The PBI file format

Version 1.2   2006-08-31

Introduction

The PBI (Problem Import) file format is intended to be a very simple line-based text file format for representing orthodox chess compositions and such information about them as can be found in a book collection or a newspaper chess column. The basic idea is that a PBI file should be an electronic equivalent of a note book or simple card register of chess problems, yet still well-defined enough to be used for information exchange.

The PBI file format is *not* an attempt to create a general exchange format for all kinds of problem-related information. There is at least one such finished format [1], and at least two another underways at the time of writing, and I know sufficiently much about the difficulties involved to be happy to leave them to somebody else.

Nor are PBI records intended to be used for some kind of general problem repository, in the way the PGN format is sometimes used, by cramming all kinds of unrelated problems/games into one file. Although it probably could be used in such way, it has been created for a much simpler task: that of collecting problems and problem-related information from a specific source. Manipulating, editing, classifying, comparing, sorting and managing those collected records in general is a task for another kind of computer program.

The description below is primarily intended for people with programming skills, some knowledge of the EBNF notation used in the syntax descriptions, or with the equivalent need for knowing all the little details.

At present there exists a PBI-editor for Windows XP. It is still under development and can't be considered to be a particularly polished or even stable piece of software.

File Format Description

PBI files are Unicode files [2] (*i.e.* they use multi-byte character codes), encoded according to UTF-8, and always include a Unicode Byte Order Mark (BOM).

PBI files are structured into lines. Line are terminated by a NLF (new-line function) — any characters that are not followed by an NLF do not form a PBI line, and should cause a warning when encountered, and be ignored.

Each PBI file contain zero or more comment lines, followed by one or more data lines.

Syntax:

```
PBIfile:  UTF8-BOM PBI-file-marker NLF
          { comment-line NLF }
           data-line NLF { data-line NLF }
          ;
```

The *UTF8-BOM* is the Unicode BOM (U+FEFF) converted to UTF-8 (*i.e.* 0xEF 0xBB 0xBF).

```
PBI-file-marker
          : '#PBI' U+0020 version
          ;

version   : digits '.' digits
          ;

digits    : digit { digit }
          ;

digit     : '0' | '1' | '2' | '3' | '4' | '5' | '6' | '7' | '8' | '9'
          ;
```

The *PBI file marker* identifies the file as a PBI file. The *version* identifies the file as conforming to a version of this document. At present, the only *version* that can be used is '1.1' and '1.2'. (*version* '1.0' is obsolete.) The version string is interpreted as a sequence of characters, not as a floating-point number. '1.20' is not the same as '1.2', nor is '01.2'.

Although the *PBI file marker* may look like a c*omment-line*, it must not be assumed to be one.

```
NLF       : U+000D
          | U+000A
          | U+000D U+000A
          | U+0085
          | U+2028
          ;
```

The *NLF* terminates each line of the file. 'Lines' not followed by an *NLF* (at the very end of a file, for example), are not legal PBI lines, and must be ignored, preferrably after a warning message.

There is no requirement that the same *NLF* must be used throughout the file: line 1 may be followed by U+000D, line 2 by U+000A.

```
comment-line
        : '#' { any-character-except-NLF }
        ;
```

Comment lines can only appear at the beginning of PBI files, in the lines immediately after the *PBI marker*. Comments are information from the creator or editor of the file to future users; they may not be removed or reformatted gratuitously.

Much of the information encoded in a PBI file is intended for human interpretation only, and not for computer program interpretation. This is a fundamental design decision.

Each data line of the file contains a record of a chess composition. The format of a data line is:

```
data-line
        : names ':' position ':' stipulation ':' used-source ':'
           referenced-sources ':' awards ':' keymove ':' status ':' comment
        ;

names   : [ name { ';' name } ]
        ;
```

A *name* is a name, pseudonym or other designation of an author of the problem in question. Multiple *name*s are separated by a semicolon. The characters ':' and ';' may not appear in the actual name strings, as they are used as delimiters. The character '\' is similarly reserved to indicate encoded characters. If they are required in some particular name, they must be expressed in encoded form.

There is no further subformat of a *name*. In particular, there is no defined way of automatically identifying first name(s) and family name and/or titles or honorifics. For each particular file, a certain convention may apply and be documented in the comments, but it must be clear that such conventions have no support from this specification.

The names field may be empty.

Examples:

```
M. Abbott:1B6/7Q/4k3/4P2K...
:1B6/7Q/4k3/4P2K...
Bettman, H;E. Bettman:8/7p/Q4Bsk...
“Noli me Tangere”:1p6...
```

Encoded Characters

The purpose of encoding certain characters is to make it possible to use practically all characters in the character set available for problem-related information, even those characters that are used to delimit record fields.

Encoded characters are expressed as the special character ‘\’ followed by an ‘x’, followed by two hexadecimal digits. (Two digits are enough, as all reserved characters are present in the Unicode C0 table).

Encoded characters can appear in any field of a data line.

Example:

```
S\x3at John:8/8/8/3R4...          ‘S:t John’
```

The *position* is a Forsyth string, using English piece letters (PRSBQK and prsbqk), describing the problem, with S/s being used to identify knights. It can also be empty, indicating a problem that is not expressed as a chessboard position.

```
position: [ Forsyth-string ]
        ;
```

The Forsyth string must be complete, *i.e.* it is not allowed to leave out trailing empty parts of rows. It need not be normalized, *i.e.* it is OK to specify single empty squares as

```
11111111/11111111/11111111/11111111/11111111/... etc
```

instead of combining them as usual:

```
8/8/8/8/8/8/8/8
```

In the present version of the file format, there is no support for non-orthodox pieces. (Possible future direction: allow positions described by other means.)

The *stipulation* is a string formulating the stipulation of the problem. There is no defined method of interpreting a stipulation automatically.

```
stipulation
        : [ string ]
        ;
```

A possible convention would be to use common stipulation forms (*e.g.* #2, h#5, s#4, r#2, +) where they can be used, and express remaining stipulation in textual form. However, such convention has no support from this file format: it should preferrably be documented in the file commentary.

Examples:

```
...:#2:...
...:White moves and mates Black in two moves:...
...:Schwarz am zuge macht daṣ Spiel unentschieden:...
...:๓ ทีหมากดำหนี ๔ ทีหมากขาไล:
```

The last example is given with some reservations for correctness. It has been taken from a book on what seems to be chess problems in Thai language. It is included mainly to illustrate that although PBI files are expected to be well-defined, the actual contents of the fields of a data line is not guaranteed to be comprehensible to everyone, although it *is* expected that it should be an accurate transcription of the contents of the used source.

The publication from which the problem was taken is stated in the *used source* field.

```
used-source
        : [ source-reference ]
        ;

source-reference
        : [ nr ] '|' [ title ] '|' [ date ] '|' [ page ]
        ;

nr      : string ;
title   : string ;
date    : string ;
page    : string ;
```

Although it is permitted to leave the *used source* field empty, quality PBI files should state the sources they are based on.

(As the character '|' is used as a field separator in this field, any use of that character in the field data must be in encoded form.)

The different subfields are expected to be used as follows:

nr	The 'number' assigned to the problem
title	The title of the source. For book sources, include author(s).
date	The date of the source
page	The page number(s) in the source

Note again that there is no expectation that any of these subfields should be possible to interpret. A *nr* may be numerical, but it may equally well be '10a', or 'cxxiv', or expressed in some other form only interpretable by a human reader. If the record is converted from a format where nr, date and page cannot easily be identified, it is recommended to use the *title* subfield for all source references. (Possible future direction: This field may be totally unstructured, *i.e.* without any subfields.)

Any or all of the subfields may be empty. If all subfields are empty, the *source reference* '|||' may be replaced by ''.

Examples:

```
...:68|White\x3aSam Loyd and his Chess Problems|1962|:...
...:1142|Chess-Monthly vol. XI||:...
...:A|Russ\x3a:Miniature Chess Problems||
...:The fourth problem from the end|American Chess-Nuts|1868|:...
```

Any referenced sources are placed in the *referenced sources* field.

```
referenced-sources
        : [ source-reference { ';' source-reference } ]
        ;
```

Examples:

```
...:|Chess Amateur|1922|45:...
...:|Schachzeitung ?||;? 200|Sissa|July, 1868|:...
```

(Note that in this field, presence of ';' needs to be encoded.)

Any award(s) are described in the *awards* field

```
awards  : [ award { ';' award } ]
        ;

award   : nr '|' rank '|' tourney '|' date
        ;
```

```
nr       : string ;
rank     : string ;
tourney  : string ;
date     : string ;
```

Again, as for source references, there is no guarantee that the separate subfields can be easily interpreted by computer, and also that any convention used in any particular file should be documented. (Possible future direction: This field will allow for completely unstructured data, *i.e.* without any defined subfields.)

Examples

```
...:2|Pr|Problem|1960:...
...:|hm|American Chess|1960:...
...:1|dícséret|Magyar Sakkélet|1974:...
```

Version 1.1: The *keymove* field is reserved for future use: it must be empty.

Version 1.2: The *keymove* is a string containing the key move of the problem as published in the used source. The purpose of the field is to provide documentation on which to base a decision if a problem has been transcribed correctly or not. The information is for human eyes only: it should not be assumed to be easily interpreted by a computer progam. The convention used should be documented in the comment section of the file. Note that problems with more than one intended solution will have a *keymove* field specifying more than one key move. (Possible future direction: this field may be extended to allow the entire printed solution of the composition: this is particularly important for endgame studies. At present, they are best placed in the *comment* field.)

```
keymove
         : [ string ]
         ;
```

Examples

```
...:Dh7-b7:...
...:Q-K sq:...
...:Je6\x3a atd:...
```

Version 1.1: The *status* field is reserved for future use: it must be empty.

Version 1.2: The *status* field is a string documenting any verification the problem has been subjected to, and is intended to give additional confidence to the user of the PBI file that the problem has been transcribed correctly.

```
status   : [ string ]
         ;
```

Although the *status* field is defined to be a string, only a set of single-character values are currently defined:

The empty *status* field indicates that there is no further information about verification, and typically indicates that no further test have been performed.

Apart from the empty field, the following values are defined, all indicating that verification has indeed been performed and showed that

'!'	the problem is correct
'*'	the problem has multiple solutions above that documented in the keymove field.
'+'	the problem has no solution
'$'	the problem has a shorter solution
'?'	there is something else wrong with the keymove

Any other contents is undefined, and need not be preserved by PBI-file processing software. (Possible future direction: allow for free-format contents of this field in addition to the already specified values.)

EBNF Notation

This section should not be considered to be a formally complete description of the syntax used for someone entirely unacquainted with EBNF, but rather a short collection of reminders for the readers who have encountered EBNF before and may need a brief referesher.

Terminals

Printable characters from the Unicode C0 page (U+0000 - U+007F). They are written within single quotes.

```
'0'   'A'   '#'
```

Sequences of such terminals are collected into strings:

```
'PBI'   'A string containing internal spaces'
```

Non-printable characters are given as Unicode U-codes:

```
U+000D   U+0020
```

Non-terminals

Non-terminals are written as (more or less) descriptive words using hyphens instead of spaces:

```
a-more-or-less-descriptive-non-terminal
```

Metacharacters

Productions are always terminated by a semicolon.

```
non-terminal : terminals-and-non-terminals ;
```

The colon is used in productions to separate the non-terminal being defined (to the left) from its definition (to the right).

```
production : the-definition ;
```

The vertical bar indicates selection:

```
digit :    ‘0’ | ‘1’ | ‘2’ | ‘3’ | ‘4’ | ‘5’ | ‘6’ | ‘7’ | ‘8’ | ‘9’ ;
```

Brackets indicate optional contents

```
comment : ‘#’ [ anything-except-NLF ] ;
```

Braces indicate optional repetition

```
comment : ‘#’ { any-single-U-code-except-NLF } ;
```

Whitespace is insignificant as are line-breaks. Longer productions are generally broken over several lines if it seems likely that that improves clarity. Similarly, the terminating semicolon is usually placed on a separate line.

References

1. Thomas Maeder: *Problem.XML.* Project home web page is at *http://problem-xml.sourceforge.net/*

2. *The Unicode Standard 4.0,* Addison-Wesley, 2003. See also *http://www.unicode.org/*