PyGrace Tutorial

Introduction

PyGrace is a Python package used for creating Grace files. Grace is a simple 2-D plotting program. PyGrace is intended to:

- enable Grace (`.agr` format) file creation within Python.
- assist in the automated creation of many figures for use in large-scale, exploratory data analysis.
- encourage code reuse, by offering an object-oriented structure.

Visualizing data is often an integral part of the data analysis process, however, **PyGrace is explicity intended not to perform any type of data analysis**. Not even computing an average. Especially not performing a linear regression. This principle underlies much of the structure of PyGrace, with the purpose of not allowing users to adopt a "black box" approach to analysis. While visualizing data should be as painless as possible, the pain required to understand the analysis process is considered necessary, or even desirable (if you're into that sort of thing).

This tutorial is meant to help with installation of PyGrace, and give new users an idea of how PyGrace is structured. After reading this brief document, the next step for learning to use PyGrace is to go through the examples that are located in the `PyGrace/Examples` directory.

Installation

As PyGrace is still young, the only method yet available to install it is to download the source code from:

`http://sourceforge.net/projects/pygrace`

The subsection titled *Installing a development version* describes how to download the source code.

Installing a development version

To check out the PyGrace repository, change to a directory where you want to put PyGrace, and type:

```
svn co https://pygrace.svn.sourceforge.net/svnroot/pygrace/trunk PyGrace
```

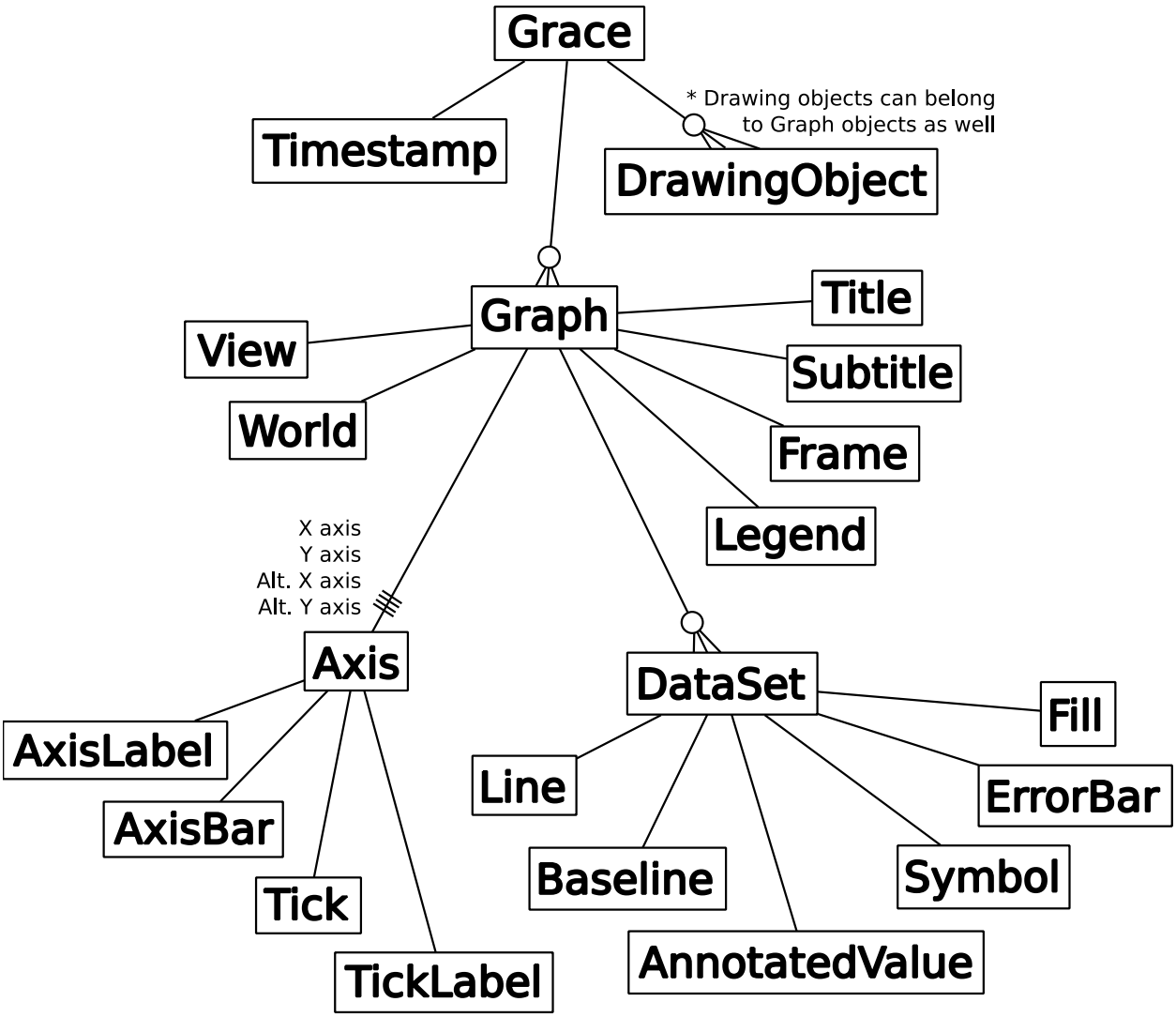

Figure 1: The inheritance structure of PyGrace mirrors the file structure of Grace. This diagram uses "crow's foot" notation to indicate the relationship between different entities.

For now, it is important to use the "correct" capitalization of PyGrace (capital P and G). Then, make sure that the directory you are in is on the python path (the `PYTHONPATH` environment variable). One convenient way to do this is to add a line in your `~/.bashrc` file that says:

```
export PYTHONPATH:$PYTHONPATH:/path/to/your/pybrary/
```

Notice that, unlike the lame-ass author of this documentation, it is not necessary to call the directory that PyGrace is located "the pybrary." After the location of PyGrace is on the `PYTHONPATH`, import statement in python scripts will look in that directory (`.../pybrary/`) for modules, and since PyGrace is a package (it has an `__init__.py` file in the directory), the directory act like a module. Now, make sure that you have executed the `~/.bashrc` file by running

```
source ~/.bashrc
```

That should do the trick. You can test that python can find the PyGrace package by opening an interactive python prompt and typing

```
import PyGrace
```

If no error is raised, then installation was succesful!

Data model

PyGrace was designed so that, "anything you can do in Grace, you can do in PyGrace." Figure **??** is a diagram showing the basic organization of the PyGrace structure — it basically mirrors the Grace file format structure. One thing to note is that Drawing Objects can be associated with either a Grace or a Graph. This is important in cases where one want to, e.g., create a custom data symbol and have the symbol show up in the coordinate system of a graph.
For a more detailed diagram, look at the file `PyGrace/Documentation/diagram.pdf`.