SPK and Associates

www.spkaa.com
Ph: 888-310-4540

SPK and Associates
900 E Hamilton Ave, Ste.100
Campbell, CA 95008

Creating Google Charts with Python

This document will describe how to create Google Charts using Python. Let's take the following case study: You have an existing script running on a server that periodically pings an internet host to test latency. The script continuously logs to a file. We'd like to take this data, and make it available on the web to view visually.

Prerequisites:
- A web server capable of running Python CGI scripts
- The client web browser must have internet access (to access the Google Charts web service)

Let's have a look at our sample data. Our existing ping script logs a timestamp, and a latency value with single decimal precision:

```
09:30:00        45.8
09:31:00        39.2
09:32:00        55.2
09:33:00        48.1
09:34:00        33.5
09:35:00        70.0
09:36:00        65.5
09:37:00        44.2
09:38:00        49.9
```

Now for our CGI script.

```
#!/usr/bin/python
```

We'll want the OS to be able to execute this script, and this header will tell the OS which interpreter to use.

```
import cgi
import cgitb
cgitb.enable()
```

You can choose to debug your script from the OS command line, but I prefer to debug directly from the browser. Importing the cgi and cgitb modules will allow us to view any python errors as a web page.

```
from GChartWrapper import *
```

The GChartWrapper is a python module that simplifies the formatting of the HTTP request. You can download the GChartWrapper from:

http://code.google.com/p/google-chartwrapper/

```
filein = "test.data"
```

Here we are referencing the ping script that logs its data to a file. See the sample data above.

```
times = []
latency = []
```

Define 2 separate lists. One will keep track of the timestamps, the other will keep track of the ping latency times.

```
thefile = open(filein, "r")
```

Create a file handle. We only need to open the file in read-only mode.

```
while thefile:
  line = thefile.readline()
  if len(line) < 2:
    break
  elements = line.split()
  times.append(elements[0])
  latency.append(elements[1])
thefile.close()
```

Read the file line by line. For each line, we'll append the first token we find to the times[] list, and the 2nd token will be appended to the latency[] list.

```
G = Line( latency, encoding='text' )
```

Instantiate a GChartWrapper object. For this example, we are going to use a simple line chart.

```
G.axes.type('xxyy')
```

We want to have a total of 4 axes labels. We initialize that here.

```
G.axes.label( 0, times[0], times[len(times)-1] )
G.axes.label( 1, 'Time' )
G.axes.label( 3, 'Milliseconds' )
```

For each of the axes labels, we need to define what they are. The label definitions are indexed according to the "xxyy" axes type listed above. So the first x axes label will be 0, the 2nd x axes label is 1, etc.

```
G.axes.range( 2, 0, max(latency) )
G.scale( 0, max(latency) )
```

By default, Google Charts creates a chart range of 0-100. We want the range to be based on our actual latency vaules, so we specify a range of zero to whatever the maximum latency is in our list.

```
G.legend( 'Latency' )
```

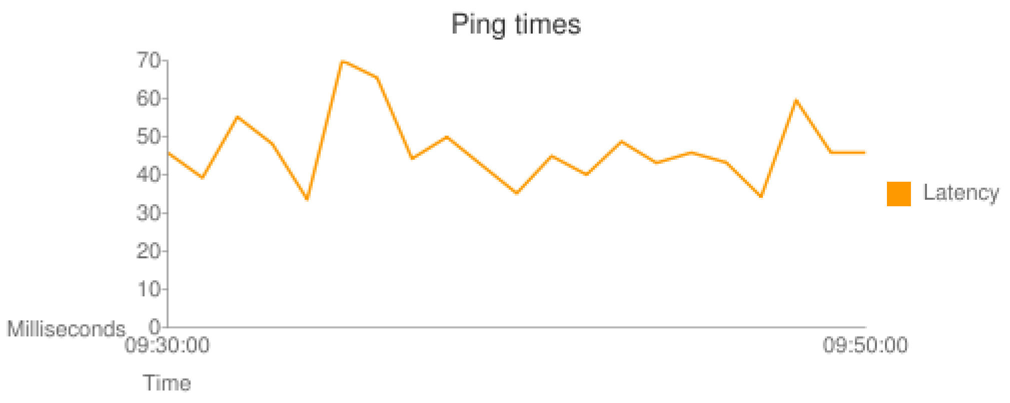
```
G.size ( 500, 200 )
G.title( "Ping times" )
```

Here we define the chart size, the main title, and the legend label.

```
print "Content-Type:  text/html\n\n"
print "<img src=\"" + str(G) + "\">"
```

Finally, we want our script to output the correct Content-Type HTTP header. The GChartWrapper object we created when converted to a string, will be the HTTP request we send to Google. Google returns a PNG image, so we are embedding this in an HTML IMG tag.

Here is the final chart:



And our final code:

```
#!/usr/bin/python

import cgi
import cgitb
cgitb.enable()

from GChartWrapper import *

filein = "test.data"

times = []
latency = []

thefile = open(filein, "r")
while thefile:
  line = thefile.readline()
  if len(line) < 2:
    break
  elements = line.split()
  times.append(elements[0])
  latency.append(elements[1])
thefile.close()

G = Line( latency, encoding='text' )
G.axes.type('xxyy')
G.axes.label( 0, times[0], times[len(times)-1] )
G.axes.label( 1, 'Time' )
G.axes.label( 3, 'Milliseconds' )
G.axes.range( 2, 0, max(latency) )
G.scale( 0, max(latency) )
G.legend( 'Latency' )
G.size ( 500, 200 )
G.title( "Ping times" )

print "Content-Type:  text/html\n\n"
print "<img src=\"" + str(G) + "\">"
```