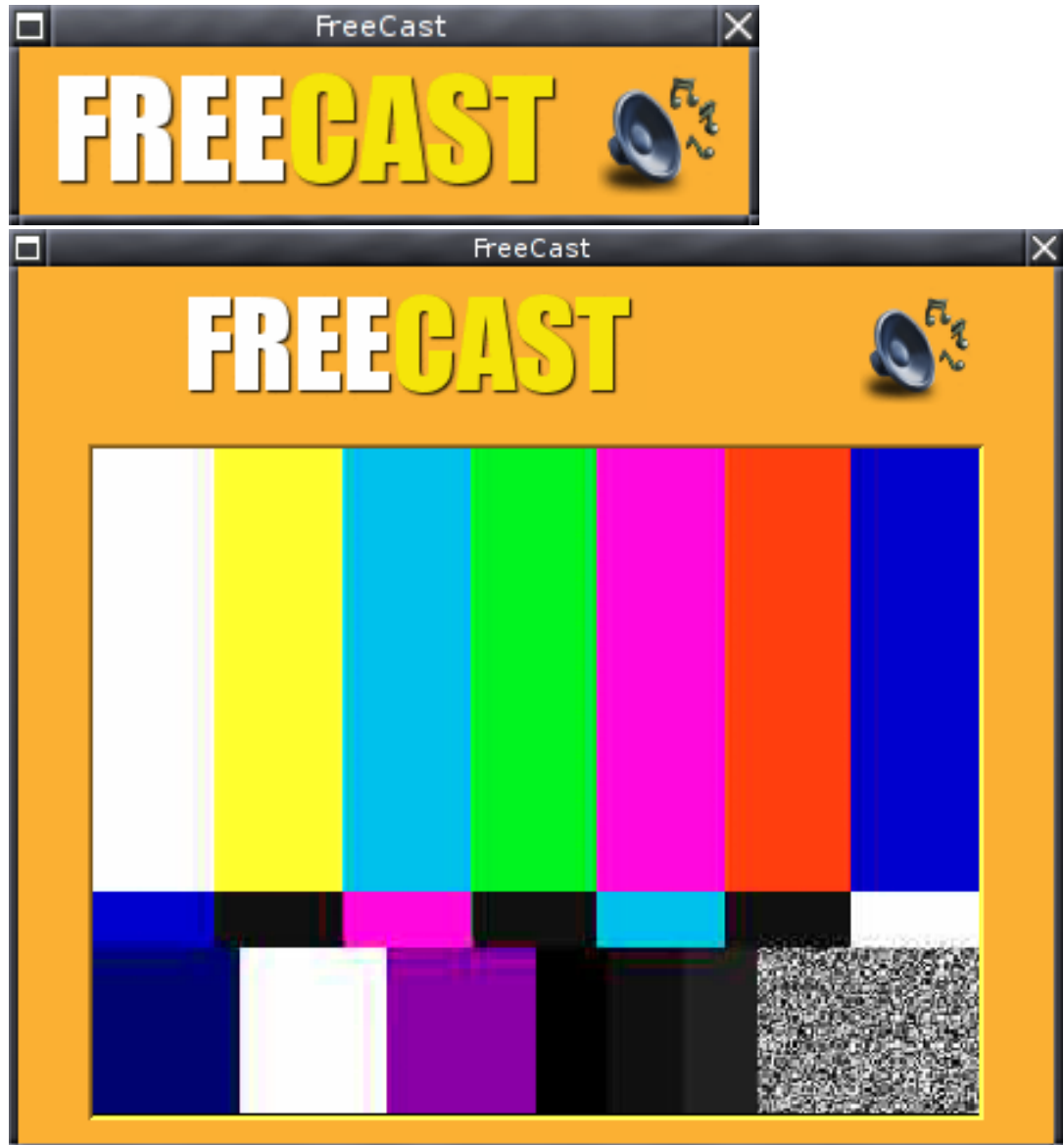
## Main dialog

The FreeCast Swing interface has been designed to be simple and pleasant. No menu, no dozen buttons like a music player. The Main dialog allows you to:

- know which stream you're listening according to the look&feel

- mute the player

That can seem few, but these are the major fonctionnalities needed for a listener.



### Mute and relay

When you press the player button, the audio or video stream is no longer played by your node. You can restart the player by pressing again the player button. The player will not restart at the same position. When it is muted, the player will release your sound card (to use an external player for instance). The FreeCast node stays connected to the FreeCast network and is able to relay other nodes. You can minimize the window and let the FreeCast node alive. It's a way to support the FreeCast network you are listening to because other listeners can be relayed.

### Options bar

Extended features are available if you click on the logo. When this option bar is opened, you can :

- see the last log message,
- start the About dialog,

- start the Log dialog.



The About dialog gives you credits information about FreeCast. The Log dialog allows you to see the messages logged by FreeCast and to save them to a file (to illustrate a bug report for instance).

---

# Chapter 6. Configuration

There are some general considerations about the configuration of FreeCast applications. Each application is configurable via XML files. These XML files follow the same logic.

The configuration file only needs to complete and overwrite the default configuration of the application.

## Common configuration elements

### Network elements

#### Host

Host elements used into the FreeCast configuration designate an IP address over a IP network (like your local network or Internet). Hosts can be specified via either host names or IP addresses. As usually, host names will be resolved by FreeCast applications.

#### Example 6.1. Host configurations

```
<host>82.68.47.19</host>
```

```
<host>stream.acme-radio.org</host>
```

This information can be used by FreeCast applications in another context than your local network. Don't use local host names or local IP addresses in a configuration file which must be used over Internet.

#### Port

Port elements designate a TCP port or a port selection:

- A single port is designated by a number.
- A port range is defined by a lower and an higher port separated by a -.
- A random port range is defined by a lower and an higher port separated by a ~.
- A port selection can contain one or more ports or port ranges separated by a ,.

#### Example 6.2. Port configurations

```
<port>1665</port>
```

```
<port>1664,1666-1680</port> <!-- port 1664 or from 1666 to 1670 -->
```

```
<port>30000~30100</port> <!-- port from 30000 to 30100 choosed randomly -->
```

## Listen TCP address

```
<listenaddress>  
  <host>0.0.0.0</host>  
  <port>1665</port>  
</listenaddress>
```

**host** the listen IP address. The default is 0.0.0.0, connections will be listened over all network interfaces. By choosing a specific address, you can limit the reachability. For instance, You can use localhost to limit the reachability to your computer. Don't modify this default parameter unless you have a specific need.

**port** the TCP port used to listen and receive requests from other FreeCast applications. By using a port selection, you can avoid problems with ports already used by other applications. In this case, the first free port will be used.

The choice of a listen address must be compatible with your network configuration. Listening connections on a computer behind a router or a firewall can require a manual configuration step about the related software or hardware.

## Target TCP address

```
<listenaddress>  
  <host>freecast.acme-radio.org</host>  
  <port>1665</port>  
</listenaddress>
```

**host** by default, the target host is localhost.

**port** you must specify the port where the requests are listened to by the remote host. You can't use a port selection.

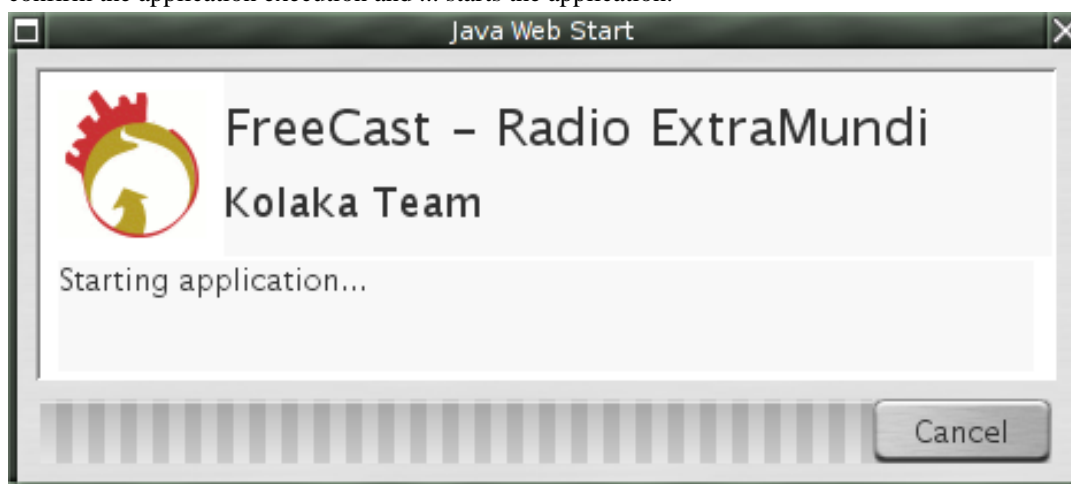
---

# Chapter 7. JavaWebStart™ deployment

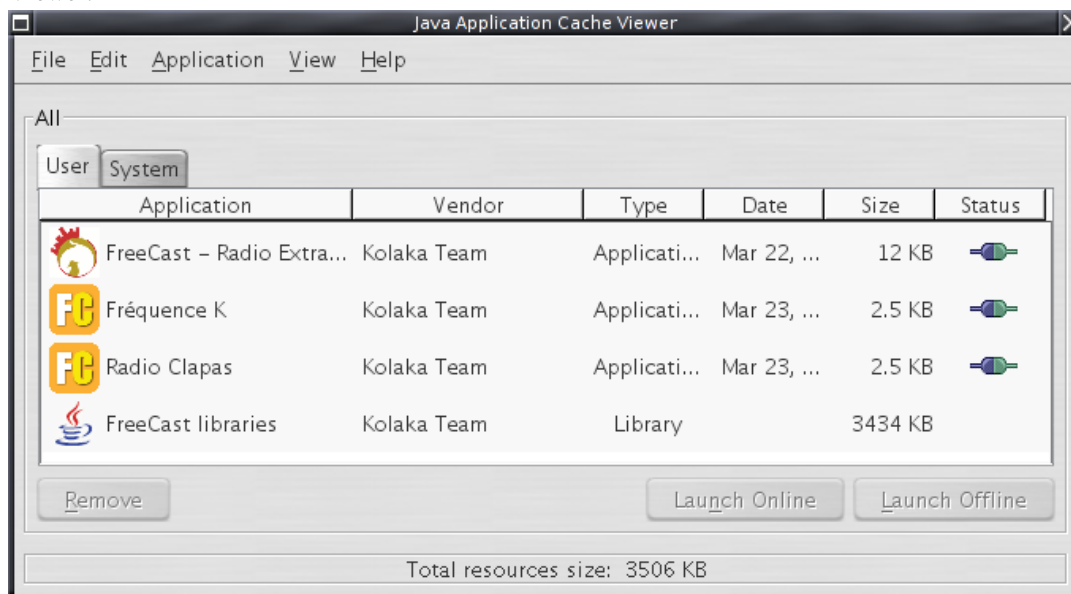
## Overview

### JavaWebStart™ in few words

JavaWebStart™ is a launcher which allows users to start Java™ applications by a single click on a website. To sum-up quickly, the user's browser downloads a `jnlp` file on a website. This file contains a technical description of the application to be started. JavaWebStart™ (which should be associated to this kind of file) starts by reading this file. It starts a GUI to make the user follow the starting process. Then, it downloads the necessary files, validates the numeric signature, asks the user to confirm the application execution and ... starts the application.



JavaWebStart™ allows the user to restart easily a previously started application. If the user restarts an application, previously downloaded files are used. Needed updates are performed automatically. The user can consult the list of known applications with the JavaWebStart™ Cache Application Viewer.



JavaWebStart™ is a product of Sun Microsystems. It's included into the Java Runtime Environment. Visit the product homepage <http://java.sun.com/products/javawebstart/>

## Benefits for FreeCast deployment

The JavaWebStart™ deployment of FreeCast allows to provide a *turnkey solution* for the end-users (your listeners):

- Listeners can install FreeCast as simply as possible.
- They always use the right and the last FreeCast release without effort.
- Even if they use FreeCast for several networks, they only download once the FreeCast application.

You just need to make available a small xml file (see in the section called “Stream description file”) on your website, include a link to the FreeCast start page (see in the section called “Start page”). You won't consume bandwidth because the needed files are downloaded from the <http://download.freecast.org>.

## Stream description file

Let's assume some information about your stream :

- your webradio is named Acme Radio
- your website is <http://acme-radio.org/>
- your tracker is available at [freecast.acme-radio.org:1665](http://freecast.acme-radio.org:1665)
- you're running the `freecast-20060629` release

In this case, your stream description file should look like this:

### Example 7.1. Minimalist stream description

```
<stream>
<name>Acme Radio</name>
<homepage>http://acme-radio.org/</homepage>
<tracker>
  <host>acme-radio.org</host>
  <port>1665</port>
</tracker>
<jnlp>
  <version>20060629</version>
</jnlp>
</stream>
```

This stream description must be published on your website. In our example, it's published at <http://acme-radio.org/freecast/descriptor.xml>.

## Stream description option list

These are all the options you can use into the stream description file.

### Table 7.1. Stream description

parameter name	comments
name	a short description of the stream
homepage	the URL of your website or the page dedicated to the FreeCast stream for instance
tracker.host	the host where your tracker is running
tracker.port	the port where your tracker is running (optional if the default port 1665 is used)
config	the URL of the node configuration file that the listener nodes must use
jnlp.version	the FreeCast release you're using for tracker and your root node. Important in order to make your listeners run a compatible release.



### Version

The `jnlp.version` option is important to make your listeners run a compatible FreeCast release. Take care to update it when you upgrade the tracker and root node release.

## Advanced examples

There are some examples of streaming configuration files.

### Specify a node configuration file

You can specify a node configuration file to override the default one. See the section called “Configuration”.

#### Example 7.2. Streaming description with a node configuration file

```
<stream>
  <name>acme Radio</name>
  <homepage>http://acme-radio.org/</homepage>
  <jnlp>
    <version>20060629</version>
    <config>http://acme-radio.org/freecast/config.xml</config>
  </jnlp>
</stream>
```

There is a possible node configuration file. It will be used by the node listeners as specified in the previous stream configuration file. It activates the stream signature and the http player.

#### Example 7.3. A possible node configuration file

```
<freecast>
  <node>
    <peerprovider>
      <trackeraddress>
        <host>acme-radio.org</host>
      </trackeraddress>
    </peerprovider>
    <receiver>
      <class>peer</class>
    </receiver>
  </node>
</freecast>
```

```
<class>signature</class>
<publickey>
  http://acme-radio.org/freecast/public.key
</publickey>
</validator>
</receiver>
<players>
  <player>
    <class>http</class>
    <listenaddress>
      <port>8001</port>
    </listenaddress>
  </player>
</players>
</node>
</freecast>
```

- ❶ set the tracker host (needed if the `tracker.host` about it isn't used in the stream configuration file)
- ❷ activate the stream validator by signature and specifies the public key to be used
- ❸ activate the HTTP player and specifies the port to be used
- ❹
- ❺
- ❻

This is only a configuration sample, see the section called “Configuration” for more information.

## Start page

Now that you have a stream descriptor file, you can use it into your web site to give listeners access to your stream. To facilitate the FreeCast node startup from websites, the FreeCast provides a customizable *start page*. You don't need to try to explain the install process or add links to the installation pages on the FreeCast website. Just use the *start page* which will be automatically customized with the information provided by your stream descriptor file.

The FreeCast website provides a small javascript function which allows you to create links to the start page by specifying the *hostname and path of your stream descriptor file*. For instance, if your descriptor is available at `http://acme-radio.org/freecast/descriptor.xml`, you must only specify `acme-radio.org/freecast/descriptor.xml`.

### Example 7.4. Use the start page in a web page

```
<html>
  <head>
    ...
    <script type="text/javascript" src="http://www.freecast.org/start.js"></script>
  </head>
  <body>
    ...
    <a href="javascript:start('acme-radio.org/freecast/descriptor.xml')">FreeCast
    ...
  </body>
</html>
```

See the FreeCast listen page [<http://www.freecast.org/listen>] source for a real example.

---

# Chapter 8. Source samples

## Source clients

A source client can be used to create the Ogg stream broadcasted on the FreeCast network. You need to configure your root node with a shoutserver receiver (see the section called “Shoutserver receiver”) and configure the source client to send the stream to the port where the node waits for the connection.

### Ices

ices2 is a very robust source client used commonly with the famous icecast.

Visit the ices2 website to know more about this tool: <http://icecast.org/ices.php>.

#### Example 8.1. Ices2 sample configuration

```
<ices>
  <loglevel>4</loglevel>
  <consolelog>1</consolelog>

  <stream>
    <metadata>
      <name>FreeCast Stream</name>
      <genre></genre>
      <description></description>
    </metadata>

    <input>
      <module>alsa</module>
      <param name="rate">44100</param>
      <param name="channels">2</param>
      <param name="device">default</param>
    </input>

    <instance>
      <hostname>127.0.0.1</hostname>
      <port>8000</port>
      <password>changeme</password>
      <mount>/stream.ogg</mount>
      <reconnectdelay>2</reconnectdelay>
      <reconnectattempts>5</reconnectattempts>

      <downmix>1</downmix>

      <encode>
        <quality>0</quality>
        <samplerate>44100</samplerate>
        <channels>1</channels>
      </encode>
    </instance>
  </stream>
</ices>
```

### Jack support

A very powerful feature of ices2 is the jack support which allows to create a sound stream retrieved from a jack server. By this way, you can integrate a lot of sound tools from a simple player like Xmms [<http://www.xmms.org/>] to a complete radio broadcast automation solution like Rivendell

[<http://www.salemradiolabs.com/rivendell/>].

This support is for the moment present into the ices-kh branch. You can know more about this ices branch on <http://ices.reboot.fm/>. Visit the jack website [<http://jackit.sf.net>] to know more about this technology.

## OddCast

OddSock.org [<http://www.oddsock.org>] provides a lot of streaming tools for the Windows platforms. oddcast is a standalone application or a WinAmp™ plugin which creates a Ogg stream from the line input or from what WinAmp™ is playing.

Visit the oddcast website: <http://www.oddsock.org/tools/oddcastv3/>.

Watch FreeCast screencasts [<http://www.freecast.org/screencasts>] dedicated to the OddCast and FreeCast collaboration.

## StreamTranscoder

streamtranscoder is a very useful tool to create a Ogg stream from an existing stream.

Visit the streamtranscoder website: <http://www.oddsock.org/tools/streamTranscoder/>

## Source servers

The Ogg stream broadcasted on the FreeCast network can be retrieved on a source server. You need to configure your root node with a `shoutclient receiver` (see the section called “Shoutclient receiver”) by specifying the stream URL and other needed parameters.

## IceCast

icecast is one of the most famous streaming servers. Visit the icecast website: <http://www.icecast.org>.

## Flumotion

flumotion is a streaming media server which allows to create Ogg Theora video streams. Visit the flumotion website: <http://www.flumotion.net>.

---

# Reference Pages

---

---

---

## Name

freecast -- starts a FreeCast node with a command line interface (CLI)

```
freecast [-config file/url] [-Dproperty=value] [-dryrun]freecast-tracker  
[-config file/url] [-Dproperty=value] [-dryrun]freecast-swing [-config file/url]  
[-Dproperty=value] [-dryrun]
```

## Description

Starts the related FreeCast application (tracker, node cli or node swing) according to the used command. The application configuration can be controlled via related options.

## Options

<code>-config <i>file</i> or <i>url</i></code>	Loads the specified configuration. Can be a file or an URL.
<code>-D<i>property=value</i></code>	Overrides the specified configuration property by the given value.
<code>-dryrun</code>	Configures the FreeCast application and stops it. Useful to test the configuration.

## Notes

Used URLs can use any protocol supported by the JVM (`file:`, `http:`, `ftp:`, etc ...).

---

---

---

# Appendix A. GNU General Public License

Version 2, June 1991  
Copyright © 1989, 1991 Free Software Foundation, Inc.

Free Software Foundation, Inc.  
59 Temple Place, Suite 330,  
Boston, MA 02111-1307  
USA

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Version 2, June 1991

## Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software - to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps:

1. copyright the software, and
2. offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for

everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

# **TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION**

## **Section 0**

This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

## **Section 1**

You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

## **Section 2**

You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

1. You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
2. You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.
3. If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License.



### **Exception:**

If the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

## Section 3

---

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

## Section 3

You may copy and distribute the Program (or a work based on it, under Section 2 in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

1. Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
2. Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
3. Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

## Section 4

You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

## Section 5

You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

## Section 6

Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

## Section 7

If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

## Section 8

If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

## Section 9

The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

## Section 10

If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

## NO WARRANTY Section 11

BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

## Section 12

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

## How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.> Copyright (C) <year> <name of author>

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 2 of the License, 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., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

## How to Apply These Terms to Your New Programs

---

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) year name of author Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'. This is free software, and you are welcome to redistribute it under certain conditions; type `show c' for details.

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program `Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989 Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.

---

# Index

## **N**

node, 2, 17, 17, 20  
root, 2

## **T**

tracker, 2, 13, 13, 13

---