gsrubylib

```ruby
require 'debuglog' unless $gs_nodebuglog
require 'pry'      unless $gs_nopry
require 'contracts'
include Contracts
```

if object.in? collection	
if object.not_nil?	non_nil?
str = object.pp_s	
o.define_method(:add) do |x,y| x + y end	
squares = (1..10).build_hash { |n| [n, n*n] }	graph
squares.values.mapf(:to_s)	collectf
h = squares.apply_keys { |k| k.to_s }	
h = squares.apply_values { |k| k.to_s }	
"foo".indent(4)	
"bar".tabto(4)	

USAGE = %{
  | usage: prog [-o dir] -h file...
  | where
  | -o dir   outputs to DIR
  | -h   prints this message
}.trim("|")	

StringIO.string { |o| o.puts "Hi…" }	

class Person
  attr_predicate :young
  attr_predicate_rw :successful
end	

Labels

```ruby
Result = GS::Label.create(:win, :lose,
                          :draw)
result = Result.lose
result.to_s / to_sym / symbol / inspect
result == Result[:lose]
```

Labels are safer than symbols because they guard against misspellings. They also "inspect" nicely.

Values

```ruby
Person =
  GS::Value.new(name: String, age: Nat, married: Bool)
        .default(married: false)
        .create

p = Person[name: 'John', age: 25]      or Person.new(…)

p.name; p.age; p.married; p.married?
p[:name]    # etc.

p.with(age: 26, married: true)

p.attributes
p.values
p.values(:name, :married)

e = p.upgrade(Employee, title: 'Nurse', salary: 58400)
p = e.downgrade(Person)

Person.info       # "Person[name: String, …]"
```

Values are read-only structs with Contracts built-in, default values, predicate methods, copy-constructors (*with*), transformers (*upgrade*, *downgrade*).

They combine type safety, state safety and convenience.