

Saphir Gem Developer's Manual



Tables of contents

| | |
|------------------------------------|----------|
| Introduction | 3 |
| License | 3 |
| Basic Network Communication | 4 |

Introduction

I developed Saphir (German for Sapphire) in order to make the introduction to the Ruby language easier. Ruby is a very powerful and intuitive programming language, but it is NOT easy to learn.

Saphir is a collection of useful functions and methods that young developers can implement so that they get faster programming result. This sense of achievement shall encourage them to keep learning Ruby.

Please keep in mind that Saphir is currently in an very early alpha stage and under active development. So if you have any suggestions for function implementation feel free to send me a email to mail@sapphirus.cf.

License

Copyright (c) 2014 Sapphirus (Marvin Peter)

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Basic Network Communication

The Basic Network Communication enables you to simply create a Ruby server and client application that can send strings, files, etc to each other.

These socket communications are encrypted using OpenSSL by default and default certificates are provided within the Ruby Gem.

The Server

The following options are available:

| Option | Default Value |
|-----------|---------------|
| password | nopassword |
| port | 7789 |
| useSSL | TRUE |
| enableLog | TRUE |

Available methods

- initialize(server, password = "nopassword", port=7789, useSSL=true, enableLog=true)
- setSSLCertificate(certFile)
- setSSLKey(keyFile)
- start

A default configured server can be created like this:

```
require "Saphir/BNCServer"

server = Saphir::BNCServer.new()
server.start do |request|
  response = ""
  case request[0..-2]
  when "Hello"
    response = "Hi"
  else
    response = "How can I help you?"
  end
  response
end
```

Or if you prefer a more advanced approach do it e.g. like this:

```
require "Saphir/BNCServer"

puts "SERVER: Loading..."

server = Saphir::BNCServer.new("secre", 5677, true, false)
thread = Thread.start do
  server.start do |request|
    response = ""
    case request[0..-2]
    when hello
      response = "Hi"
    else
      response = "How can I hep you?"
    end
    response
  end
end

puts "SERVER: Done loading!"

begin thread.join end
```

The Client

The following options are available:

| Option | Default Value |
|--------|---------------------------|
| server | <i>is always required</i> |
| port | 7789 |
| useSSL | TRUE |

Available methods

- initialize(server, password = "nopassword", port=7789, useSSL=true)
- setSSLCertificate(certFile)
- sendMessage(msg)
- start
- close

A default configured client can be created like this:

```
require "Saphir/BNCClient"

c = Saphir::BNCClient.new("localhost", "secre")
c.start()
puts "Response received: #{c.sendMessage("HELLO")}"
c.close()
```

Very simple. Isn't it?