

---

# User manual

(SRT Pusher)

*Happytimesoft Technology Co., LTD*

---

## Declaration

All rights reserved. No part of this publication may be excerpted, reproduced, translated, annotated or edited, in any form or by any means, without the prior written permission of the copyright owner.

Since the product version upgrade or other reasons, this manual will subsequently be updated. Unless otherwise agreed, this manual only as a guide, this manual all statements, information, recommendations do not constitute any express or implied warranties.

---

[www.happytimesoft.com](http://www.happytimesoft.com)

---

## Table of Contents

Chapter 1 Introduction.....	4
Chapter 2 Key features.....	5
Chapter 3 <b>Configuration</b> .....	<b>6</b>
3.1 Configuration Templates .....	6
3.2 Configuring Node Description .....	6
3.2.1 <i>System parameters</i> .....	6
3.2.2 <i>Pusher node</i> .....	7
Chapter 4 <b>Run SRT Pusher</b> .....	<b>10</b>

---

## Chapter 1 Introduction

Happytime SRT pusher is a high-efficiency SRT pusher app, it support push the local media files, audio/video devices, living screen, application windows and the rtsp/rtmp/srt/http mjpeg stream, support multiple pushers at the same time, stable and reliable.

It can be perfectly applied to live broadcast requirements in various industries, desktop live broadcast, live camera, live broadcast, etc.

It supports various platforms such as Windows, Linux, MAC, ARM, Android, and iOS, supports cross compilation.

It provides the live audio/video stub class, just need to implement a few interfaces to push the live audio/video SRT stream.

---

## Chapter 2 Key features

Support variety of audio and video files

Support push video from camera and living screen

Support push video from application windows

Support push audio from audio device

Support recording system audio on Windows

Support video codec H264,H265

Support audio codec AAC

Support automatic transcoding function

Support RTSP/RTMP stream to SRT stream

Support push media data from SRT stream

Support for configuring audio and video output parameters

Small size, suitable for embedded development

Code structure clear, easy to use

---

## Chapter 3 Configuration

### 3.1 Configuration Templates

```
<?xml version="1.0" encoding="utf-8"?>
<config>
  <log_enable>1</log_enable>
  <log_level>1</log_level>
  <loop_nums>-1</loop_nums>
  <reconn_interval>15</reconn_interval>

  <pusher>
    <srcurl>test.mp4</srcurl>
    <dsturl>srt://192.168.3.36:8080?streamid=myapp/live,m=publish</dsturl>
    <video>
      <codec>H264</codec>
      <width></width>
      <height></height>
      <framerate></framerate>
      <bitrate></bitrate>
    </video>
    <audio>
      <codec>AAC</codec>
      <samplerate>44100</samplerate>
      <channels>2</channels>
      <bitrate></bitrate>
    </audio>
  </pusher>
</config>
```

### 3.2 Configuring Node Description

#### 3.2.1 System parameters

**<log\_enable>**

Whether enable the log function,0-disable,1-enable

**<log\_level>**

The log level:

---

TRACE	0
DEBUG	1
INFO	2
WARN	3
ERROR	4
FATAL	5

#### **<loop\_nums>**

When streaming local media files, specify the number of loop playback, -1 means infinite loop.

#### **<reconn\_interval>**

When the connection with the server is disconnected, the interval between reconnecting to the server, unit is second.

### 3.2.2 *Pusher* node

**<Pusher>** : Represents a push stream, specify the audio and video output parameters, it can configure multiple nodes

#### **<srcurl>**

The push stream source, it supports the following parameters:

*filename*: local media file name, the media files need to be placed in the directory where srtpusher is located

*rtsp, rtmp, srt or http mjpeg url*: the original rtsp/rtmp/srt/http mjpeg stream address, such as

rtsp://user:pass@ip:port/url or rtmp://user:pass@ip:port/app/stream.

*screenlive* : stream the living screen

*videodevice* : stream from the camera

*audiodevice*: stream from the audio device

*screenlive+audiodevice*: stream the living screen and stream audio from the audio device

*videodevice+audiodevice*: stream video from the video device and stream audio from the audio device

*window=[window title]*: stream video from application windows

If your system have multiple audio capture device, you can use

*audiodeviceN*, the *N* to specify the audio capture device index, start from 0, such as:

*audiodevice* ; stream audio from the first audio device

*audiodevice1* ; stream audio from the second audio device

---

If your system have multiple video capture device, you can use  
videodevice $N$ , the  $N$  to specify the video capture device index, start from 0, such as:  
videodevice ; stream video from the first video device  
videodevice1 ; stream video from the second video device

If your system have multiple monitors, you can use  
screenlive $N$ , the  $N$  to specify the monitor index, start from 0, such as:  
screenlive ; stream living screen from the first monitor  
screenlive1 ; stream living screen from the second monitor

The audio index or video index represents which device can execute ***srtpusher -l device*** to view.

videodevice or audiodevice can also specify the device name, such as  
videodevice=testvideo

Execute the ***srtpusher -l device*** command to get the device name.

**Note that** there can be no spaces in the device name, if the device name contains spaces, you need to use %20 instead of spaces.

If the device name is “FaceTime HD Camera (Built-in)”:  
videodevice=FaceTime%20HD%20Camera%20(Built-in)

Captures the window with the specified window title, execute the ***srtpusher -l window*** command to view valid window titles

**Note that** there can be no spaces in the window title, if the window tilte contains spaces, you need to use %20 instead of spaces.

If the window title is “VLC media player”:  
window=VLC%20media%20player

**<dsturl>**: Specify the push destination address, it should be a SRT stream address.

**<user>**: if the <srcurl> is a RTSP stream address, specify the rtsp server’s login user name

**<pass>**: if the <srcurl> is a RTSP stream address, specify the rtsp server’s login password

**<video>** : Specify the video output parameters



---

**<codec>**

Specify the video stream codec, it can specify the following value:

H264 : output H264 video stream

H265 : output H265 video stream

**<width>**

Specify the output video width, if 0 it use the original video width (living screen use the screen width, camera use the default width)

**<height>**

Specify the output video height, if 0 it use the original video height (living screen use the screen height, camera use the default height)

**<framerate>**

Specify the output video framerate, if 0 it use the original video framerate (living screen and camera use the default value 25)

**<bitrate>**

Specify the output video bit rate, if it is 0, it is automatically calculated

**<audio>** : Specify the audio output parameters

**<codec>**

Specify the audio stream codec, it can specify the following value:

AAC: output AAC audio stream

**<samplerate>**

Specify the audio sample rate, it can specify the following values:

8000, 11025, 12000, 16000, 22050, 24000, 32000, 44100, 48000

If 0 it use the original audio sample rate (audio device use the default value 8000)

**<channels>**

Specify the audio channel number, 1 is mono, 2 is stereo

If 0 it use the original audio channel number (audio device use the default value 2)

**<bitrate>**

Specify the output audio bit rate, if it is 0, it is automatically calculated

**Note :** If the data source specified by <srcurl> has no audio, but the <audio> tag is configured, output silent audio data.

If the data source specified by <srcurl> has audio, but no <audio> tag is configured, no audio data will be output.

---

## Chapter 4 Run SRT Pusher

The srt pusher is a console application.

Windows: to run the srt pusher, simply type "srtpusher".

Linux: to run the srt pusher, type "./start.sh", on linux platform, srt pusher run as daemon by default.

srt pusher supports the following command line options:

`-c config` specify the configuration file

`-c` option specifies the configuration file, if not specified, the default configuration srtpusher.cfg is used.

`-l [device|videodevice|audiodevice|window]`

`-l device` list available video and audio capture device

`-l videodevice` list available video capture device

`-l audiodevice` list available audio capture device

`-l window` list available application window

Below is sample output of `-l device`:

*srtpusher -l device*

*Available video capture device :*

*index : 0, name : FaceTime HD Camera (Built-in)*

*Available audio capture device :*

*index : 0, name : Headset Microphone (Apple Audio Device)*

*index : 1, name : Internal Digital Microphone (Apple Audio Device)*

**Note :** The demo version has the following limitations

Maximum support two simultaneous pusher stream.