

```
1  #!/usr/bin/python3
2  import gi
3  gi.require_version('Gtk','3.0')
4  from gi.repository import Gtk
5  from setting import Setting
6  import subprocess
7
8  class MainWindow(Gtk.Window):
9      def __init__(self):
10         Gtk.Window.__init__(self)
11         self.nginx = 'nginx'
12         self.php = 'php7.2-fpm'
13         self.mysql = 'mysql'
14         self.jsonSetting = Setting().fetchSetting()
15
16         self.importObject()
17         self.initial(self.jsonSetting["nginxPortEntry"])
18         self.statusL()
19         # self.headerLogoStatus()
20         self.switchStatusText()
21         self.switchButton.set_active((self.statusAllProcess()))
22         self.switchButton.connect("notify::active", self.switchToggle)
23         self.settingButton.connect("clicked", self.onSettingClicked)
24         self.mainWin = self.builder.get_object('mainWin')
25         self.mainWin.connect('delete-event', self.quit)
26
27         self.mainWin.show_all()
28
29         Gtk.main()
30
31     def switchStatusText(self):
32         if(self.statusAllProcess()):
33             self.switchStatus.set_text("Server Started")
34             self.markup(self.switchStatus, '<b>Server Started</b>')
35         else:
36             self.switchStatus.set_text("Server Stopped")
37
38
39     def startStop(self):
40         if self.jsonSetting["startWithGemp"]:
41             self.startAllProcess()
42         else:
43             print("Error: StartWithGemp")
44
45     def onSettingClicked(self, widget):
46         self.mainWin.set_opacity(0.5)
47         self.setting = Setting()
48         self.setting.main()
49         print("back")
50         self.backFromSetting()
51
52         # print("Back")
53     def backFromSetting(self):
54         self.mainWin.set_opacity(1)
55         self.mainWin.connect('delete-event', self.quit)
56         if self.reloadAllServices(True):
57             print("All Service Reloaded")
58         else: print("Reloading Error")
59
60     def reloadAllServices(self, widget):
61         return (self.processControl(self.nginx, "restart") and
self.processControl(self.php, "restart") and self.processControl(self.mysql,
```

```

        "restart"))
62
63     def initial(self, jsonSetting):
64         self.osStatus.set_text(self.osStatusCheck())
65         self.uidStatus.set_text(subprocess.getstatusoutput('whoami')[1])
66         self.linkButton.set_label('Open Web')
67         if self.processControl(self.nginx, 'status'):
68             self.linkButton.set_uri("http://localhost:{}".format(jsonSetting))
69         else:
70             self.linkButton.set_uri('#')
71         self.switchButton.set_active(self.statusAllProcess)
72
73     def osStatusCheck(self):
74         return subprocess.getstatusoutput('cat /etc/*-release')[1].split('\n')
75         [3].split('=')[1].replace(' ', '')
76
77     def importObject(self):
78         self.builder = Gtk.Builder()
79         self.builder.add_from_file('gui/main.glade')
80
81         self.logo = self.builder.get_object("logo")
82         self.reloadButton = self.builder.get_object("reloadButton")
83         self.reloadButton.connect("clicked", self.reloadAllServices)
84         self.switchButton = self.builder.get_object('switchButton')
85         self.switchStatus = self.builder.get_object('switchStatus')
86         self.nginxStatusL = self.builder.get_object('nginxStatusL')
87         self.mysqlStatusL = self.builder.get_object('mysqlStatusL')
88         self.phpStatusL = self.builder.get_object('phpStatusL')
89         self.osStatus = self.builder.get_object('osStatus')
90         self.uidStatus = self.builder.get_object('uidStatus')
91         self.ipStatus = self.builder.get_object('ipStatus')
92         self.msgText = self.builder.get_object('msgText')
93         self.msgText.set_no_show_all(True)
94         self.warningBox = self.builder.get_object("warningBox")
95         self.warningBox.set_no_show_all(True)
96         self.linkButton = self.builder.get_object('linkButton')
97         self.settingButton = self.builder.get_object("settingButton")
98         self.nginxdot = self.builder.get_object("nginxdot")
99         self.mysqldot = self.builder.get_object("mysqldot")
100        # self.mysqldot.set_no_show_all(True)
101        self.phpdot = self.builder.get_object("phpdot")
102
103        # self.warningBox = self.builder.get_object('warningBox')
104        # self.warningBox.set_visible(False)
105
106    def installServices(self, service):
107        # if (service == 'php7.1-fpm'):
108        #     status = subprocess.getstatus
109        pass
110
111    def statusAllProcess(self):
112        if ((self.processControl(self.nginx, 'status')) and
113            (self.processControl(self.php, 'status')) and (self.processControl(self.mysql,
114            'status'))):
115            self.logo.set_from_file('gui/colorHeader.png')
116            return True
117        else:
118            self.logo.set_from_file("gui/blackHeader.png")
119            return False

```

```
120     def switchToggle(self, switch, gparam):
121         if switch.get_active():
122             if self.startAllProcess():
123                 self.logo.set_from_file('gui/colorHeader.png')
124                 print("All Process Started")
125             else:
126                 print("Error while Starting All Process")
127         else:
128             if self.stopAllProcess():
129                 print("All process Stopped")
130                 self.logo.set_from_file("gui/blackHeader.png")
131             else:
132                 print("Error while stopping Process")
133         self.switchStatusText()
134         self.statusL()
135         # self.switchButton.set_active(not(self.statusAllProcess()))
136
137     def startAllProcess(self):
138         # if not((subprocess.call(['sudo', 'service', 'nginx', 'start']) and
139         # subprocess.call(['sudo', 'service', 'php7.0-fpm', 'start']) and
140         # subprocess.call(['sudo', 'service', 'mysql', 'start'])))
141         if (self.processControl(self.nginx, 'start') and
142 self.processControl(self.php, 'start') and self.processControl(self.mysql,
143 'start')):
144             return True
145         else:
146             print("StartAllProcess Error")
147             return False
148
149     def stopAllProcess(self):
150         if (self.processControl(self.nginx, 'stop') and
151 self.processControl(self.php, 'stop') and self.processControl(self.mysql,
152 'stop')):
153             return True
154         else:
155             print("stopAllProcess Error")
156             return False
157
158     def statusL(self):
159         if self.processControl(self.nginx, 'status'):
160             self.markup(self.nginxStatusL, '<b>Nginx</b>')
161             self.nginxdot.set_from_file("gui/16green.png")
162         else:
163             self.markup(self.nginxStatusL, 'Nginx')
164             self.nginxdot.set_from_file("gui/16red.png")
165
166         if self.processControl(self.php, 'status'):
167             self.markup(self.phpStatusL, '<b>PHP</b>')
168             self.phpdot.set_from_file("gui/16green.png")
169         else:
170             self.markup(self.phpStatusL, 'PHP')
171             self.phpdot.set_from_file("gui/16red.png")
172
173         if self.processControl(self.mysql, 'status'):
174             self.markup(self.mysqlStatusL, '<b>MySQL</b>')
175             self.mysqlldot.set_from_file("gui/16green.png")
176         else:
177             self.markup(self.mysqlStatusL, 'MySQL')
178             self.mysqlldot.set_from_file("gui/16red.png")
179
180     def markup(self, object, text):
181         object.set_markup(text)
```

```
176
177
178     def processControl(self, service, signal):
179         status = (subprocess.getstatusoutput('sudo systemctl {}
180         {}'.format(signal, service))[0])
181         if (status == 1):
182             print('Process {} is Masked\nUnmasking
183             service...!'.format(service))
184             if(self.processControl(service,'unmask')):
185                 print("unable to Unmask...!")
186             else:
187                 print("Unmasking {} Sucessufull".format(service))
188         elif (status == 0):
189             print("ProcessControl", status, service)
190             return True
191         elif (status in [4,5]):
192             if (self.installServices(service)):
193                 print("{} install Sucessfully".format(service))
194             else:
195                 print("Error in installing...")
196         else:
197             print("Process: {} {}".format(service, status))
198             return False
199
200     def quit(self, signal, s):
201         print("stopWithGemp", self.jsonSetting["stopWithGemp"])
202         if self.jsonSetting["stopWithGemp"]:
203             self.stopAllProcess()
204         else:
205             print("Error: stopWithQuit")
206             print("Quiting")
207             self.mainWin.close()
208             Gtk.main_quit()
```