Jirc documentation

Table of contents

Note:
If looking for a user friendly way of transforming a Lirc file to something else, *do not read this article*. Use the IrScrutinizer instead, in particular the Lirc import feature.

Date	Description
2014-01-12	Initial version.
2014-05-06	Updated for version 0.3.0, improving the handling of repeats, and fixing some bugs.

Table 1: Revision history

1 Introduction

There is a large body of captured IR signals for different remotes/devices in the Internet, in particular at lirc.sourceforge.net/remotes/. Sometime, it is desirable to transfer this information to more universally known formats for IR signals, such as the CCF format. Although attempts have been made to document the format of the Lirc format, writing a complete and with Lirc coinciding parser/renderer for these files is probably a very difficult undertaking. From this insight, "only Lirc understands Lirc files", the program lirc2xml was created, as a special driver for the `lircd` daemon, writing an XML file instead of accessing hardware, available as a patch to the C sources of Lirc.

For portability reason, a pure Java solution was desired. So I decided to translate the appropriate Lirc files to Java, and the present project was born. The base of the translation was Lirc version 0.9.0.

The initial version, 0.2.0, did not implement the handling of repeats. The current version 0.3.0 however implements repetition and the handling of multiple command numbers, and fixes several small problems, e.g. the handling of multiple remotes with the same name.

1.1 Copyright

The original work is copyrighted under the GNU General Public License, version 2. Being a derived work, this goes for the present work too.

The program uses JCommander by Cédric Beust to parse the command line arguments. It is free software with Apache 2 license.

1.2 Program overview

The main goal was, with a moderate effort, to create a (pure) Java version of Lirc2XML, not to make a faithful translation, nor to create maintainable Java code of the highest quality.

The program basically consists of a parser of the configuration files (class `ConfigFile`), some helper classes, and the class `Transmit`. The parser has been written from scratch using ANTLR version 3.4. The classes `Hardware`, `IrNCode`, `IrCodeNode`, and `IrRemote` are essentially translations of Lirc's C structs with similar names. The class `Transmit`, essentially a translation of the C file `translate.c`, carries out the rendering of the IR signals. The main routine resides in the class `Lirc2Xml`, which is written from scratch. The output file is generated in the general Girr format, using its supporting Java library.

All references to serial commands have been deleted; it does not belong to the Lirc domain anyhow (IMHO).

2 Installation

The program is distributed in source format only. For full functionality, also the DecodeIR library has to be installed. For this, giving the directory path to the Java VM using the `-Djava.library.path` option may be necessary. The supplied wrapper `lirc2xml.sh` can be used as an example for this.

3 Usage

3.1 Command line arguments

```
Usage: Lirc2Xml [options] [configfile]
  Options:
    -c, --ccf
       Generate the CCF ("Hex", "Pronto") form of the signals
       Default: false
    -d, --debug
       Debug
       Default: 0
    -e, --encoding
       Character encoding of the generated XML file
       Default: UTF-8
    -f, --fatraw
       Use the fat format for raw signals
       Default: false
    -h, --help, -?
       Display help message
       Default: false
    -o, --outfile
       Output filename
    -p, --parameters
       Generate the protocol name and parameters (if possible) for the signals
       Default: false
    -R, --raw
       Generate the raw form of the signals
       Default: false
    -r, --remote
       Name of the remote to include in the export (will export all if left
       empty)
    -s, --schemalocation
       Create schema location attribute
       Default: false
    -v, --version
       Display version information
       Default: false
```

```
     -x, --xslt
        Link to XSLT stylesheet
```

The program has a traditional "Unix" usage. If no file argument is given, standard input is read. As argument, also a directory can be given, in which case an entire file hierarchy is traversed for Lirc files, handling unparsable files gracefully. Also URLs, e.g. http://lirc.sourceforge.net/remotes/yamaha/RX-V995 can be given as arguments.

On my computer, a local copy of the complete Lirc remote collection (slightly over 3000 files) is transformed in 51 seconds, allocating around 900MB of memory.

3.2 API documentation

Javadoc available here.

3.3 Restrictions

The parser presently requires the command numbers following the command names to be hexadecimal only. Command names and remote names containing spaces are rejected. Commands named "end" are presently not accepted. Names are assumed to consist of 8-bit characters between "!" and "~", or between hexadecimal 0xA1 to 0xFF (e.g. using the ASCII or the ISO-8859-1 character sets). The input files are being read asuming the encoding ISO-8859-1.

4 Download

- Jirc-src-0.3.0.zip. Also needed is
- Girr-src-1.0.0.zip, and
- IrpMaster-src-1.0.1.zip.