FIAT 1.2 data file format

Greg Kochanski

October 27, 2010

1 History and General Description

FIAT 1.2 is an extension of the FIAT file format originally defined by David Wittman (UC Davis). FIAT 1.0 is defined at `http://dls.physics.ucdavis.edu/fiat/fiat.html`. FIAT format is a multicolumn text file, with header information and comment lines. It has named columns (like CSV formats). FIAT is released under the Lesser Gnu Public License, Version 2 or later.[1]

2 Features

- Header information looks like a comment to most programs, so they will treat a FIAT file as simple multi-column Unicode. (Header lines begin with '#'.) For instance, you can read FIAT files in most spreadsheet programs.
- FIAT defines column names in the header, so it's easy to make your programs upwards compatible: you can add new columns to a FIAT file, and any existing scripts that read FIAT files will continue to run.
- Simple to parse.
- Easy to generate.
- Readable in a text editor, and understandable by humans.
- Can represent any value without restriction. Specifically, it can carry Unicode character strings without any need for quoting.[2]

3 General Description

This describes fiat 1.2 format, which is nearly 100% upwards compatible with fiat 1.0 format. It is defined as follows:

[1] This document is released under the Creative Commons attribution-sharealike 2.0 UK and Wales license, at `http://creativecommons.org/licenses/by-sa/2.0/uk/`.

[2] As long as those strings don't contain a few special characters. See §6.

1. Lines are separated by newlines. Every line (including the last) *must* end in a newline.

2. There are three kinds of line: data lines, header lines, and comment lines. They can happen in any order, though typically all the header and comment lines are at the top of the file, and then the data lines follow.

   (a) Data lines are a list of items that are (normally) separated by any kind of whitespace. (NB: you can specify a different separator character if desired.)

   (b) Header lines set a particular *attribute* to a certain *value*.

   (c) Comments lines begin with a hash mark and can contain anything except a newline. They are there primarily for documentation.

   (d) Note that a blank line is treated as a data line.

4 A Simple Example

```
# fiat 1.2
# TTYPE1 = b
# TTYPE2 = a
# SAMPRATE = 2.3
# This is a comment.
2 1
3 2
0 1
```

Here, the attribute SAMPRATE is set to "2.3". We have two columns of data, called "a" and "b". So, in the first line, $b = 2$ and $a = 1$, et cetera. Note that in this file, "b" is the first column.

5 Availability

A Python module that can read and write FIAT format files is available from the gmisclib package of the speechresearch project of `http://sourceforge.org`. If you use the *fiatio* module, you can ignore most of the details below. Documentation on that module is available under `http://kochanski.org/gpk/code/speechresearch/gmisclib`. In short, it takes a python dictionary and writes it to the FIAT file header; it takes a list of dictionaries and converts them, one at a time, into data lines. A companion function is available to read the FIAT file back into a python dictionary (from the header information) and a list of dictionaries (one dictionary per data line in the FIAT file).

6 Detailed Description

1. FIAT reads and writes a file as Unicode.

2. Any character in an attribute, a value or a data line may be encoded by replacing it with a percent character (%) followed by the corresponding two-digit hex code.[3]

   (a) Newlines, semicolons (;) and percent signs (%) *must* be replaced. Initial and final spaces in values (at least the ones you wish to be preserved) *must* be replaced.

   (b) `#`, `|` and `=` should be replaced. `#` *must* be replaced if it is the first character in the first column, to avoid confusion with a comments line. `|` *must* be replaced inside attributes or values quoted with the vertical bar `|` character. To avoid an incorrect attribute/value split, attributes containing `=` *must* be quoted with a vertical bar and be encoded.

   (c) There are alternative, more human-friendly codes that may be used instead of pure hex. Notably, `%S` is space, `%L` is newline, `%R` is carriage return, `%t` is tab, and `%T` is percent.

3. At the top of the file, you have a line identifying the format: `# fiat 1.2` (regexp: `# fiat 1\.[0-9.]+`). If this is missing, the file format is assumed to be the newest version of FIAT.[4][5]

4. Header Lines always begin with a hash mark ("#").

   (a) Header lines are in the form `# attribute = value` where white space is optional and can be a mixture of spaces and tabs.

   (b) The attribute should match the regular expression `[a-zA-Z_][a-zA-Z_0-9]*`.

   (c) The value is whatever follows the equals sign, after leading and following white space is stripped. If the value begins and ends with the same quote character, either `'` or `"`, the quotes are also stripped off. Values may normally contain any character except newline and the chosen quote. Note that you must quote a value if it begins or ends with whitespace.

   (d) One may encode a value or an attribute by using vertical bars `|` as the quoting character. In that case, the %-sign encoding rules apply. However, attributes and values are not encoded as a default.

[3] This is an obvious restriction, in that two hex digits cannot represent most unicode characters. However, this is mostly done to replace characters used in the definition of the format (e.g. newline and tab), and those characters have small unicode numbers. So, it works in practice.

[4] FIAT readers are not required to be backwards compatible, so they can just ignore this line.

[5] NB: This line would normally be absorbed by the FIAT I/O library, and not passed out to user code as a comment.

(e) If there is any chance your attribute names contain an equals sign or spaces, you should use the vertical bar encoding scheme.

5. Special Header Attributes

   - Attributes in the form `TTYPEn` (where n is a positive integer) name the columns of the data (the leftmost column named by `TTYPE1`).
   - If you don't name the I^{th} column, its name will default to I.
   - The `COL_SEPARATOR` attribute contains the numeric code(s) (Unicode) of the column separator character(s). This defaults to "9", horizontal tab. If you want a multi-character separator string, use several numeric codes separated by whitespace, like this: `COL_SEPARATOR="44 32"` to get ", " between columns.
   - The `COL_EMPTY` attribute with the string used to mark missing data item. (This defaults to `%na`.) Note that nonexistant is not the same as a zero-length string.
   - Special header attributes may appear anywhere in the file. They take effect immediately.
   - The file should contain a `DATE` attribute in the form `ccyy-mm-ddThh:mm:ss` (as defined in the NASA FITS format).
   - No Special Header Attributes are required.

6. Data Lines.

   - Columns are separated (by default) with any white space, but if there is a `COL_SEPARATOR` attribute, it is used instead.
   - Missing entries for columns should be indicated by whatever code is specified in `COL_EMPTY`, if that is set. It defaults to `%na`.
   - If `COL_SEPARATOR` is set, `COL_SEPARATOR` characters separate items, some of which may simply be empty. (E.g. if comma is a separator, the line `%na,,x` gives a zero length string for the second column, but the first column has missing data. (In all cases, a completely blank line is treated as a datum where all columns are missing.

7 Interpretations

7.1 General Data Model

Fiat files can be interpreted as a list of attribute-value mappings, one mapping per line. The mapping contains all the header attribute-value pairs seen so far, and all the columns of data that are defined (but not missing) for that line. So, if a datum is missing, there will be no mapping from the column name to a value for that line. If you try to access a missing datum, this will normally cause an error or raise an exception, unless you first check to see if the mapping exists.

7.2 Header / Columns Data Model

Of course, you can choose to interpret all of the header lines as header information and not worry about where in the file it appears. In this case, a FIAT file is a dictionary (the header lines) plus multicolumn data, where some of the data may be missing.

8 Examples

8.1 Minimal Example

```
2 1
```

This has no attributes and one data line. In that line, the column named "0" is "2" and the column named "1" is "1". Note that an empty file is not legal FIAT because it doesn't end with a newline. A file consisting of a single newline is legal, though: it contains no data lines.

8.2 A More Complex Example

```
# fiat 1.2
# TTYPE1 = b
# TTYPE2 = a
# SAMPRATE = 2.3
# DATE = 2001-09-21T21:32:32
# COL_EMPTY = "%na"
# COL_SEPARATOR = "9"
# Comment1
# Comment2
# b a
2 1
3 2
3 %na
%na 3
%na %na
0 1
```

This uses tab as a column separator, so that values in data lines could contain spaces without the necessity of encoding them. It uses "%na" to indicate missing data, so there is no value for "a" on data line 3. On data line five, there is no data at all.

8.3 Header Lines in the Middle of the File

```
# fiat 1.2
# TTYPE1 = b
# sampling_rate = 2.3
```

```
2
# TTYPE2 = a
3 2
3 5
# sampling_rate = 2.1
0 1
0 2
```

This data series starts as a single column, named "b", at a sampling_rate of "2.3". Then, a second column (named "a") is added, and soon thereafter the sampling_rate is changed to "2.5". Note that you cannot get rid of a column, but if the measurements of "b" stopped, you could always fill that column with `%na`.

9 Differences from FIAT 1.0

- David Wittman FIAT (1.0), requires that a value to either be quoted or to contain no white space. Dwittman FIAT will take a line in the form `#a=b c`, and interpret c as a comment, setting the value of `a` to "b". However, FIAT 1.2 will interpret the value to be "b c".
- FIAT 1.0 did not have any encoding mechanisms. It could not represent data or values containing "difficult" characters.

These notwithstanding, most files will be interpreted the same way as Fiat 1.0.