Data containers. Table and Layer Classes

© 2013 gvSIG Association

© 2013 gvSIG Association
This document is distributed under the Creative Commons

Content

© 2013 gvSIG Association

1. The "containers" of data. Objects Layer and Table

There are two objects that have the common feature that store data, even though they have great differences. These objects are, the *table* (corresponding to the object java TableDocument) and *Layer* (corresponds to the Java object FLAYER). It should be noted that although the methods shared data management are different objects, so that the other properties are not the same. Let's see an example. We charge a table from document from gvSIG project manager, to correspond with a shape file dbf. Then open this shape in a view, and open its attribute table. In the project manager if you check the documents table lists, two documents appear, whose names and whose attributes do not match (the document that we have opened as a document Table doesn't have GEOMETRY field), although they are the same file dbf.

Opening each of the items from the Jython console we can see that are different:

```
Jython Completion Shell
Jython 2.5.2 (Release_2_5_2:7206, Mar 2 2011, 23:12:06)
[Java HotSpot(TM) Server VM (Sun Microsystems Inc.)] on java1.6.0_26
>>> from gvsig import *
>>> currentTable().getName()
u'areaDeInfluencia'
>>> currentTable().getSchema().getAttrNames()
[u'Id_muni', u'Influencia']
>>> currentTable().getName()
u'Tabla de atributos: areaDeInfluencia.shp'
>>> currentTable().getSchema().getAttrNames()
[u'Id_muni', u'Influencia', u'GEOMETRY']
>>>
```

As mentioned, both the *layer as* the *Table* contains a set of data or records, the difference, about their data, is that there is layer field "geometry", which is the definition of the element mapping, and the table does not, so that methods of managing their data collections are the same.

These methods are:

```
- features([expresion][, sortBy][, asc])
- edit()
- append( values )
- updateSchema( schema )
- getSchema()
- commit()
- abort()
- getSelection()
```

© 2013 gvSIG Association

- features ([expression] [, sortBy] [, asc]): Returns the data collection.
    - expression, string (optional): A condition that the phenomenon must meet in order to be included in the collection that is returned.
    - sortBy, string (optional) field identifier to be used for ordering the collection of phenomena that is returned.
    - asc, boolean (optional): True if ordered in ascending order, otherwise False. True is set up as default.

  The parameter string expression defines a filter that is evaluated to determine if a phenomenon should be included in the collection that is returned. Eg:

```
expression = "ID > 10 AND ID < 20"
features = currentLayer().features(expresion, 'ID', True)
```

  This code returns the collection of phenomena of the active layer whose field ID is greater than 10 and less than 20, ordered by ascending ID field. If you use an expression to filter the events and this expression is invalid, it will return None. If the expression is valid but does not produce results, it will return a set of 0 elements.

  If you just want to sort the collection by a field downstream, the code would be:

```
features = currentLayer().features(sortBy='ID', asc=True)
```

- edit (): Enables the object editing. It is necessary that the object is in edit, bothto update and to add new objects to the collection of data.

  Once you do not need to make changes will be necessary to invoke the commit method if you want to persist the changes or abort if it is to be saved.
- append (values): Creates a new phenomenon and adds it to the collection of data.
    - values, dict: Adds the key property of the phenomenon, the corresponding value.

  If the object is not in the edit state, when using this method will changes the state into the edit mode.
- updateSchema (schema): Updates the object data model with the new model that is passed as a parameter. The object must be in edit mode.
    - schema, Schema: Data model that will use the table or layer
- getSchema (): Returns the data model of the object.
- Commit (): Ends the edit mode keeping any changes
- abort (): Finish the edit mode discarding any changes.
- getSelection (): Returns the subset of events that are selected in the order of all elements in the collection. If no items are selected will return a *Selection* object with 0 events.
- select (selection): Adds a phenomenon or a set of them to the selection.
    - selection, Feature, FeatureSet: Parameter selection can be a phenomenon (Feature) or set of phenomena (FeatureSet)

© 2013 gvSIG Association

- deselected(feature): Removes a phenomenon or a set of them to the selection.
    - selection, Feature, FeatureSet: Selection Parameter witch can be a phenomenon (Feature) or a set of phenomena (FeatureSet)
- isSelected (feature): Returns True if the phenomenon is selected.
    - feature, Feature: Phenomenon we want to know if it is selected.

      Other common methods are:

```
- getProjectionCode()
- getName()
```

- getProjectionCode (): Returns a string with the code of the projection of the layer/table
- getName (): Returns the name of the layer in the Table of Contents view (TOC, in English, Table Of Contents)

Exclusive Methods of Layer Object

```
- getTypeVectorLayer()
```

- getTypeVectorLayer (): Returns the geometry type of the layer.

Exclusive Methods of Layer Object

```
- getAssociatedLayer()
```

- getAssociatedLayer (): Returns the associated object Layer attribute table or None if none exists.

© 2013 gvSIG Association

gvSIG Association

Plaza Don Juan de Villarrasa 14-5,

46001, Valencia (Spain)

Registro Nacional de Asociaciones (National Register of Associations): 596206

e-mail : info@gvsig.com

Web: www.gvsig.com

Project website: http://www.gvsig.org

Documentation made by Víctor Acevedo. Translated by Elisabet Adeva

Distribution Lists

There are three mailing lists in order to facilitate communication between all stakeholders in gvSIG project. The first two are for users and developers, mainly oriented to the Hispanic community, with the use of the Castilian lenguage preferred. The third list, is a international list, aimed at other communities where is preferd the use of English.

- **Members List.** . Here you can leave your opinions on the operation of the software: what things would you like to be developed, doubts related to the use of gvSIG and anything you believe should be in discussion and accommodates in the list of users. The link to subscribe to the user list is the following:
  http://listserv.gva.es/mailman/listinfo/gvsig_usuarios

- **Developers list.** It is geared/oriented to anyone interested in knowing how gvSIG is developed. The link to subscribe to this list is:
  http://listserv.gva.es/mailman/listinfo/gvsig_desarrolladores

- **International list.** It is intended for both users and developers witch do not speak Spanish. The preferably language used is English. The link to subscribe to this list is:
  http://listserv.gva.es/mailman/listinfo/gvsig_internacional

All names of programs, operating systems, computer hardware etc., that appear in this course are trademarks of their respective companies or organizations.

© 2013 gvSIG Association

This manual is distributed under the Creative Commons Attribution-ShareAlike 3.0 Unported *(http://creativecommons.org/licenses/cc-by-sa/3.0/deed.es) - See conditions in Annexes*