Developer Overview

The purpose of this documentation is to give a high-level overview of how IPRStats works. This will discuss the code base in terms of modules and classes. For more detailed information, please consult the API docs.

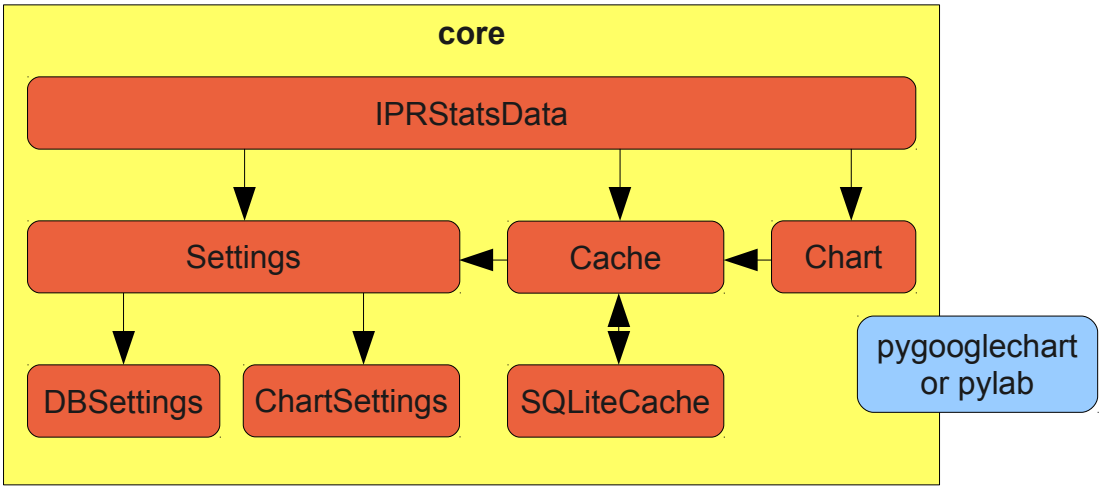
IPRStats was designed to be modular so that it is easy to add and remove different aspects of the application. In particular, the modules illustrate the main parts of the application and their significance. These modules are:

- core
- standalone
- importers
- exporters

While the purpose of these modules may be evident from their name, we'll take a look at the classes contained within each module and their purpose.

Core

The core package contains the essential objects of the program upon which everything else relies. This includes objects to store the program's settings, cache, and shared data.



IPRStatsData

This class functions as the main object of the application, and there should only ever be one instance at a time. It essentially bundles the three necessary classes for other operations:

- The Settings object is necessary for accessing all user defined settings.
- The Cache object is used for storing data extracted from the InterProScan XML file as well as any additional information to be displayed (such as Gene Ontology term names and definitions).
- The Chart object is used for generating any charts needed.

Passing this object to any importers, exporters, or the standalone GUI will give those objects the necessary data to function.

Settings

The Settings object provides a central location for commonly used variables and settings in order to maintain consistency and reduce code duplication. In particular, it provides the following types of information:

- Chart settings read from the user-defined configuration file
- MySQL connection information read from the user-defined configuration file
- Standard directory paths (including the settings, export, and installation directories)
- Other miscellaneous settings (including max table length, GO lookup, and SQLite use)

The Settings object also provides functions to update itself and create additional necessary directories. This includes opening a new session and saving settings when changed by the user.

ChartSettings

The ChartSettings object essentially encapsulates any settings that are needed in order to generate a chart. This makes it easy to pass these settings to a Chart object in order to generate a specific type of chart. These settings include the max number of results, the chart type (pie or bar), and the chart generator (google or pylab).

DBSettings

The DBSettings object encapsulates the settings necessary to establish a connection to a MySQL server. The Settings object has two instances: one for a local MySQL connection (if not using SQLite) and one for a Gene Ontology terms database connection (if GO lookup is enabled).

Cache

The Cache object stores any data retrieved from querying a database and stores it in memory. It stores both match table results and chart data. It also provides methods from retrieving table data given a particular row and chart data with labels. Future development might look towards generalizing this object to handle any query provided to it. This object is intended to be used for creating subclasses that do disk-based storage.

SQLiteCache (Cache)

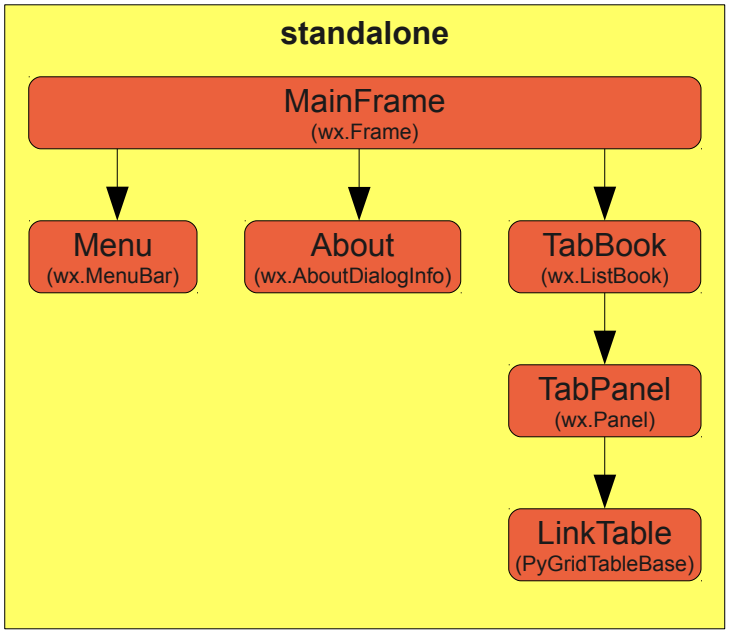
This is a Cache subclass that stores the retrieved data in a SQLite database for later retrieval. Originally there was another subclass that used PyTables, however this was dropped due to the complex dependencies of PyTables and time constraints. A MySQLCache object might also be beneficial in the future.

Chart

The Chart object provides methods to generate and save different types of charts. It uses the common Settings and Cache objects to determine how the chart looks and with what data.

Standalone

This module contains the wxPython subclasses needed to generate the IPRStats GUI. All GUI elements should be contained in this module separate from core functions so that CLI and server versions of IPRStats can be created.



MainFrame *(wx.Frame)*

This class represents the parent window frame upon which other GUI elements are drawn. On initialization, it creates a Menu, an About, and a TabBook object.

Menu *(wx.MenuBar)*

This class defines the options available in the menu bar. Menu items must be bound to a function in initializing the IPRStats object.

About *(wx.AboutDialogInfo)*

Provides information about the program, particularly the name, description, version and developers. In order to support the native Windows and Mac dialogs, the website URL and license text is conditionally excluded.

TabBook *(wx.Listbook)*

This object is responsible for creating and displaying the tabs in the GUI application. It creates a TabPanel object for each InterProScan member database (apps).

TabPanel *(wx.Panel)*

This panel contains the chart and tabular data displayed in the GUI. During initialization, it creates a LinkTable and a wx.StaticBitmap object. These objects are later updated with data.

LinkTable *(wx.grid.PyGridTableBase)*

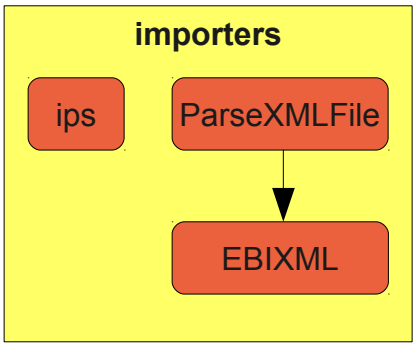
This object defines the data and properties of the table displayed in the GUI. It overrides many of the standard wx.grid.PyGridTableBase methods to efficiently pull information from the Cache object only when the data is being viewed (within the visible scrolling area). It also adds properties such as creating blue underlined text for links. The actual linking behavior is defined in the IPRStats object.

PropertiesDlg *(wx.Dialog)*

Creates a dialog with all the labels, text boxes, spinners, check boxes, tabs, and spacers necessary for a clean layout. It has methods used to populate the inputs with data from the Settings object. It also has a method to save the modified settings to the Settings object and configuration file.

Importers

This module is reserved for objects that will take a given file and convert it to a format that IPRStats can handle. Based on the information specified in the Settings object, objects in this module should either insert data into the specified database or extract it to a session folder.



ParseXMLFile *(Thread)*

This is a thread that when run will create an xml.sax parser using EBIXML as the content handler. It parses the specified file using the values contained in the Settings object.

EBIXML *(ContentHandler)*

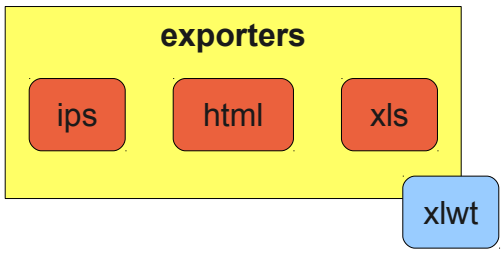
The XML content handler that generates SQL to a specific file. Once the XML file is entirely parsed and the SQL file completely generated, it inserts the extracted data into the database specified in the Settings object.

ips

This importer takes a .ips file (which is really a tar.bz2 file) and extracts it to the sessions directory, returning the new session id.

Exporters

Module intended to hold classes that take the data imported into IPRStats and output it in a particular format. Eventually, we should add a superclass to these classes so that they have a similar, more modular design.



html

This object first copies a 'style.css' file to the chosen directory. Then, with the data contained in the Settings, Cache, and Chart objects, generates an HTML file for each InterProScan member database (app). It contains many helper functions for the individual elements of the HTML files.

xls

This object creates a spreadsheet document that can be opened in Excel or OpenOffice. Using the external package xlwt, it extracts data from the Cache object and writes it to an xls file based on the values in the Settings object. It creates a new worksheet for each InterProScan member database (app).

ips

This is a simple object that compresses the current session folder as a tar.bz2 file. This can be extracted and used by importing with iprstats.importers.ips.

Iprstats

IPRStats

This object ties all the other modules together. During initialization, it creates a Settings & MainFrame object, binds all menu and grid actions to its own functions, and then launches the application. The bound functions perform actions such as launching dialogs, opening web pages, or creating import/export objects to manipulate data.