# ASTRO+ DATABASE

# USER MANUAL

# (v1.0 November 2024)


# Amparo Marco and the ASTRO+ team

**0. Astro+: The largest High Mass Star Database.**
In this database, you can upload and restore your own spectra of high mass stars and besides search spectra of a large amount of these stars. All data will automatically be archived with their parameters from Simbad and Gaia. In addition, you may obtain their physical parameters calculated in a homogeneous way by using local programs (see Rubke et al. 2022).

**1. Access to the webpage:**
https:// astroplus.ua.es

**2. Register:** The first time you access the database, you must apply for a user account. You will receive this information from the administrator to sign in.

**3. Login:** Username and password

**4. Go to Upload:** You choose the data format to upload your data.

**4a. FITS Configuration:**

**FITS** (Flexible Image Transport System) is the data format most widely used within astronomy for transporting, analyzing, and archiving scientific data files and is designed to store scientific data sets consisting of multidimensional arrays (images) and 2-dimensional tables organized into rows and columns of information.
The FITS file format allows for much flexibility in how data is stored and accessed. Depending on the complexity of a given telescope and instrument setup, spectroscopic observations may be stored in a myriad of different ways and each professional observatory uses a different method to store such observations. This situation creates multiple issues when one attempts to compile large sets of data originating from different observatories and instruments. *To get around these complications, the ASTRO+ database defines several standard FITS data structures to allow users to upload files correctly and consistently to the database*.
To define these file types, the standards documented by the IAU FITS standard working group are followed as much as possible.

A FITS file is comprised of segments called Header/Data Units (**HDUs**):

1. The first HDU is called the `Primary HDU', or `Primary Array'. The primary data array can contain a **1-999** dimensional array of 1, 2 or 4 byte integers or 4 or 8 byte floating point numbers using IEEE representations. A typical primary array could contain a 1-D spectrum, a 2-D image, or a 3-D data cube. ***In the ASTRO+ database, we only allow the upload of 1-D spectra.***

2. Any number of additional **HDUs** may follow the primary array. These **additional HDUs** are referred to as FITS `**extensions**'. Three types of standard extensions are currently defined:

> Image Extensions contain a 0-999 dimensional array of pixels, similar to a primary array
> (header begins with EXTENSION = 'IMAGE ')
>
> ASCII Table Extensions store tabular information with all numeric information stored in ASCII formats. While ASCII tables are generally less efficient than binary tables,

they can be made relatively human readable and can store numeric information with essentially arbitrary size and accuracy (e.g., 16 byte reals).
(header begins with EXTENSION = 'TABLE ')

Binary Table Extensions store tabular information in a binary representation. Each cell in the table can be an array, but the dimensionality of the array must be constant within a column. The strict standard supports only one-dimensional arrays, but a convention to support multi-dimensional arrays is widely accepted.
(header begins with EXTENSION = 'BINTABLE')

A FITS file composed of only *the primary HDU* is sometimes referred to as a *Basic FITS file*, or a *Single Image FITS (SIF) file* **[only Extension 0]**, and a FITS file containing one or more additional **HDUs** called FITS extensions following the primary HDU is sometimes referred to as a *Multi-Extension FITS (MEF) file* **[Extension 0, Extension 1, Extension 2, Extension 3 …].** Each extension or additional HDU has a header (mandatory) and optionally one (single data array) or more data blocks (multiple data arrays) containing data as well. These data blocks can be defined with specific names: WAVE, FLUX, ERROR in the columns of a table or with no names.

At this point, we need to study the structure of our FITS data before continuing. With this purpose, we can use the following sentences in the iPython code:

**In [1]: from astropy.io import fits**

**In [2]: fitin=fits.open("FITS filename")**

**In [3]: fitin.info()**

**In [4]: fitin[Number of extension].data**

**In [5]: fitin[Number of extension].columns**

**In [6]: fitin[Number of extension].header**

---

**In [3]: fitin.info()** we obtain the number of extensions. In each extension, we can explore the data, columns and header using the following sentences:

**In [4]: fitin[Number of extension].data**

**In [5]: fitin[Number of extension].columns**

**In [6]: fitin[Number of extension].header**

Next, we describe the characteristics of 10 different examples from spectra coming from a variety of observatories:

**Example 1**

**In [3]: fitin.info()**

No. Name Ver Type Cards Dimensions Format

- 0 PRIMARY 1 PrimaryHDU 215 (400,) float32 (only extension 0)

**In [4]: fitin[0].data**

- array([3434.2231, 3436.472 , 3389.4485, 3465.2449, 3454.6982, 3393.2725,

……,

3255.3958, 3231.2266, 3293.1692, 3373.4382], dtype=float32) (only one array containing the flux)

**In [5]: fitin[0].columns**

- 'PrimaryHDU' object has no attribute 'columns' (no names in columns). In this case, we have to explore the names of the parameters: **Ref. Position** and **Pixel Scale** with the next command as well.

**In [6]: fitin[0].header**

We explore the names of the following parameters:

**Main_ID:** Object

**RA:** RA

**DEC:** DEC

**Observation Date:** DATE

**Telescope:** TELESCOP

**Instrument:** INSTRUME

**Exposure Time:** EXPTIME

---

**Ref. Position:** CRVAL1

**Pixel Scale:** CDELT1 or CD1_1

**Example 2**

**In [3]: fitin.info()**

No. Name Ver Type Cards Dimensions Format

- 0 PRIMARY 1 PrimaryHDU 213 () (extension 0)

- 1 1 ImageHDU 35 (1023,) float32 (extension 1)

**In [4]: fitin[1].data** (we have data only in extension 1)

array([1.0274824 , 1.0468955 , 1.0186049 , ..., 0.8481021 , 0.85893154,

0.8707505 ], dtype=float32)

**In [5]: fitin[1].columns**

- '1 ImageHDU' object has no attribute 'columns' (no names in columns). In this case, we have to explore the name of the parameters: **Ref. Position** and **Pixel Scale** with the next command as well.

**In [6]: fitin[0].header** (Ext 0 contains only the old header)

**In [6]: fitin[1].header** (Ext 1 contains the new header where we find the names of the following parameters)

**Main_ID:** Object

**RA:** RA

**DEC:** DEC

**Observation Date:** DATE-OBS

**Telescope:** TELESCOP

**Instrument:** INSTRUME

**Exposure Time:** EXPTIME

---

**Ref. Position:** CRVAL1

**Pixel Scale:** CDELT1 or CD1_1

**Example 3**

**In [3]: fitin.info()**

No. Name Ver Type Cards Dimensions Format

- 0 PRIMARY 1 PrimaryHDU 167 (155404, 2) float64 (only extension 0)

**In [4]: fitin[0].data**

array([[2.1376055 , 2.1372672 , 2.1369289 , ..., 1.0182667 , 1.02216799,
1.0271952 ],
[0. , 0. , 0. , ..., 0.47679529, 0.47863132,
0.48099464]]) (two arrays containing the flux and flux_error)

**In [5]: fitin[0].columns**

'PrimaryHDU' object has no attribute 'columns' (notnames in columns). In this case, we have to explore the name of the parameters: **Ref. Position** and **Pixel Scale** wiht the next command as well.

**In [6]: fitin[0].header**

We explore the names of the following parameters:

**Main_ID:** Object

**RA:** RA

**DEC:** DEC

**Observation Date:** DATE-OBS

**Telescope:** TELESCOP

**Instrument:** INSTRUME

**Exposure Time:** EXPTIME

---

**Ref. Position:** CRVAL1

**Pixel Scale: CDELT1 or CD1_1**

**Example 4**

**In [3]: fitin.info()**

No. Name Ver Type Cards Dimensions Format

- 0 PRIMARY 1 PrimaryHDU 472 (335519,) float64 (extension 0)

- 1 INSTRUMENTCONFIG.XML 1 BinTableHDU 11 173R x 1C [1020A] (extension 1)

**In [4]: fitin[0].data**

array([ nan, nan, nan, ..., 2058.63569211,

2023.55079739, 1935.64915593]) (we have data only in extension 0; one array containing flux)

**In [4]: fitin[1].data**

uded) with first pixel counted as 1',),

('<middleCcdRowOrderPositions>middleCcdRowOrderPositions.fits</middleCcdRowOrderPositions>',),

(' \t</LRF_WRF>',), (' \t</values>',),

('</pipeline>',)], dtype=(numpy.record, [('xml', 'S1020')])) (one table with information about the configuration of the observation)

**In [5]: fitin[0].columns**

'PrimaryHDU' object has no attribute 'columns' (no names in columns). In this case, we have to explore the name of the parameters: **Ref. Position** and **Pixel Scale** with the next command as well.

**In [5]: fitin[1].columns**

ColDefs(

name = 'xml'; format = '1020A'  (Information about the table format)

**In [6]: fitin[0].header**

We explore the names of the following parameters:

**Main_ID:** Object

**RA:** RA

**DEC:** DEC

**Observation Date:** DATE

**Telescope:** TELESCOP

**Instrument:** INSTRUME

**Exposure Time:** EXPTIME

---

**Ref. Position:** CRVAL1

**Pixel Scale:** CDELT1 or CD1_1

**Example 5**

**In [3]: fitin.info()**

```
 No.    Name      Ver    Type      Cards   Dimensions   Format
-  0  FLUX        1 PrimaryHDU     593   (12854,)   float32  (extension 0)
-  1  ERRS        1 ImageHDU        23   (12854,)   float32   (extension 1)
-  2  QUAL        1 ImageHDU        23   (12854,)   int32     (extension 2)
```

**In [4]: fitin[0].data**

```
array([-5.1916034e-15,  4.8787698e-14, -7.1954798e-14, ...,
        1.0968022e-14,  1.1460177e-14,  1.1645745e-14], dtype=float32)
```
(one array containing the flux)

**In [4]: fitin[1].data**

```
array([1.5614693e-13, 7.2924644e-14, 7.2841107e-14, ..., 2.7457476e-16,
       2.7599531e-16, 2.7777667e-16], dtype=float32)
```
(one array containing the flux error)

**In [4]: fitin[2].data**

```
Out[22]: array([0, 0, 0, ..., 0, 0, 0], dtype=int32)
```
(one array without data)

**In [5]: fitin[0].columns**

'PrimaryHDU' object has no attribute 'columns'  (no names in columns). In this case, we have to explore the name of the parameters: **Ref. Position** and **Pixel Scale** with the next command as well.

**In [5]: fitin[1].columns**

'ImageHDU' object has no attribute 'columns'  (no names in columns). In this case, we have to explore the name of the parameters: **Ref. Position** and **Pixel Scale** with the next command as well.

**In [5]: fitin[2].columns**

'ImageHDU' object has no attribute 'columns'  (no names in columns). In this case, we have to explore the name of the parameters: **Ref. Position** and **Pixel Scale** with the next command as well.

**In [6]: fitin[0].header**

We explore the names of the following parameters:

**Main_ID:** Object

**RA:** RA

**DEC:** DEC

**Observation Date:** DATE

**Telescope:** TELESCOP

**Instrument:** INSTRUME

**Exposure Time:** EXPTIME

---

**Ref. Position:** CRVAL1

**Pixel Scale:** CDELT1 or CD1_1

**Example 6**

**In [3]: fitin.info()**

No.  Name  Ver  Type  Cards  Dimensions  Format

- 0  PRIMARY  1 PrimaryHDU  215  (4200, 1, 2)  float32  (extension 0)

**In [4]: fitin[0].data**

array([[[ 1.  ,  1.  ,  1.  , ...,  1.  ,

1.  ,  1.  ]],

[[-71.858635, -71.86063 , -71.8626  , ..., -93.83298 ,

-96.106705, -90.92938 ]]], dtype=float32) (We have 2 arrays containing normalized flux and flux_error; each array is enclosed by a double square bracket)

**In [5]: fitin[0].columns**

'PrimaryHDU' object has no attribute 'columns' (no names in columns). In this case, we have to explore the name of the parameters: **Ref. Position** and **Pixel Scale** with the next command as well.

**In [6]: fitin[0].header**

We explore the names of the following parameters:

**Main_ID:** Object

**RA:** RA

**DEC:** DEC

**Observation Date:** DATE

**Telescope:** TELESCOP

**Instrument:** INSTRUME

**Exposure Time:** EXPTIME

---

**Ref. Position:** CRVAL1

**Pixel Scale:** CDELT1 or CD1_1

**Example 7**

**In [3]: fitin.info()**

No. Name Ver Type Cards Dimensions Format

- 0 PRIMARY 1 PrimaryHDU 419 (2860,) float64 (extension 0)
- 1 SKY SUBTRACTED 1 ImageHDU 11 (2860,) float64 (extension 1)
- 2 MEDIAN SKY 1 ImageHDU 11 (2860,) float64 (extension 2)
- 3 PROCESSED ERROR 1 ImageHDU 11 (2860,) float64 (extension 3)
- 4 ORIGINAL ERROR 1 ImageHDU 11 (2860,) float32 (extension 4)
- 5 FIBER_SETUP 1 BinTableHDU 37 126R x 14C [1J, 1J, 1J, 1J, 1J, 11A, 1J, 6A, 1D, 1D, 1D, 1D, 1D, 1D] (extension 5)

**In [4]: fitin[0].data**

array([647.77288437, 646.16413498, 650.79735184, ..., 447.62348175,

426.9555397 , 428.03517532]) (one array containing flux)

**In [4]: fitin[3].data**

array([27.82828034, 33.53277702, 33.51065329, ..., 20.81859773,

19.33348236, 20.72485267]) (one array containing processed flux_error)

**In [4]: fitin[4].data**

array([24.649857, 25.01904 , 24.849052, ..., 19.061787, 18.606007,

18.95015 ], dtype=float32) (one array containing original flux_error)

**In [5]: fitin[0].columns**

'PrimaryHDU' object has no attribute 'columns' (no names in columns). In this case, we have to explore the name of the parameters: **Ref. Position** and **Pixel Scale** with the next command as well.

**In [5]: fitin[3].columns**

'PrimaryHDU' object has no attribute 'columns' (no names in columns). In this case, we have to explore the name of the parameters: **Ref. Position** and **Pixel Scale** with the next command as well.

**In [5]: fitin[4].columns**

'PrimaryHDU' object has no attribute 'columns' (no names in columns). In this case, we have to explore the name of the parameters: **Ref. Position** and **Pixel Scale** with the next command as well.

**In [6]: fitin[0].header**

We explore the names of the following parameters:

**Main_ID:** Object

**RA:** RA

**DEC:** DEC

**Observation Date:** DATE

**Telescope:** TELESCOP

**Instrument:** INSTRUME

**Exposure Time:** EXPTIME

---

**Ref. Position:** CRVAL1

**Pixel Scale:** CDELT1 or CD1_1

**Example 8**

**In [3]: fitin.info()**

No. Name Ver Type Cards Dimensions Format

- 0 PRIMARY 1 PrimaryHDU 278 (215930, 2) float64 (extension 0)

**In [4]: fitin[0].data**

array([[ 1.33151500e+00, 7.01154950e-01, 3.61924034e-01, ...,

-1.37273511e+02, -1.37279891e+02, -1.37286271e+02],

[ 5.29002964e-01, 2.78829843e-01, 1.44063950e-01, ...,

8.36747214e-02, 8.84862766e-02, 9.27733853e-02]]) (two arrays containing flux and flux error)

**In [5]: fitin[0].columns**

'PrimaryHDU' object has no attribute 'columns' (no names in columns). In this case, we have to explore the name of the parameters: **Ref. Position** and **Pixel Scale** with the next command as well.

**In [6]: fitin[0].header**

We explore the names of the following parameters:

**Main_ID:** Object

**RA:** RA

**DEC:** DEC

**Observation Date:** DATE

**Telescope:** TELESCOP

**Instrument:** INSTRUME

**Exposure Time:** EXPTIME

---

**Ref. Position:** CRVAL1

**Pixel Scale:** CDELT1 or CD1_1

**Example 9**

**In [3]: fitin.info()**

No. Name Ver Type Cards Dimensions Format

- 0 PRIMARY 1 PrimaryHDU 3347 () (extension 0)

- 1 SPECTRUM 1 BinTableHDU 46 1R x 3C [313162D, 313162E, 313162E] (extension 1)

**In [4]: fitin[1].data**

dtype=(numpy.record, [('WAVE', '>f8', (313162,)), ('FLUX', '>f4', (313162,)), ('ERR', '>f4', (313162,))])) (in extension 0, we have no data; extension 1 contains 3 arrays with wavelength, flux and flux error )

**fitin[1].data.field('WAVE')**

array([[3781.31, 3781.32, 3781.33, ..., 6912.91, 6912.92, 6912.93]])

(the array is formed by a growing sequence of numbers)

**fitin[1].data.field('FLUX')**

array([[-86.441864, 50.30547 , 219.69885 , ..., 150.27798 , 98.98375 ,

70.85126 ]], dtype=float32)

**fitin[1].data.field('ERR')**

array([[nan, nan, nan, ..., nan, nan, nan]], dtype=float32)

**In [5]: fitin[1].columns**

ColDefs(

name = 'WAVE'; format = '313162D'; unit = 'Angstrom'

name = 'FLUX'; format = '313162E'; unit = 'adu'

name = 'ERR'; format = '313162E'; unit = 'adu'

) (The columns are defined with names: WAVE, FLUX and ERR)

**In [6]: fitin[0].header**

We explore the names of the following parameters:

**Main_ID:** Object

**RA:** RA

**DEC:** DEC

**Observation Date:** DATE

**Telescope:** TELESCOP

**Instrument:** INSTRUME

**Exposure Time:** EXPTIME

**Example 10**

**In [3]: fitin.info()**

No. Name Ver Type Cards Dimensions Format

0 PRIMARY 1 PrimaryHDU 13 ()

1 1 ImageHDU 7 (141391, 3) float64

2 1 ImageHDU 141 ()

**In [4]: fitin[1].data**

array([[3.70385693e+03, 3.70388216e+03, 3.70390739e+03, ...,

7.27113753e+03, 7.27116276e+03, 7.27118799e+03],

[7.46043223e-01, 8.49061882e-01, 8.98117545e-01, ...,

1.08175183e+03, 1.08182535e+03, 1.08189886e+03],

[0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,

0.00000000e+00, 0.00000000e+00, 0.00000000e+00]]) (in extensions 0 and 2, we have no data; extension 1 contains 3 arrays with wavelength, flux and flux error without data.The first array which correspond to the wavelength is formed by a growing sequence of numbers).

**In [5]: fitin[1].columns**

'ImageHDU' object has no attribute 'columns' (no names in columns).

---

Once we know the structure of our FITS files, we are ready to upload the spectra using the parameters found above

**4a.1 Units:** choose the units of wavelength in the spectrum. There are different units for the wavelength scale

**4a.2 Type of data:** choose between: Normalized; Not Normalized; Flux calibrated. There are three different data types for your spectra

**4a.3 Batch Name:** choose the name for your data in the upload. It is the visible name for the whole data block (one or more spectra) uploaded every time

**4a.4 Configuration Templates:** choose between ***Custom*** or 3 options corresponding to different observatories or telescopes: **ESO, NOT and Mercator, WHT and INT**. There are some template examples to help the upload of spectra with standard configurations corresponding to different observatories or telescopes. They can be modified to fit the structure of your data

**4a.5 FITS Configuration_1 Header**: You must check that your file header is not empty and that it contains the following parameters and in which extension: ***MAIN_ID; RA; DEC; OBSERVATION DATE; TELESCOPE; INSTRUMENT; EXPOSURE TIME***. *You must write their exact names from your file header in the white spaces*. Having this information, you can continue. If your file header is empty, you must use the original FITS files from which the spectrum was extracted, which contain a non-empty file header, and add the header information to the FITS files you are trying to upload. The following program in python "*Recover_header.py*" can be used to add this information to the final FITS files. This program is provided with this document. You can also add keywords to the header by hand.

**4a.6 FITS Configuration_2 Wavelength**:

**4a.6.1** If you have column names defining your data, you choose ***Array Method*** and write in the white space the name of the wavelength column used in the appropriate extension.

**4a.6.2** If the data in a given extension comes as a single data array corresponding to the wavelength (flux is another in extension), we must select ***Single data***. If, on the contrary, we have two or more data arrays, corresponding to dimension 0 (first array), dimension 1 (second array), etc …, we choose the ***Dimension*** for the wavelength data array. This array can be easily identified because it contains a monotonosly increasing sequence of numbers.

**4a.6.3** If your data do not contain names in columns or there is no array corresponding to wavelength, you must choose the ***Keyword Method.*** Choose the **Wavelength extension** (as a general rule, it is in the same extension as the flux) and write in the white spaces the names of **Ref. Position** and **Pixel scale** from the file header.

**4a.7 FITS Configuration_3 Flux**:

**4a.7.1** Select the ***Flux Extension***. If in this extension there is a single data array corresponding to the flux, we must select ***Single data***. If, on the contrary, we have two or more data arrays,, corresponding to dimension 0 (first array), dimension 1 (second array), etc …, we choose the ***Dimension*** for the flux data array.

Select the ***Flux Error***. If in this extension there is a single data array corresponding to the error flux, we must select ***Single data***. If, on the contrary, we have two or more data arrays, corresponding to dimension 0 (first array), dimension 1 (second array), etc …, we choose the ***Dimension*** for the error flux data array. In some cases, there is no Flux Error associated with the data, and then we select ***None***.

**4a.7.2** If you have column names defining your data, you choose ***Array Method*** and write in the white spaces the name of the **Flux** and **Flux Error** columns used in the appropriate extensions. In some cases, there is no Flux Error associated with the data, and then we select ***None***.

**4a.8. Upload.** We load the FITS spectra with the configuration selected in the template from the Browser and upload them.

**4b. ASCII Configuration:**

**4b.1 Units:** choose the units of wavelength in the spectrum. There are different units for the wavelength scale

**4b.2 Type of data:** choose between: Normalized; Not Normalized; Flux calibrated. There are three different types data for your spectra

**4b.3 Batch Name:** choose the name for your data in the upload. It is the visible name for the whole data block (one or more spectra) uploaded every time

**4b.4 Upload files:**

**4b.4.1 Download ASCII Parameters Template** and then fill the different fields. There are some mandatory fields: **FILENAME; MAIN_ID; DATE_OBS; RA; DEC; TELESCOP; INSTRUME; EXPTIME.** We can write the parameters for all spectra in the same template.

**4b.4.2** Finally, **Drag and drop the csv header and ascii files** selecting from the Browser

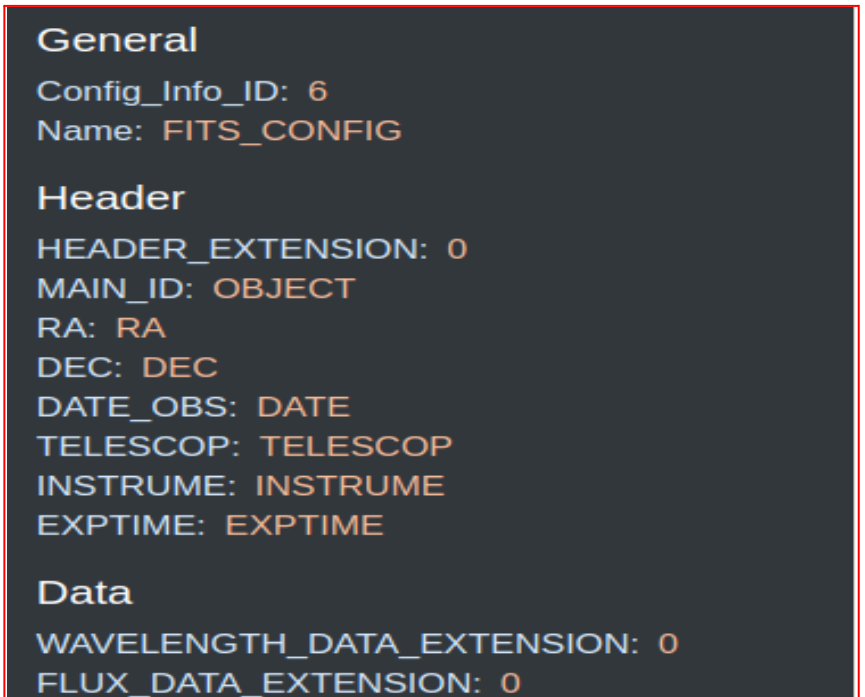
**5. Go to Upload tracking.** We can visualize the state of the upload process: **Processing**; **Processed with errors** and **Successfully processed**. When the spectra are processed, we open ***Details***. If the result is **Processed with errors,** we go to the **Batch Info** and read the **log details.** We have to understand the errors and correct them in the option **Reprocess.** But if the process is **Success,** we go to **Resolution** and write the approximate spectral resolution of your data in the white space. Then, we can visualize the spectra, as well as associated data and sky map from Simbad. When everything is alright, we only have to wait for the response of the administrator, who has to authorize the upload to the public server.

## APENDIX

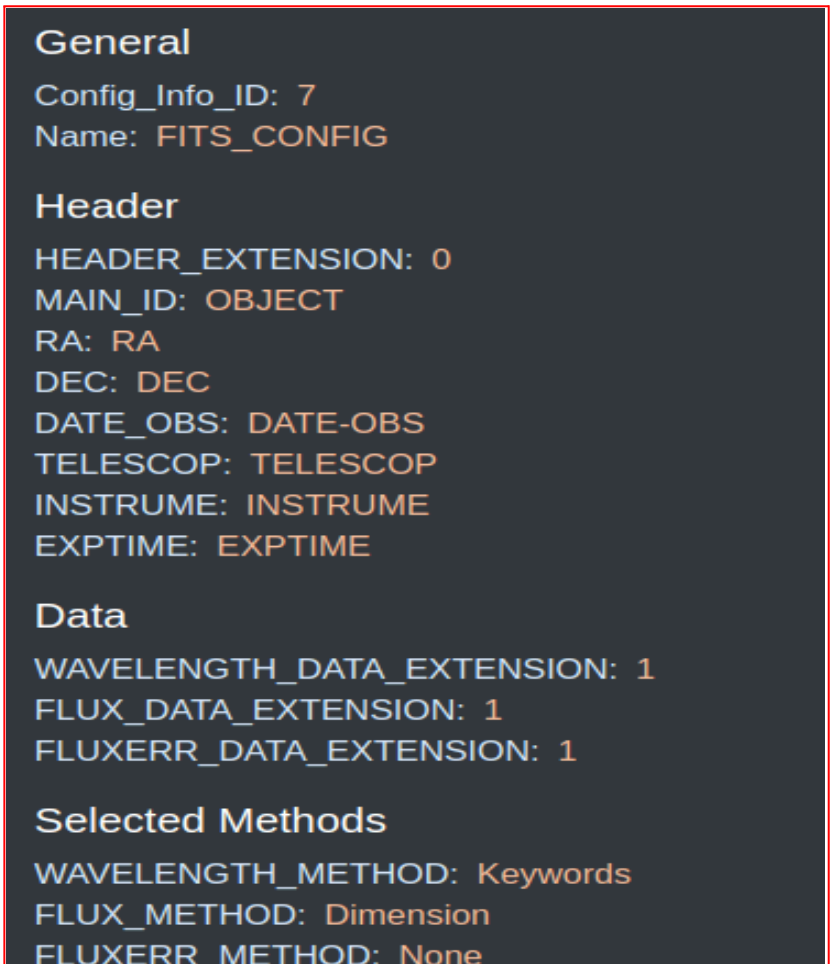
## UPLOADING SPECTRA FROM DIFFERENT TELESCOPES+INSTRUMENTS.

The following configurations correspond to the above examples with different headers:

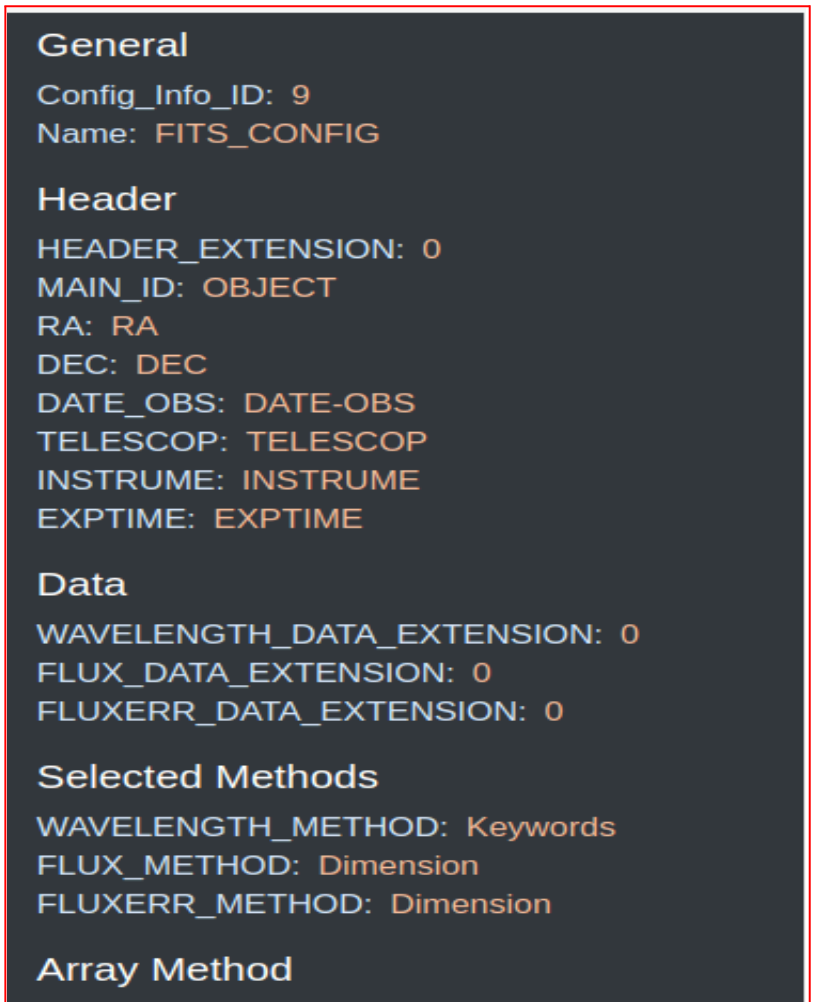
**Example 1**



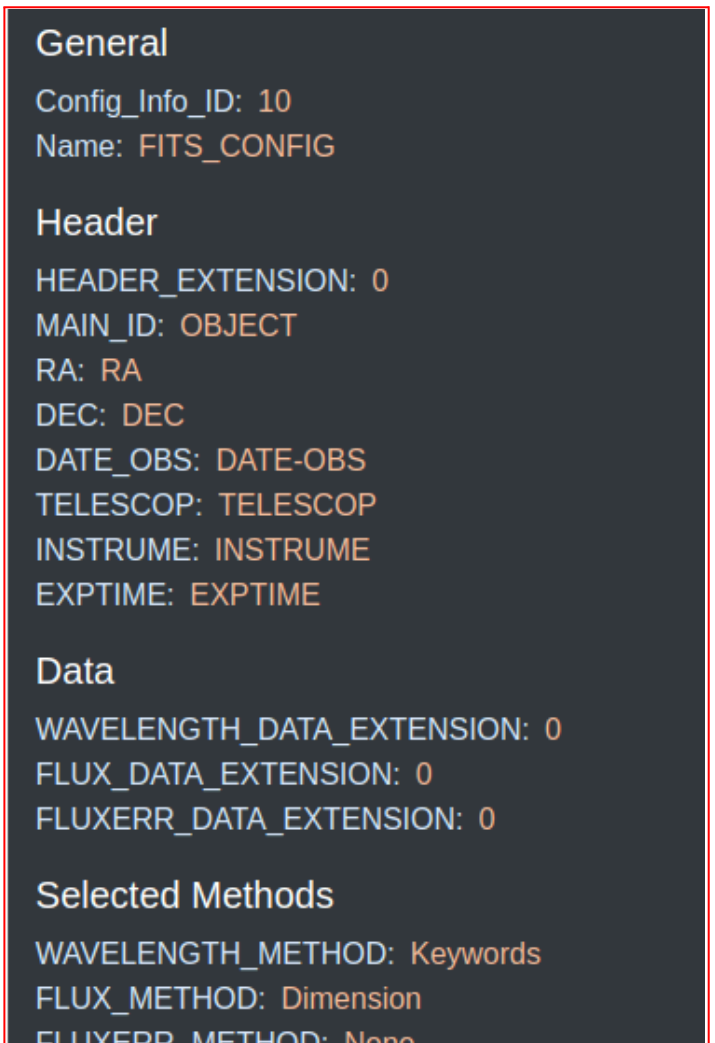
**Example 2**


General
Config_Info_ID: 7
Name: FITS_CONFIG
Header
HEADER_EXTENSION: 0
MAIN_ID: OBJECT
RA: RA
DEC: DEC
DATE_OBS: DATE-OBS
TELESCOP: TELESCOP
INSTRUME: INSTRUME
EXPTIME: EXPTIME
Data
WAVELENGTH_DATA_EXTENSION: 1
FLUX_DATA_EXTENSION: 1
FLUXERR_DATA_EXTENSION: 1
Selected Methods
WAVELENGTH_METHOD: Keywords
FLUX_METHOD: Dimension
FLUXERR_METHOD: None

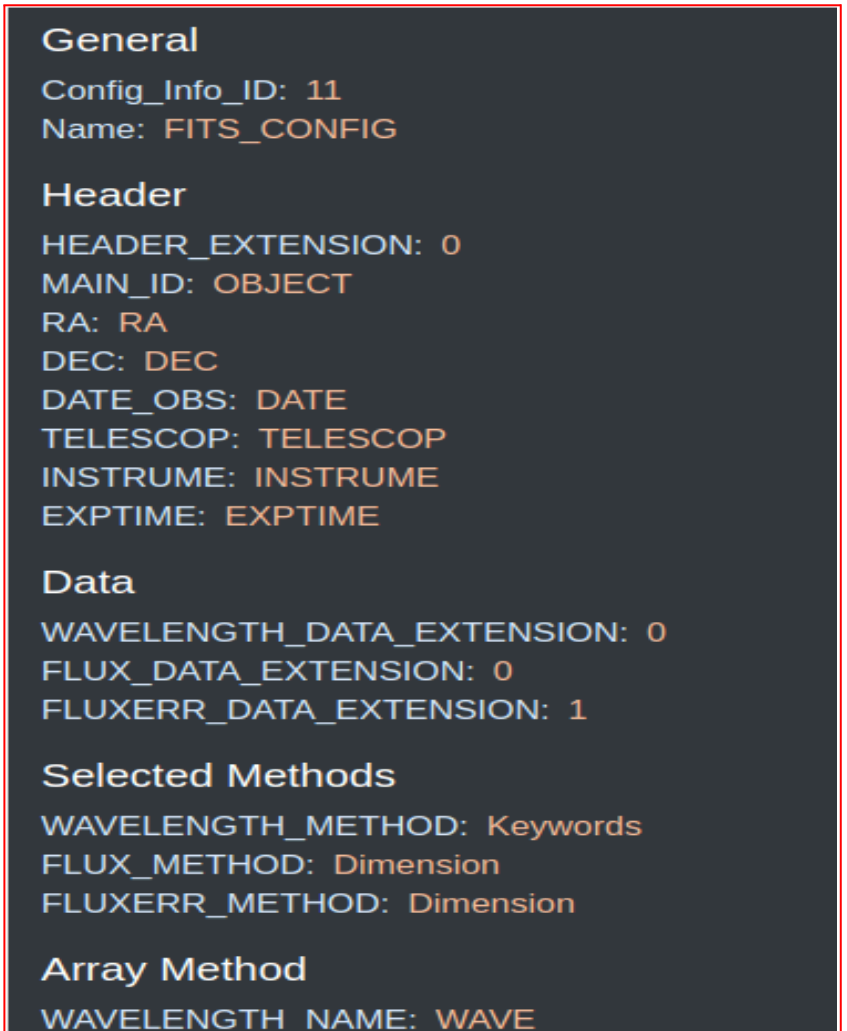
**Example 3**


General
Config_Info_ID: 9
Name: FITS_CONFIG
Header
HEADER_EXTENSION: 0
MAIN_ID: OBJECT
RA: RA
DEC: DEC
DATE_OBS: DATE-OBS
TELESCOP: TELESCOP
INSTRUME: INSTRUME
EXPTIME: EXPTIME
Data
WAVELENGTH_DATA_EXTENSION: 0
FLUX_DATA_EXTENSION: 0
FLUXERR_DATA_EXTENSION: 0
Selected Methods
WAVELENGTH_METHOD: Keywords
FLUX_METHOD: Dimension
FLUXERR_METHOD: Dimension
Array Method

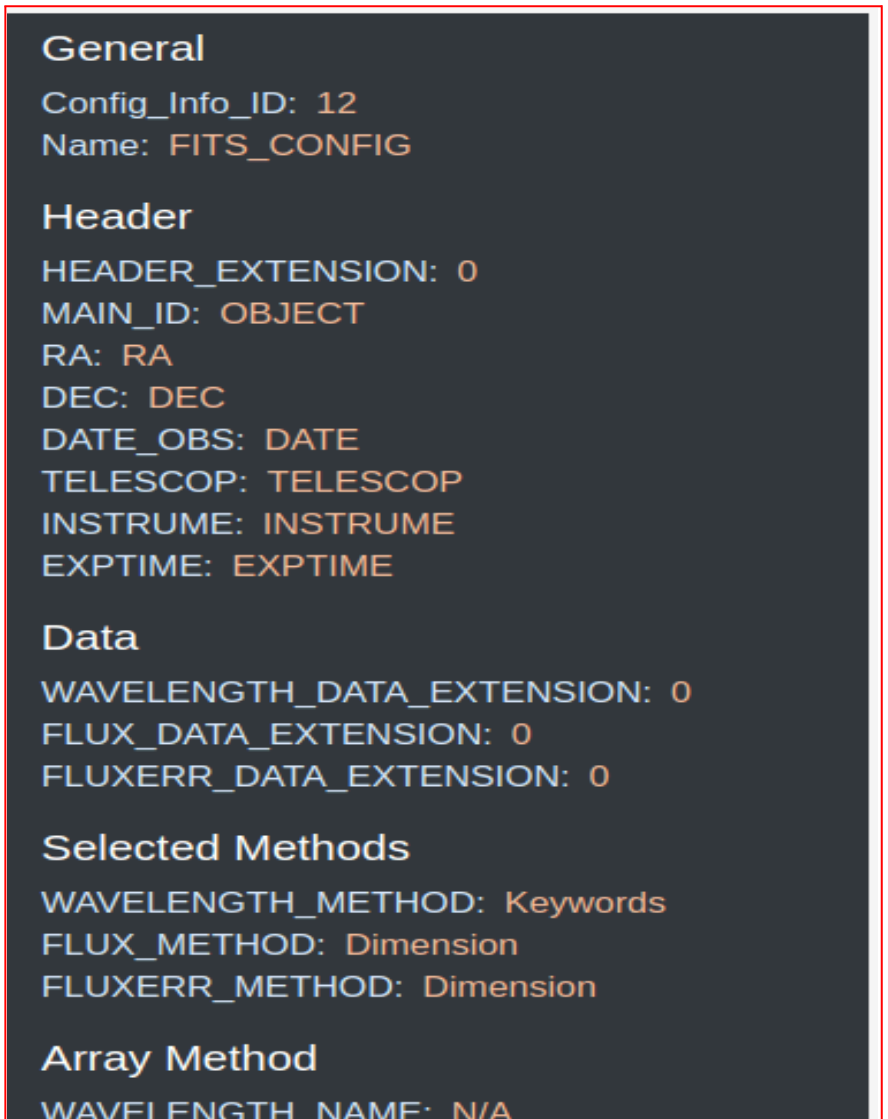
**Example 4**


General
Config_Info_ID: 10
Name: FITS_CONFIG
Header
HEADER_EXTENSION: 0
MAIN_ID: OBJECT
RA: RA
DEC: DEC
DATE_OBS: DATE-OBS
TELESCOP: TELESCOP
INSTRUME: INSTRUME
EXPTIME: EXPTIME
Data
WAVELENGTH_DATA_EXTENSION: 0
FLUX_DATA_EXTENSION: 0
FLUXERR_DATA_EXTENSION: 0
Selected Methods
WAVELENGTH_METHOD: Keywords
FLUX_METHOD: Dimension

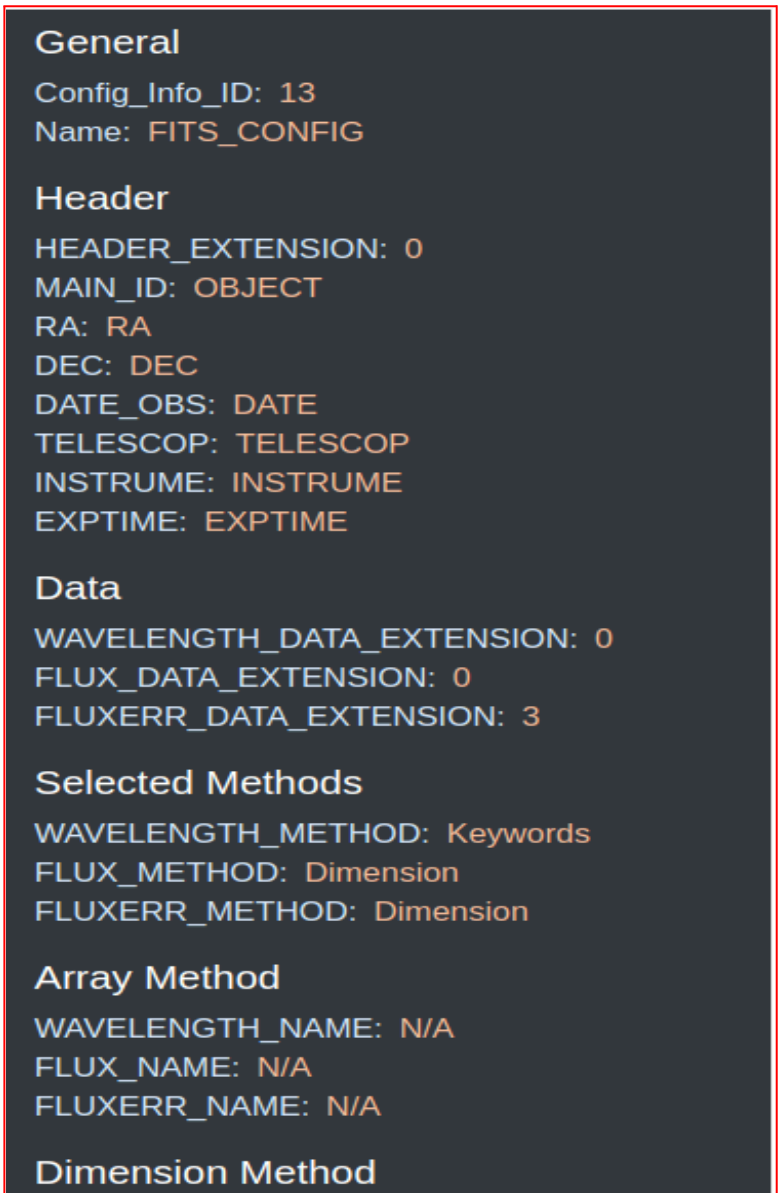
**Example 5**


General
Config_Info_ID: 11
Name: FITS_CONFIG
Header
HEADER_EXTENSION: 0
MAIN_ID: OBJECT
RA: RA
DEC: DEC
DATE_OBS: DATE
TELESCOP: TELESCOP
INSTRUME: INSTRUME
EXPTIME: EXPTIME
Data
WAVELENGTH_DATA_EXTENSION: 0
FLUX_DATA_EXTENSION: 0
FLUXERR_DATA_EXTENSION: 1
Selected Methods
WAVELENGTH_METHOD: Keywords
FLUX_METHOD: Dimension
FLUXERR_METHOD: Dimension
Array Method
WAVELENGTH_NAME: WAVE

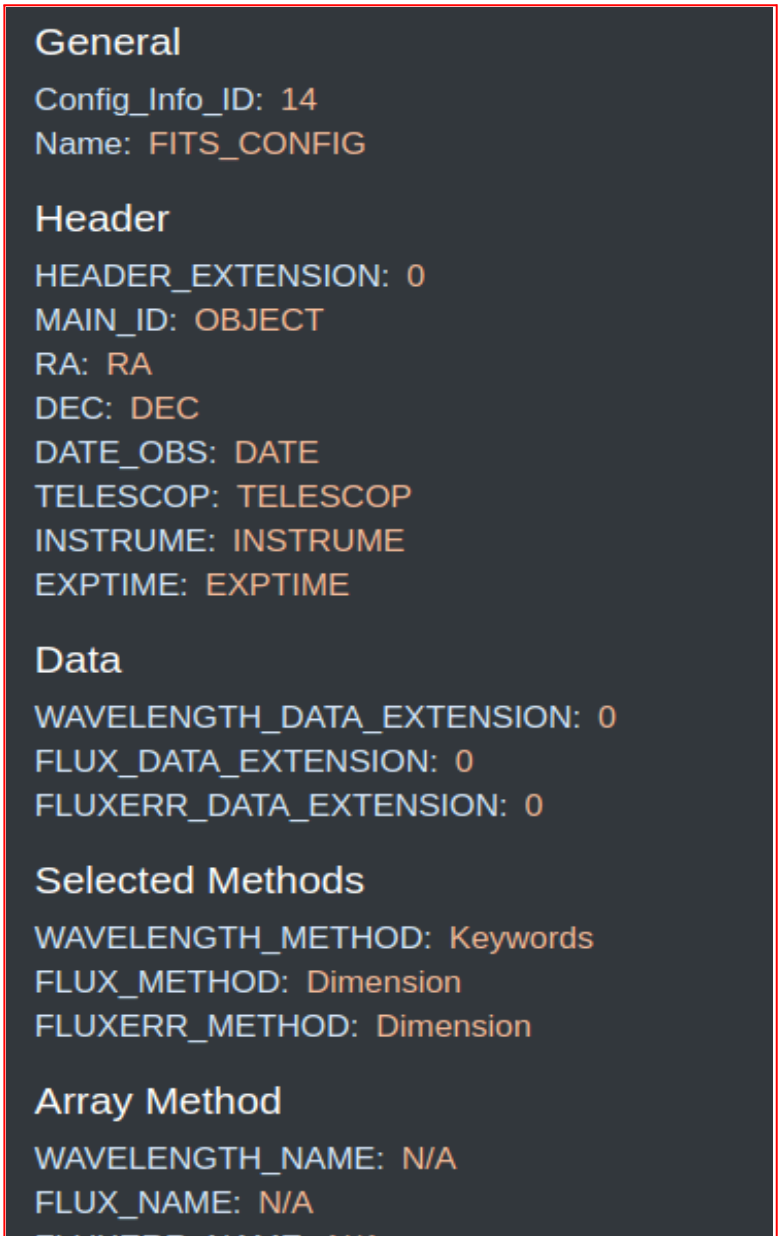
**Example 6**


General
Config_Info_ID: 12
Name: FITS_CONFIG
Header
HEADER_EXTENSION: 0
MAIN_ID: OBJECT
RA: RA
DEC: DEC
DATE_OBS: DATE
TELESCOP: TELESCOP
INSTRUME: INSTRUME
EXPTIME: EXPTIME
Data
WAVELENGTH_DATA_EXTENSION: 0
FLUX_DATA_EXTENSION: 0
FLUXERR_DATA_EXTENSION: 0
Selected Methods
WAVELENGTH_METHOD: Keywords
FLUX_METHOD: Dimension
FLUXERR_METHOD: Dimension
Array Method
WAVELENGTH_NAME: N/A

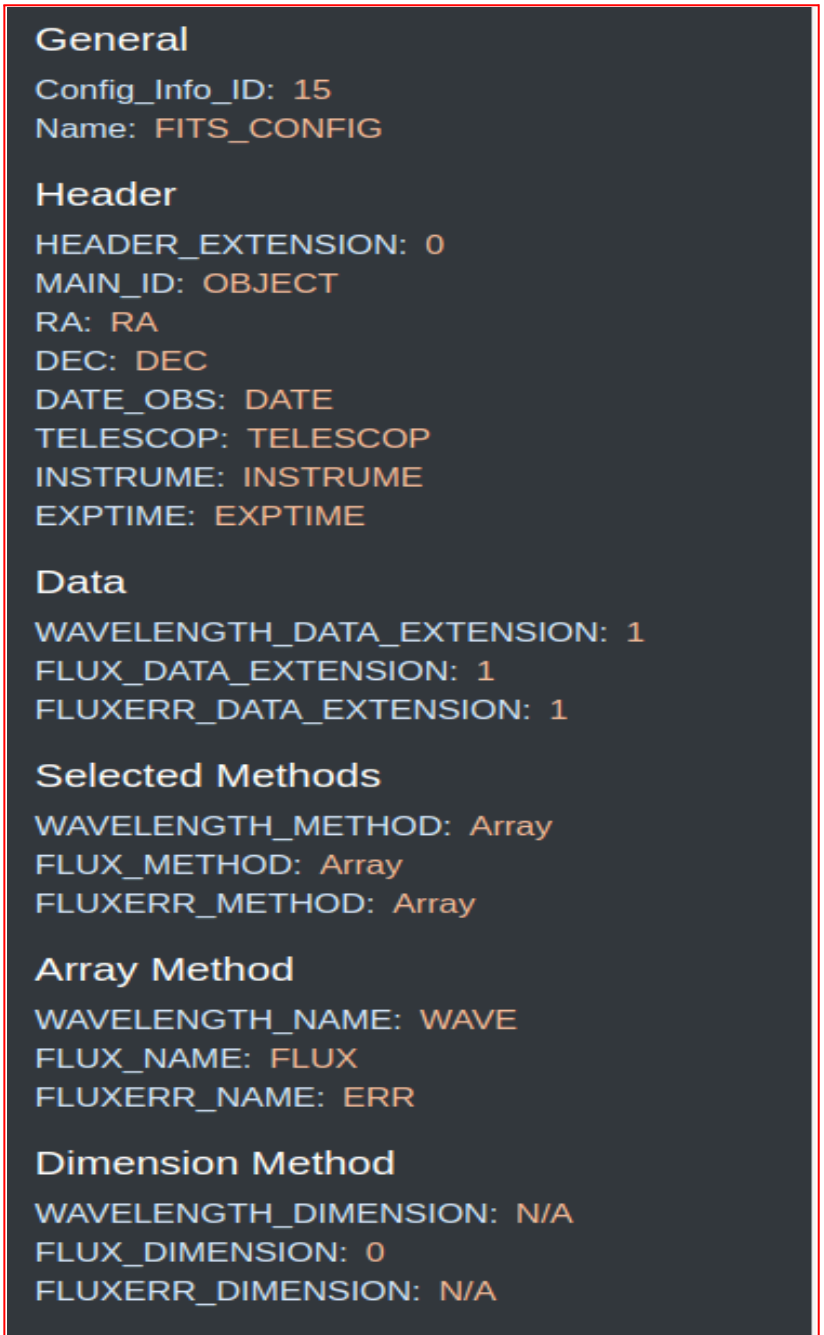
**Example 7**


General
Config_Info_ID: 13
Name: FITS_CONFIG
Header
HEADER_EXTENSION: 0
MAIN_ID: OBJECT
RA: RA
DEC: DEC
DATE_OBS: DATE
TELESCOP: TELESCOP
INSTRUME: INSTRUME
EXPTIME: EXPTIME
Data
WAVELENGTH_DATA_EXTENSION: 0
FLUX_DATA_EXTENSION: 0
FLUXERR_DATA_EXTENSION: 3
Selected Methods
WAVELENGTH_METHOD: Keywords
FLUX_METHOD: Dimension
FLUXERR_METHOD: Dimension
Array Method
WAVELENGTH_NAME: N/A
FLUX_NAME: N/A
FLUXERR_NAME: N/A
Dimension Method

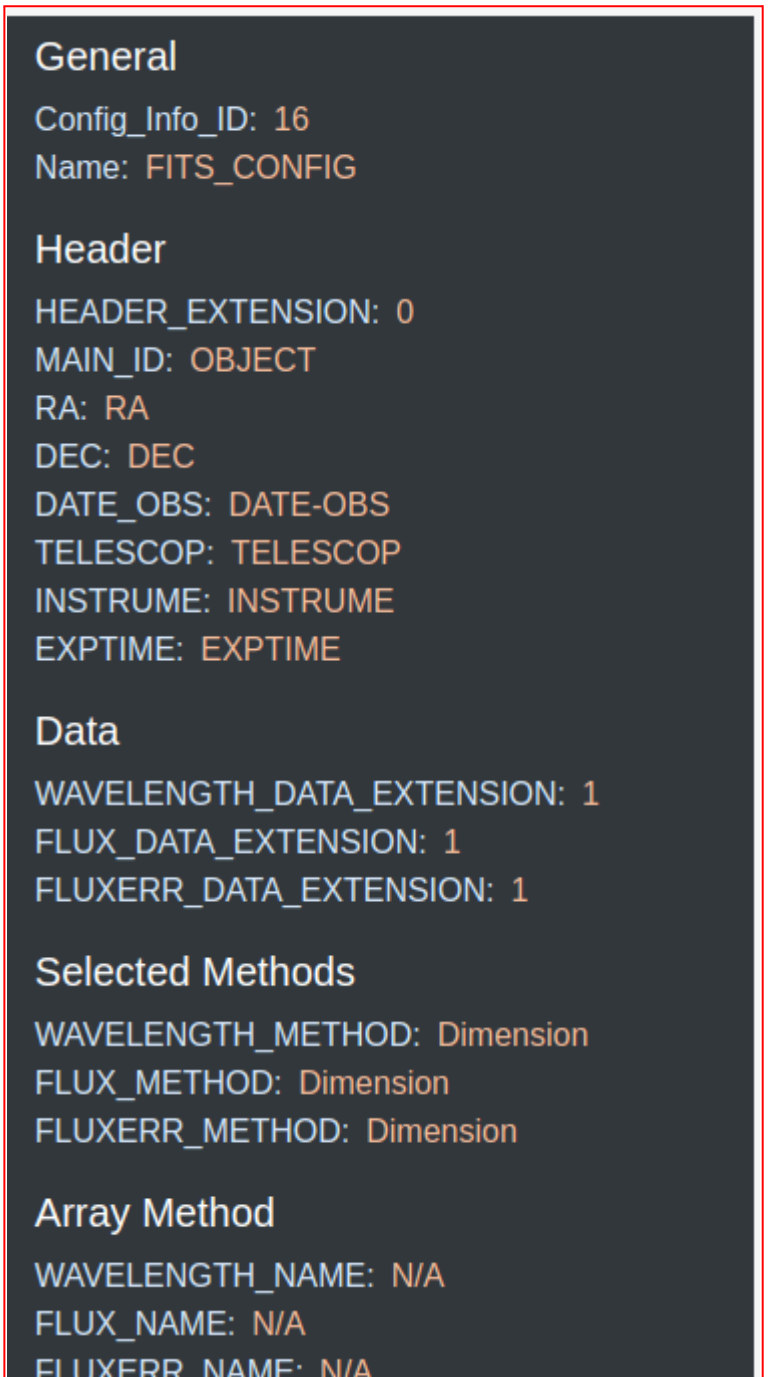
**Example 8**


General
Config_Info_ID: 14
Name: FITS_CONFIG
Header
HEADER_EXTENSION: 0
MAIN_ID: OBJECT
RA: RA
DEC: DEC
DATE_OBS: DATE
TELESCOP: TELESCOP
INSTRUME: INSTRUME
EXPTIME: EXPTIME
Data
WAVELENGTH_DATA_EXTENSION: 0
FLUX_DATA_EXTENSION: 0
FLUXERR_DATA_EXTENSION: 0
Selected Methods
WAVELENGTH_METHOD: Keywords
FLUX_METHOD: Dimension
FLUXERR_METHOD: Dimension
Array Method
WAVELENGTH_NAME: N/A
FLUX_NAME: N/A

**Example 9**


General
Config_Info_ID: 15
Name: FITS_CONFIG
Header
HEADER_EXTENSION: 0
MAIN_ID: OBJECT
RA: RA
DEC: DEC
DATE_OBS: DATE
TELESCOP: TELESCOP
INSTRUME: INSTRUME
EXPTIME: EXPTIME
Data
WAVELENGTH_DATA_EXTENSION: 1
FLUX_DATA_EXTENSION: 1
FLUXERR_DATA_EXTENSION: 1
Selected Methods
WAVELENGTH_METHOD: Array
FLUX_METHOD: Array
FLUXERR_METHOD: Array
Array Method
WAVELENGTH_NAME: WAVE
FLUX_NAME: FLUX
FLUXERR_NAME: ERR
Dimension Method
WAVELENGTH_DIMENSION: N/A
FLUX_DIMENSION: 0
FLUXERR_DIMENSION: N/A

**Example 10**


General
Config_Info_ID: 16
Name: FITS_CONFIG
Header
HEADER_EXTENSION: 0
MAIN_ID: OBJECT
RA: RA
DEC: DEC
DATE_OBS: DATE-OBS
TELESCOP: TELESCOP
INSTRUME: INSTRUME
EXPTIME: EXPTIME
Data
WAVELENGTH_DATA_EXTENSION: 1
FLUX_DATA_EXTENSION: 1
FLUXERR_DATA_EXTENSION: 1
Selected Methods
WAVELENGTH_METHOD: Dimension
FLUX_METHOD: Dimension
FLUXERR_METHOD: Dimension
Array Method
WAVELENGTH_NAME: N/A
FLUX_NAME: N/A