ResultsFileManagement

De Wiki
Aller à : navigation, rechercher
ResultsFileManagement

GENIUS proposes different management for data files. We may identify two different type of file:

- "**Madona**" files
- SQLite files

Sommaire

Madona files

This format is more or less derived from the old one used with previous Fortran tools. Thus, its format is a bit different but the principles remains the same. Here is an example of such file with its header then the data ranked by colums:

```
#<AM-acces:COL-V2.0>
<INIT:
 Logiciel="DOORS"
 VERSION="V10.0"
<COL:
  0 : DATE ~cal (Date)
  1 : TIME ~s (Time)
  2 : X ~m (Integration Frame X Coordinates)
  3 : Y ~m (Integration Frame Y Coordinates)
  4 : Z ~m (Integration Frame Z Coordinates)
  5 : VX ~m/s (Integration Frame X Velocity Coordinates)
  6 : VY ~m/s (Integration Frame Y Velocity Coordinates)
  7 : VZ ~m/s (Integration Frame Z Velocity Coordinates)
  8 : PSI ~deg (Integration Frame Pitch Angle)
  9 : TETA ~deg (Integration Frame Yaw Angle)
  10 : PHI ~deg (Integration Frame Roll Angle)
```

```
>
>
2010-01-01T00:00:00.000 0.000000000000000e+00  6.678133638255782e+06
3.787739409565575e-01 -6.700773491792874e+03 -6.052524398242765e+00
4.780628779989441e+03 -6.031804544978191e+03  9.000000000000000e+01
5.160000022816957e+01  2.699999999999999e+02
2010-01-01T00:00:10.000 1.000000000000000e+01  6.677625571129634e+06
4.780559859589861e+04 -6.701701843926338e+04 -9.555978350214156e+01
4.780308392028196e+03 -6.031309305094114e+03  9.106301358852552e+01
5.159519405116432e+01  2.708330583318874e+02
...
```

Note that to be interpreted, dates must have the "yyyy-mm-ddThh:mm:ss.sss" format (and the "unit" must be ~cal).

Since V1.10, it is possible to take into account a "plot break" if the difference between two consecutive points are greater than a threshold given int the header as it:

```
  8 : PSI ~deg (Integration Frame Pitch Angle) 180.
  9 : TETA ~deg (Integration Frame Yaw Angle) 90.
  10 : PHI ~deg (Integration Frame Roll Angle) 180.
```

Of course, **GENIUS** includes classes and methods to read or write such file.

How to read it

Here is an example for reading it:

```
// Opening it and load data
final GPlotDataMadonaReader fileData = new GPlotDataMadonaReader();
fileData.load(new File("EPHEM.txt"));

// Recovery of the colums information
final int nbColumnsMadona = fileData.getNumberColums(null);

// Loop on the columns
for (int i = 0; i < nbColumnsMadona; i++) {

    final GPlotColumnInfo colInfo = fileData.getColumnInfo(null, i);
    System.out.print("Index: " + colInfo.getIndex());
    System.out.print(" / Name: " + colInfo.getName());
    System.out.print(" / Unit: " + colInfo.getUnitName());
    System.out.println(" / Description: " + colInfo.getDesc());

}

// We recover only colums 2, 12, 13 & 14
final List<Double[]> dataEphem = fileData.getColumns(null, new Integer[] { 1,
12, 13, 14 });
```

```
final int nbLines = dataEphem.get(0).length;
System.out.println("Amount of lines of data: " + nbLines);

System.out.println(String.format("%s %10e %10e %10e %10e", "First line: ",
            dataEphem.get(0)[0],
            dataEphem.get(1)[0],
            dataEphem.get(2)[0],
            dataEphem.get(3)[0]));
System.out.println(String.format("%s %10e %10e %10e %10e", "Last line: ",
            dataEphem.get(0)[nbLines-1],
            dataEphem.get(1)[nbLines-1],
            dataEphem.get(2)[nbLines-1],
            dataEphem.get(3)[nbLines-1]));
```

How to write it

To write such files, we have to respect the following steps:

Header information

```
// Header information
ArrayList<String> headerInfoLines = new ArrayList<String>();
headerInfoLines.add("Logiciel=\"TEST\"");
headerInfoLines.add("VERSION=\"Vx.x\"");
```

Link with a file

```
// Initialization
final MadonaWriter madonaWriter = new MadonaWriter(headerInfoLines);
madonaWriter.createFile(new File(EPH_FILE));
```

Column information

```
// Column information
final ArrayList<ColumnInfo> columnInfoList =  new ArrayList<ColumnInfo>();
ColumnInfo infoDate = new ColumnInfo("DATE", "Date", ColumnType.DATE, "cal",
null, true);
ColumnInfo infoAlta  = new ColumnInfo("ALTITUDE", "Altitude",
ColumnType.REAL, "km", null, true);
...
columnInfoList.add(infoDate);
columnInfoList.add(infoAlt);
...
```

Storing data in lists

```
// Storing data in lists
final ArrayList<Object> dateValues = new ArrayList<Object>();
final ArrayList<Object> altValues  = new ArrayList<Object>();

for (int i = 0; i < array.length; i++) {
    dateValues.add(xtab[i]);
    altValues.add(ytab[i]);
}
```

Adding colums

```
// Adding columns
madonaWriter.addColumns(infoDate, dateValues, 0);
madonaWriter.addColumns(infoAlt, altValues, 1);
...
```

Storing data in file

```
// Storing data in file
madonaWriter.writeHeader(columnInfoList);;
madonaWriter.writeColumns();
madonaWriter.close();
```

SqLite files

Since V1.7, **GENIUS** proposes also to create data file with the SQLite format. The following example shows how to create such data base.

Note that these kind of file may be read by a lot of free tool to be downloaded. Anyway, **GENIUS** proposes a specific widget able to plot data contained in such file (see here).

First, the following lines of code correspond to the initialization of data to be stored int the SQLite file. In fact, we will store two tables:

- an "**ephemeris**" table
- an "**event**" table

Each table will contain three colums of data (**time**, **altitude** and **longitude**). "**ephemeris**" table will have 5 points as "**event**" table will only get two (the first and the last point of the "**ephemeris**" table).

```
// Data initialization

final String NAME_EPHEM_TABLE = "ephemTable";
final String NAME_EVENT_TABLE = "eventTable";

final int nbColumnsSql = 3;
```

```
final String[] dataNames = { "TIME", "LATG", "LONG" };
final Double[]  gapThresholds = { 1.e+99, Math.PI, Math.PI };
final String[]  unitNames = { "s", "deg", "deg" };
final String[]  description = { "Time", "Latitude", "Longitude" };

final int nbPtsEphem = 5;
final double[] timeTable = { 0., 10., 20., 30., 40. };
final double[] latgTable = { -0.5*Math.PI, -0.25*Math.PI, 0.,  0.25*Math.PI,
0.5*Math.PI };
final double[] longTable = { -Math.PI, -0.5*Math.PI, 0.,  0.5*Math.PI,
Math.PI};

final int nbPtsEvent = 2;
 final double[] timeTableEvent = { 0., 40. };
final double[] latgTableEvent = { -0.5*Math.PI, 0.5*Math.PI };
final double[] longTableEvent = { -Math.PI, Math.PI};
```

Then the lines below, will show how to create, fill then close the file.

First, we will open the file ...

```
// File creation and reset if it already exists

final File sqliteFile = new File("EPHEM.db");
if ( sqliteFile.exists() && !sqliteFile.delete() ) {
   System.out.println("Sqlite output file failed to delete: %s");
}

final ResultWriter  resultWriter = new ResultWriter(sqliteFile);
resultWriter.open();
```

Then, we will define two tables giving their name and description ...

```
// Table configuration
resultWriter.addTable (NAME_EPHEM_TABLE);
resultWriter.addTable (NAME_EVENT_TABLE);

// Columns configuration
/// Note:
// gapThreshold is the value given for plot discontinuities
// userVisible may be used to add columns but not visible via the GUI (for
example for plotting).
for (int i = 0; i < nbColumnsSql; i++) {
    resultWriter.addColumn(NAME_EPHEM_TABLE, dataNames[i], description[i],
ColumnType.REAL, unitNames[i], gapThresholds[i], true);
    resultWriter.addColumn(NAME_EVENT_TABLE, dataNames[i], description[i],
ColumnType.REAL, unitNames[i], gapThresholds[i], true);
}
```

At last, we will add the data inside the table then close the file.

```
// Adding values for each line
// Note: each table is independant and one may fill them asynchroneously.
for (int i = 0; i < nbPtsEphem; i++) {
    // Note: it is mandatory to add a value for each column. If it is not the
case, it will raise an error.
    // On the other hand, if we add two times a value for the same
line/column, the last value will be taken into account.
    resultWriter.addValue(NAME_EPHEM_TABLE, dataNames[0], timeTable[i]);
    resultWriter.addValue(NAME_EPHEM_TABLE, dataNames[1], latgTable[i]);
    resultWriter.addValue(NAME_EPHEM_TABLE, dataNames[2], longTable[i]);
    // Line is stored in the database
    resultWriter.writeLine(NAME_EPHEM_TABLE);
}

// Same for events
for (int i = 0; i < nbPtsEvent; i++) {
    resultWriter.addValue(NAME_EVENT_TABLE, dataNames[0], timeTableEvent[i]);
    resultWriter.addValue(NAME_EVENT_TABLE, dataNames[1], latgTableEvent[i]);
    resultWriter.addValue(NAME_EVENT_TABLE, dataNames[2], longTableEvent[i]);
    resultWriter.writeLine(NAME_EVENT_TABLE);
}

// Closing the data base
resultWriter.close();
```


Menu de navigation

Affichages

- Lire
- Voir le texte source
- Historique
- Exporter en PDF

Plus

Rechercher

Rechercher Lire

GENIUS

- Welcome
- Quick Start
- News

Basic principles

- GFrame and GPanel
- Main widgets
- Links with Swing
- GLayout
- Conditional Display
- GListener interface

More deeper in the concept

- Units management
- GContainer
- GReadWrite interface and data files management
- Modified data
- Process management

Still more ...

- Validity controls
- Menu bar
- Icons
- GClear interface

Still more again ...

- Tooltips
- Shortcuts
- Copy & paste
- Plots
- Results File Management
- GPlotPanel
- GGroundPlotPanel
- Internationalization
- Log file
- Update data

Some other widgets

- GTabbedPane
- GTable1D
- GTable2D
- GComponentList
- GDialog and GDetachedPanel
- GContextFileManagement
- How to build a standard application
- GPanTest
- Create your own widget

Evolutions

- Main differences between V1.11.4 and V1.12.1
- Main differences between V1.10.1 and V1.11.4
- Main differences between V1.10 and V1.10.1
- Main differences between V1.9.1 and V1.10
- Main differences between V1.9 and V1.9.1
- Main differences between V1.8 and V1.9
- Main differences between V1.7 and V1.8
- Main differences between V1.6.2 and V1.7
- Main differences between V1.6.1 and V1.6.2
- Main differences between V1.6 and V1.6.1
- Main differences between V1.5 and V1.6
- Main differences between V1.4.1 and V1.5
- Main differences between V1.3 and V1.4.1

Training

- Training slides
- Tutorials package for V1.12.1
- Tutorials package for V1.11.4
- Tutorials package for V1.10.1
- Tutorials package for V1.9.1

- Training & tutorials package for V1.8
- Training & tutorials package for V1.7
- Training & tutorials package for V1.6

Links

- CNES freeware server

Outils

- Pages liées
- Suivi des pages liées
- Pages spéciales
- Adresse de cette version
- Information sur la page
- Citer cette page

- Dernière modification de cette page le 17 décembre 2020 à 17:09.

- Politique de confidentialité
- À propos de Wiki
- Avertissements
- Powered By MediaWiki