

## 10 Tales of Python’s Encoding

by Frederik Braun

Many beginners of Python have suffered at the hand of the almighty `SyntaxError`. One of the less frequently seen, yet still not uncommon instances is something like the following, which appears when Unicode or other non-ASCII characters are used in a Python script.

```
SyntaxError: Non-ASCII character ... in ..., but no encoding declared;
see http://www.python.org/peps/pep-0263.html for details
```

The common solution to this error is to place this magic comment as the first or second line of your Python script. This tells the interpreter that the script is written in UTF8, so that it can properly parse the file.

```
# encoding: utf-8
```

I have stumbled upon the following hack many times, but I have yet to see a complete write-up in our circles. It saddens me that I can’t correctly attribute this trick to a specific neighbor, as I have forgotten who originally introduced me to this hackery. But hackery it is.

### 10.1 The background

Each October, the neighborly FluxFingers team hosts `hack.lu`’s CTF competition in Luxembourg. Just last year, I created a tiny challenge for this CTF that consists of a single file called “packed” which was supposed to contain some juicy data. As with every decent CTF task, it has been written up on a few blogs. To my distress, none of those summaries contains the full solution.

The challenge was in identifying the hidden content of the file, of which there were three. Using the liberal interpretation of the PDF format,<sup>9</sup> one could place a document at the end of a Python script, enclosed in multi-line string quotes.<sup>10</sup>

The Python script itself was surrounded by weird unprintable characters that make rendering in command line tools like `less` or `cat` rather unenjoyable. What most people identified was an encoding hint.

```
0000a0: 0c0c 0c0c 0c0c 0c0c 2364 6973 6162 6c65  ....#disable
0000b0: 642d 656e 636f 6469 6e67 3a09 5f72 6f74  d-encoding:..rot
...
0000180: 5f5f 5f5f 5f5f 5f5f 5f5f 5f5f 5f5f 5f5f 5f5f  _____
0000190: 3133 037c 1716 0803 2010 1403 1e1b 1511  13.|.... .....
```

Despite the unprintables, the long range of underscores didn’t really fend off any serious adventurer. The following content therefore had to be rot13 decoded. The rest of the challenge made up a typical crackme. Hoping that the reader is entertained by a puzzle like this, the remaining parts of that crackme will be left as an exercise.

The real trick was sadly never discovered by any participant of the CTF. The file itself was not a PDF that contained a Python script, but a python script that contained a PDF. The whole file is actually executable with your python interpreter!

Due to this hideous encoding hint, which is better known as a magic comment,<sup>11</sup> the python interpreter will fetch the codec’s name using a quite liberal regex to accept typical editor settings, such as “vim: set fileencoding=foo” or “-\*- coding: foo”. With this codec name, the interpreter will now import a python file with the matching name<sup>12</sup> and use it to modify the existing code on the fly.

<sup>9</sup>As seems to be mentioned in every PoC||GTFO issue, the header doesn’t need to appear exactly at the file’s beginning, but within the first 1,024 bytes.

<sup>10</sup>"""This is a multiline Python string.

It has three quotes."""

<sup>11</sup>See Python PEP 0263, Defining Python Source Code Encodings

<sup>12</sup>See `/usr/lib/python2.7/encoding/__init__.py` near line 99.

## 10.2 The PoC

Recognizing that `cevag` is the Rot13 encoding of Python's `print` command, it's easy to test this strange behavior.

```
% cat poc.py
#! /usr/bin/python
#encoding: rot13
cevag 'Hello World'
% ./poc.py
Hello World
%
```

## 10.3 Caveats

Sadly, this only works in Python versions 2.X, starting with 2.5. My current test with Python 3.3 yields first an unknown encoding error (the "rot13" alias has sadly been removed, so that only "rot-13" and "rot\_13" could work). But Python 3 also distinguishes `strings` from `bytearrays`, which leads to type errors when trying this PoC in general. Perhaps `rot_13.py` in the python distribution might itself be broken?

There are numerous other formats to be found in the encodings directory, such as ZIP, BZip2 and Base64, but I've been unable to make them work. Most lead to padding and similar errors, but perhaps a clever reader can make them work.

And with this, I close the chapter of Python encoding stories. TGSB!

### You can use the versatile new BETSI to plug the more than 150 S-100 bus expansion boards directly into your PET\*!

On a single PC card, BETSI has both interface circuitry and a 4-slot S-100 motherboard. With BETSI, you can instantly use the better than 150 boards developed for the S-100 bus. For expanding your PET's memory and I/O, BETSI gives you the interface. The single board has both the complete interface circuitry required and a 4-slot S-100 motherboard, plus an 80-pin PET connector. BETSI connects to any S-100 type power supply and plugs directly into the memory expansion connector on the side of your PET's case. And that's it. You need no additional cables, interfaces or backplanes. You don't have to modify your PET in any way, and BETSI doesn't interfere with PET's IEEE or parallel ports. And — when you want to move your system — BETSI instantly detaches from your PET.

**BETSI is compatible with virtually all of the S-100 boards on the market, including memory and I/O boards.** BETSI has an on-board controller that allows the use of the high-density low-power "Expandoram" dynamic memory board from S.D. Sales. This means you can expand your PET to its full 32K limit on a single S-100 card! Plus, you won't reduce PET's speed when you use either dynamic or static RAM expansion with BETSI. Additionally, BETSI has four on-board sockets and decoding circuitry for up to 8K of 2716-type PROM expansion (to make use of future PET software available on PROM). BETSI jumpers will address the PROMs anywhere within your PET's ROM area, too.

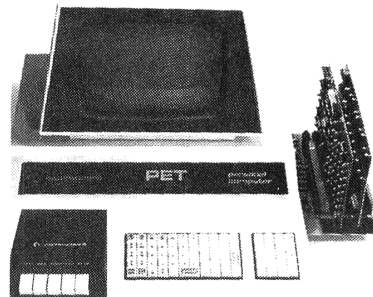
**MAIL ORDERS ARE  
NORMALLY SHIPPED  
WITHIN 48 HOURS.  
VISA AND MASTER-  
CHARGE ORDERS ARE  
BOTH ACCEPTED.**

The BETSI Interface/Motherboard Kit includes all components, a 100-pin connector, and complete assembly and operating instructions for \$119.

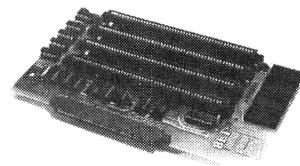
The Assembled BETSI board has four 100-pin connectors, complete operating instructions and a full 6-month Warranty for just \$165.

### FORETHOUGHT PRODUCTS

87070 Dukhobar Road #K  
Eugene, Oregon 97402  
Phone (503) 485-8575.



*BETSI is the new Interface/Motherboard from Forethought Products—the makers of KIMSI™—which allows users of Commodore's PET Personal Computer to instantly work with the scores of memory and I/O boards developed for the S-100 (Imsaï/Altair type) bus. BETSI is available from stock on a single 5½" x 10" printed circuit card.*



*BETSI is available off-the-shelf from your local dealer or (if they're out) directly from the manufacturer.*

Ask about our memory prices, too!