Binding generator

Here describe the bindings generator

How it works ?

The parser for the C API is in the file source:bindings/bindings.py. It crawls the directory tree starting with source:lasso to find declarations of GObject classes, export public methods of said object, enum and define declarations. For each language the subdirectory bindings/language contain a file named lang.py which contains the specific code to produce the given binding.

The work-flow is usually the same:

- map enums and defines to symbols in the given language
- implement converters between native data structures and data structures from glib like:
  - C strings
  - xmlNode from libxml2
  - GList of string, xmlNode or GObject
  - GHashTable mapping strings to strings or GObject
- associate native language object instance which GObjects, keep a mapping of the correspondance, when an already mapped object is asked return a copy of the native object and increment its reference count (for language using reference counting, for others like Java, do nothing)
- map constructors using the previous mechanism to binding GObject to native language objects
- map methods
- map object fields to getters and setters

What to improve ?

The modularity of thing is a bit lacking, it should use more inheritance and share some code between languages, like the mapping code for GList or GHashTable which could be generated given some base mappings for contained type (string,xmlNode,GObject) and container types (GList and GHashTable).