4. Headers

The first two sections of this chapter define the structure and content of header keyword records. This is followed in Sect. 4.3 with recommendations on how physical units should be expressed. The final section defines the mandatory and reserved keywords for primary arrays and conforming extensions.

4.1. Keyword records

4.1.1. Syntax

Each 80-character header keyword record *shall* consist of a keyword name, a value indicator (only required if a value is present), an *optional* value, and an *optional* comment. Keywords *may* appear in any order except where specifically stated otherwise in this standard. It is *recommended* that the order of the keywords in *FITS* files be preserved during data processing operations because the designers of the *FITS* file may have used conventions that attach particular significance to the order of certain keywords (e.g., by grouping sequences of `COMMENT` keywords at specific locations in the header, or appending `HISTORY` keywords in chronological order of the data processing steps) or using `CONTINUE` keywords to generate long string keyword values.

A formal syntax, giving a complete definition of the syntax of *FITS* keyword records, is given in Appendix A. It is intended as an aid in interpreting the text defining the standard.

In earlier versions of this standard a FITS keyword, assumed as an item whose value is to be looked up by name (and presumably assigned to a variable) by a FITS reading program, was associated one to one to a single header keyword record. With the introduction of continued string keywords (see 4.2.1.2), such FITS keywords may span more than one header keyword records, and the value *shall* created by concatenation as explained in 4.2.1.2.

4.1.2. Components

4.1.2.1. Keyword name (bytes 1 through 8)

The keyword name *shall* be a left justified, 8-character, space-filled, ASCII string with no embedded spaces. All digits 0 through 9 (decimal ASCII codes 48 to 57, or hexadecimal 30 to 39) and upper case Latin alphabetic characters `'A'` through `'Z'` (decimal 65 to 90 or hexadecimal 41 to 5A) are permitted; lower case characters *shall not* be used. The underscore (`'_'`, decimal 95 or hexadecimal 5F) and hyphen (`'-'`, decimal 45 or hexadecimal 2D) are also permitted. No other characters are permitted.[1] For indexed keyword names that have a single positive integer index counter appended to the root name, the counter *shall not* have leading zeroes (e.g.,`NAXIS1`, not `NAXIS001`). Note that keyword names that begin with (or consist solely of) any combination of hyphens, underscores, and digits are legal.

4.1.2.2. Value indicator (bytes 9 and 10)

If the two ASCII characters `'=␣'` (decimal 61 followed by decimal 32) are present in bytes 9 and 10 of the keyword record this indicates that the keyword has a value field associated with it, unless it is one of the commentary keywords defined in Sect. 4.4.2 (i.e., a `HISTORY`, `COMMENT`, or completely blank keyword name) which by definition have no value.

4.1.2.3. Value/comment (bytes 11 through 80)

In keyword records that contain the value indicator in bytes 9 and 10, the remaining bytes 11 through 80 of the record *shall* contain the value, if any, of the keyword, followed by *optional* comments. In keyword records without a value indicator, bytes 9 through 80 *should* be interpreted as commentary text, however, this does not preclude conventions that interpret the content of these bytes in other ways.

The value field, when present, *shall* contain the ASCII text representation of a literal string constant, a logical constant, or a numerical constant, in the format specified in Sect. 4.2. The value field *may* be a null field; i.e., it *may* consist entirely of spaces, in which case the value associated with the keyword is undefined.

The mandatory *FITS* keywords defined in this standard *must not* appear more than once within a header. All other keywords that have a value *should not* appear more than once. If a keyword does appear multiple times with different values, then the value is indeterminate.

If a comment follows the value field, it *must* be preceded by a slash (`'/'`, decimal 47 or hexadecimal 2F).[1] A space between the value and the slash is strongly *recommended*. The comment *may* contain any of the restricted set of ASCII text characters, decimal 32 through 126 (hexadecimal 20 through 7E). The ASCII control characters with decimal values less than 32 (including the null, tab, carriage return, and line feed characters), and the delete character (decimal 127 or hexadecimal 7F) *must not* appear anywhere within a keyword record.

4.2. Value

The structure of the value field depends on the data type of the value. The value field represents a single value and not an array of values.[1] The value field *must* be in one of two formats: fixed or free. The fixed-format is required for values of mandatory keywords and is *recommended* for values of all other keywords.

4.2.1. Character string

4.2.1.1 Single record string keywords

A character string value *shall* be composed only of the set of restricted ASCII text characters, decimal 32 through 126 (hexadecimal 20 through 7E) enclosed by single quote characters (“’”, decimal 39, hexadecimal 27). A single quote is represented within a string as two successive single quotes, e.g., O’HARA = `'O''HARA'`. Leading spaces are significant; trailing spaces are not. This standard imposes no requirements on the case sensitivity of character string values unless explicitly stated in the definition of specific keywords.

If the value is a fixed-format character string, the starting single quote character *must* be in byte 11 of the keyword record and the closing single quote *must* occur in or before byte 80. Earlier versions of this standard also *required* that fixed-format

[1] This requirement differs from the wording in the original *FITS* papers. See Appendix H.

characters strings *must* be padded with space characters to at least a length of eight characters so that the closing quote character does not occur before byte 20. This minimum character string length is no longer required, except for the value of the XTENSION keyword (e.g., 'IMAGE␣␣␣' and 'TABLE␣␣␣'; see Sect. 7) which *must* be padded to a length of eight characters for backward compatibility with previous usage.

Free-format character strings follow the same rules as fixed-format character strings except that the starting single quote character *may* occur after byte 11. Any bytes preceding the starting quote character and after byte 10 *must* contain the space character.

Note that there is a subtle distinction between the following three keywords:

```
KEYWORD1= ''                / null string keyword
KEYWORD2= '   '             / empty string keyword
KEYWORD3=                   / undefined keyword
```

The value of KEYWORD1 is a null, or zero length string whereas the value of the KEYWORD2 is an empty string (nominally a single space character because the first space in the string is significant, but trailing spaces are not). The value of KEYWORD3 is undefined and has an indeterminate data type as well, except in cases where the data type of the specified keyword is explicitly defined in this standard.

The maximum length of a string value that can be represented on a single keyword record is 68 characters, with the opening and closing quote characters in bytes 11 and 80, respectively. In general, no length limit less than 68 is implied for character-valued keywords.

4.2.1.2 Continued string keywords

Earlier versions of this Standard only defined single record string keywords as described in the previous section. The Standard now incorporates a convention (originally developed for use in *FITS* files from high energy astrophysics missions) for continuing arbitrarily long string values over multiple consecutive keyword records using the following procedure:

1. Divide the long string value into a sequence of smaller substrings, each of which is no longer than 67 characters in length. (Note that if the string contains any literal single quote characters, then these must be represented as a pair of single quote characters in the *FITS* keyword value, and these 2 characters must both be contained within a single substring).
2. Append an ampersand character (‘&’) to the end of each substring, except for the last substring. This character serves as a flag to *FITS* reading software that this string value *may* be continued on the following keyword in the header.
3. Enclose each substring with single quote characters. Non-significant space characters may occur between the ampersand character and the closing quote character.
4. Write the first substring as the value of the specified keyword.
5. Write each subsequent substring, in order, to a series of keywords that all have the name CONTINUE in bytes 1 through 8 and have space characters in bytes 9 and 10 of the keyword record. The substring may be located anywhere in bytes 11 through 80 of the keyword record and may be preceded by non-significant space characters starting in byte 11. A comment string may follow the substring; if present, the comment string must be separated from the substring by at least 1 space character followed by a forward slash character (‘/’).

The following keyword records illustrate a string value that is continued over multiple keyword records. (Note: the length of the substrings have been reduced to fit within the page layout):

```
WEATHER = 'Partly cloudy during the evening f&'
CONTINUE  'ollowed by cloudy skies overnight.&'
CONTINUE  ' Low 21C. Winds NNE at 5 to 10 mph.'
```

If needed, additional space for the keyword comment field can be generated by continuing the string value with one or more null strings, as illustrated schematically below:

```
STRKEY  = 'This keyword value is continued &'
CONTINUE  ' over multiple keyword records.&'
CONTINUE  '&' / The comment field for this
CONTINUE  '&' / keyword is also continued
CONTINUE  ''  / over multiple records.
```

FITS reading software can reconstruct the long string value by following an inverse procedure of checking if the string value ends with the ‘&’ character and is immediately followed by a conforming CONTINUE keyword record. If both conditions are true, then concatenate the substring from the CONTINUE record onto the previous substring after first deleting the trailing ‘&’ character. Repeat these steps until all subsequent CONTINUE records have been processed.

Note that if a string value ends with the ‘&’ character, but is not immediately followed by a conforming CONTINUE keyword, then the ‘&’ character should be interpreted as the literal last character in the string. Also, any ‘orphaned’ CONTINUE keyword records (formally not invalidating the FITS file, although likely representing an error with respect to what the author of the file meant) should be interpreted as containing commentary text in bytes 9 – 80 (similar to a COMMENT keyword).

4.4.2.4. Commentary keywords

These keywords provide commentary information about the contents or history of the *FITS* file and *may* occur any number of times in a header. These keywords *shall* have no associated value even if the value indicator characters '=␣' appear in bytes 9 and 10 (hence it is *recommended* that these keywords not contain the value indicator). Bytes 9 through 80 *may* contain any of the restricted set of ASCII text characters, decimal 32 through 126 (hexadecimal 20 through 7E).

In earlier versions of this standard continued string keywords (see 4.2.1.2) could be handled as commentary keywords if the relevant convention was not supported. Now `CONTINUE` keywords *shall* be honoured as specified in Section 4.2.1.2.

`COMMENT` keyword. This keyword *may* be used to supply any comments regarding the *FITS* file.

`HISTORY` keyword. This keyword *should* be used to describe the history of steps and procedures associated with the processing of the associated data.

Keyword field is blank. This keyword *may* be used to supply any comments regarding the *FITS* file. It is frequently used for aesthetic purposes to provide a break between groups of related keywords in the header.