

POST from a CGI

Source: <http://coding.derkeiler.com/Archive/Python/comp.lang.python/2004-09/4053.html>

From: Michael Foord (fuzzyman_at_gmail.com)

Date: 09/22/04

Date: 22 Sep 2004 01:27:42 -0700

I'm receiving POST data to a CGI, which I'd like to forward to another CGI using urllib2.

I have two options –

1) Parse the data using `cgi.FieldStorage()` and then rebuild the POST request into a dictionary – including any files (? uploading files by urllib2 untested and undocumented – examples seem to be for `httplib`).

2) Read the whole POST data in using `sys.stdin.read()`, rebuild the 'Content-type' and 'Content-length' headers and make the POST.

Obviously (2) is a **lot** less fiddly and less error prone. **But** I'm getting 400 errors when I try it (server thinks request is malformed). I've checked the headers and the body data and they seem normal and I can't work it out.

I wonder if anyone can see what I'm doing wrong.....
(Simple code shown first – then a straightforward example that ought to work and fails).

```
MAXUPLOAD = 10000 # set
max file size of 10 000 bytes
txdata = None
if os.environ.get('REQUEST_METHOD','').lower()=='post': # read
in the POST data from stdin
    txdata = sys.stdin.read(MAXUPLOAD)
    if len(txdata)==MAXUPLOAD and sys.stdin.read(1):
        print overmax
        sys.exit(1) #
print an error message if too big a file is uploaded

envdict = { 'HTTP_CACHE_CONTROL' : 'Cache-control', 'CONTENT_TYPE' :
'Content-type',
            'HTTP_ACCEPT_LANGUAGE' : 'Accept-language', 'HTTP_ACCEPT'
: 'Accept',
            'HTTP_PRAGMA' : 'Pragma', 'HTTP_CONNECTION' :
'Connection',
```

comp.lang.python: POST from a CGI

```
    } #
these are request headers from the browser to the CGI
    # that
we're going to rebuild from the environment variables
defaultuseragent = 'Mozilla/4.0 (compatible; MSIE 5.5; Windows NT)'
    # default User-agent
txheaders = {}
for envvar, header in envdict.items(): # go through
the environment variables putting any headers back
    testval = os.environ.get(envvar)
    if testval:
        txheaders[header] = testval
txheaders['User-agent'] = os.environ.get('HTTP_USER_AGENT',
defaultuseragent) # put in a default user agent if we haven't got
one
if txdata and txheaders.has_key('Content-type'):
txheaders['Content-length'] = str(len(txdata)) # we only need a
'Content-length' header if we have a 'Content-type' one...

theurl = 'http://www.someserver.com/somepath/somepage.html' #
In actual fact I normally decode the url from the PATH_INFO

req = urllib2.Request(theurl, txdata, txheaders) # create the
request object
try:
    u = urllib2.urlopen(req) # fetch a handle on the url !
except IOError, e:
    if not hasattr(e,'code'):
        thecode = 0
    else:
        thecode = e.code
    print 'Content-type: text/html'
    print '<HTML><BODY><H1>Error Code %s</H1></BODY></HTML>' % e.code

info = u.info() # info about the url
pagetype = info.gettype()
print 'Content-type: ' + pagetype + '\n'
print u.read() # print the received page
```

#####

In the code I actually use I make a logfile as well which logs txdata, txheaders and all the environment variables.

Making a simple post to my guestbook (<http://www.voidspace.xennos.com/cgi-bin/guestbook.py>) using this method I get a 400 error.

>From the log I can see the following headers :
(printed using headername, ':', value)

comp.lang.python: POST from a CGI

Connection : keep-alive
Accept-language : en-gb
Pragma : no-cache
Cache-control : max-age=259200
Content-type : application/x-www-form-urlencoded
Content-length : 62
Accept : image/gif, image/x-xbitmap, image/jpeg, image/pjpeg,
application/x-shockwave-flash, application/msword, */*
User-agent : Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)

And the following POST data :

name=Fuzzyman&email=&homepage=&location=&comment=CGI+post+test

Which even has a length of 62.... The only thing I can think of is
that the headers are appearing in the wrong order ?

Can anyone else see what I'm doing wrong ?

Regards,

Fuzzy

<http://www.voidspace.org.uk/atlantibots/pythonutils.html>