Python

Python

- Python
  - Langage de programmation libre et Open Source.
    - Orienté Objet et dispose d'un typage fort
    - Permet également une programmation de type fonctionnelle
    - Apparu en 1990
    - http://www.python.org
  - Syntaxe
    - Simple, inspirée par C, PHP
  - Extensions
    - Il est possible d'installer des « add-ons » c'est-à-dire des extensions

Prérequis

- Vous devez disposer des produits suivants

```
SC1 / *BASE      IBM Portable Utilities for i
SC1 / 1          OpenSSH, OpenSSL, zlib
SS1 / 33         PASE
```

- Comme d'habitude, les derniers Group PTF sont recommandés

Installation

- Python est packagé dans
  - 5733-OPS (Open Source for IBM i) Option 2
  - PTF SI57008
- Cela vous permet d'installer le runtime Python
  - /QOpenSys/QIBM/ProdData/OPS/Python3.4
- Package à télécharger via ESS
  - F_MULTI_NLV_110_IBM_i_Open_Source_Solutions

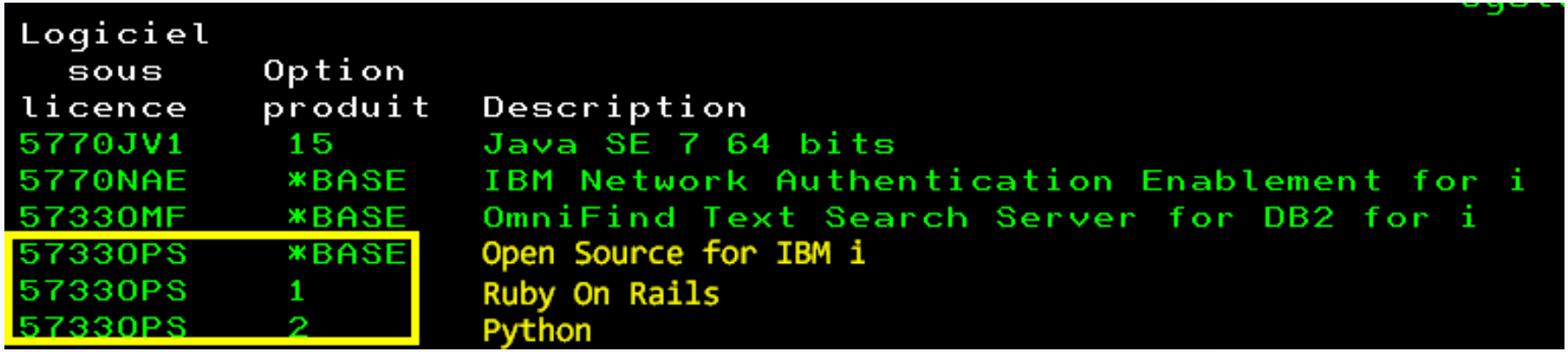
    Gratuit si vous possédez IBM i 7.1 ou 7.2

Installation

- Les extensions sont installables sous forme de PTF
  - DB2 for i connector (basé sur le projet ibm_db
    PTF SI57253
  - Toolkit for IBM I
    PTF SI57254
  - fastCGI gateway (permet l'usage du framework flipflop
    PTF SI57255
  - lightweight web framework (permet l'utilisation de bottle.py)
    PTF SI57256
- Les extensions son installées dans
  - `/QOpenSys/QIBM/ProdData/OPS/Python-pkgs`
- Ces PTF sont à télécharger via FixCentral
  - Facultatives à l'exécution de Python

Installation

- Produit 5733 OPS
  - Option *BASE
  - Option 2 pour Python 3.4

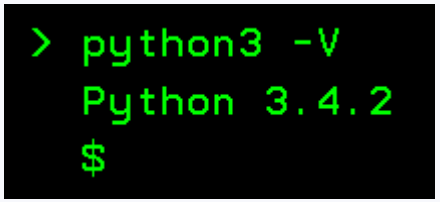


Logiciel sous licence	Option produit	Description
5770JV1	15	Java SE 7 64 bits
5770NAE	*BASE	IBM Network Authentication Enablement for i
5733OMF	*BASE	OmniFind Text Search Server for DB2 for i
5733OPS	*BASE	Open Source for IBM i
5733OPS	1	Ruby On Rails
5733OPS	2	Python

Installation

- Pour vérifier l'installation
  - Sous QSH ou QP2TERM

```
python3 -V
```



Principales commandes

- `python3 / python3.4`
  - Programme principal Python
- `pip3 / pip3.4`
  - Installation de packages
- `2to3 / 2to3-3.4`
  - Transformer du code Python 2.x en code Python 3.x
- `pydoc3 / pydoc3.4`
  - Générateur de documentation
- `pyvenv / pyvenv-3.4`
  - Fourni les environnements virtuels
- `easy_install3 / easy_install-3.4`
  - Compiler, installer et gérer les packages Python, y compris les extensions livrées en standard

Installer les extensions fournies

- Il faut avoir télécharger et appliquer les PTFs correspondantes
- Ensuite
  - Installation du connecteur DB2 natif

    ```
    easy_install3 /QOpenSys/QIBM/ProdData/OPS/Python-pkgs/ibm_db/ibm_db-*.egg
    ```

  - Installation du Toolkit pour IBM i

    ```
    easy_install3 /QOpenSys/QIBM/ProdData/OPS/Python-pkgs/itoolkit/itoolkit-*.egg
    ```

  - Installation du support FastCGI gateway

    ```
    easy_install3 /QOpenSys/QIBM/ProdData/OPS/Python-pkgs/flipflop/flipflop-*.egg
    ```

  - Installation du lightweight web framework

    ```
    easy_install3 /QOpenSys/QIBM/ProdData/OPS/Python-pkgs/bottle/bottle-*.egg
    ```

Installer les extensions fournies

- Exemple d'installation d'une extension

```
  $
> easy_install3 /QOpenSys/QIBM/ProdData/OPS/Python-pkgs/ibm_db/ibm_db-*.egg
  Processing ibm_db-2.0.5.1-py3.4-os400-powerpc.egg
  creating /QOpenSys/QIBM/ProdData/OPS/Python3.4/lib/python3.4/site-packages/
  Extracting ibm_db-2.0.5.1-py3.4-os400-powerpc.egg to /QOpenSys/QIBM/ProdDat
  Adding ibm-db 2.0.5.1 to easy-install.pth file

  Installed /QOpenSys/QIBM/ProdData/OPS/Python3.4/lib/python3.4/site-packages
  Processing dependencies for ibm-db==2.0.5.1
  Finished processing dependencies for ibm-db==2.0.5.1
  $
```

Installer d’autres extensions

- Des packages sont disponibles dans PyPI (Python Package Index)
  - Modules Python Open Source
- Il est possible d’installer ces packages
  - Via la commande pip3
  - Nécessite une connexion internet
- Fonctionne avec
  - QSH
  - PASE (QP2TERM)
- Documentation
  - https://docs.python.org/3/installing/

Installer d'autres extensions

- Exemple
  - `pip3 install xlsxwriter`
  - `pip3 uninstall xlsxwriter`

```
pip3 install xlsxwriter
Downloading/unpacking xlsxwriter
  Downloading XlsxWriter-0.7.3-py2.py3-none-any.whl (132kB): 132kB downloaded
Installing collected packages: xlsxwriter
Successfully installed xlsxwriter
Cleaning up...
#
```

Compilation native

- Certains packages Python nécessitent de compiler du code C/C++
- Il faut donc un compilateur que pip3 puisse utiliser
  - gcc (GNU Compiler Collection)

    gcc est un compilateur libre, très utilisé dans le monde Linux

    Vous pouvez l'installer en suivant les indications données ici

    http://www.youngiprofessionals.com/wiki/index.php/PASE/GCC
  - IBM XL C/C++ for AIX

    Vous pouvez installer le compilateur IBM pour AIX

    L'ensemble des packages n'est pas compatible

Exemple 1

- Pour créer un programme Python
  - Le code des exemples est fourni en .zip annexe
  - Créer le fichier sample.py sur l'IFS avec le contenu suivant

```
from bottle import route, run

port_number=9090                ← Remplacer par votre choix
host_location='10.2.0.1'        ← Remplacer par votre choix

@route('/')
def hello():
    return "OK ! Ca marche !"

run(host=host_location, port=port_number, debug=True)
```

Exemple 1

- Tapez la commande
  - `python3 sample.py`

```
> python3 /home/NB/python/sample.py
  Bottle v0.12.8 server starting up (using WSGIRefServer())...
  Listening on http://10.2.0.1:9090/
  Hit Ctrl-C to quit.
```

- Dans un navigateur web
  - L'IP et le port dépendent de ce que vous avez indiqué dans le fichier `sample.py`

Exemple 1

- Cela génère les jobs
  - QZSHSH

    Interpréteur QSH
  - QP0ZSPWR

    Process Python : serveur web

```
QINTER          QSYS        SBS       0,0                       DEQW
  QPADEV0004    QSECOFR     INT       0,0   CMD-WRKACTJOB       RUN
  QP0ZSPWT      QSECOFR     BCI       0,2   PGM-python3         SELW
  QZSHSH        QSECOFR     BCI       0,0   PGM-QZSHSH          EVTW
```

Exemple 2

- Servir une page web statique au travers d'un programme Python
  - Créer une page web statique de votre choix

    `sample.html`
  - Créer un fichier sample2.py contenant le code Python

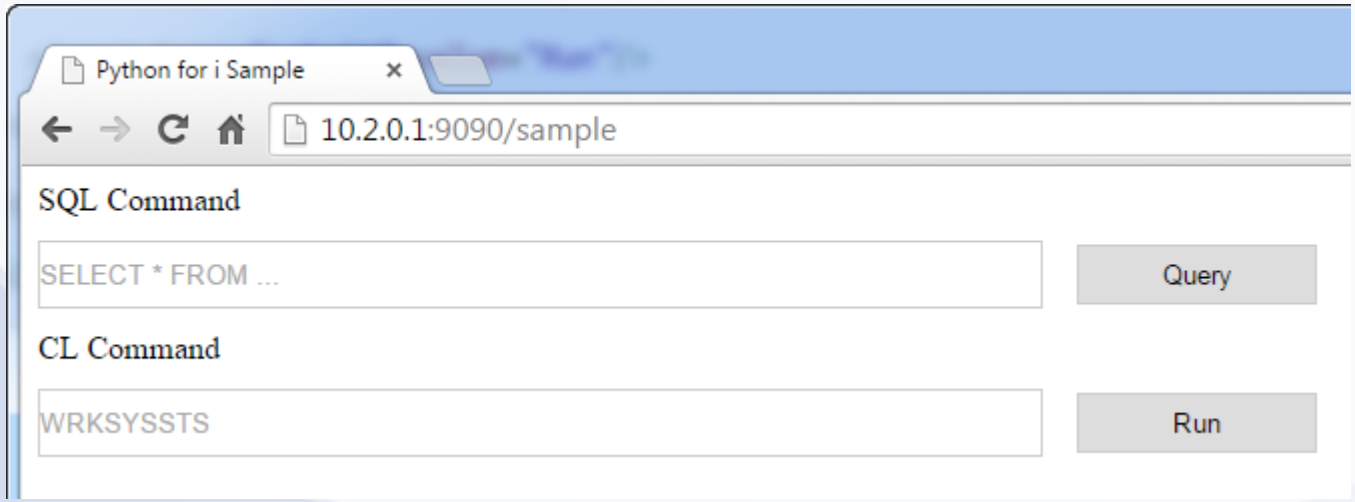
```
from bottle import static_file, route, run

port_number=9090                ← Remplacer par votre choix
host_location='10.2.0.1'        ← Remplacer par votre choix
file_name='sample.html'

@route('/sample')
def sample():
    return static_file(file_name, root='/home/NB/python')

run(host=host_location, port=port_number, debug=True)
```

Exemple 2

- Tapez la commande
  - `python3 sample2.py`

```
> python3 sample2.py
  Bottle v0.12.8 server starting up (using WSGIRefServer())...
  Listening on http://10.2.0.1:9090/
  Hit Ctrl-C to quit.
```

- Dans un navigateur web
  - Le résultat dépend du contenu de votre fichier HTML



Exemple 3

- Accéder à des données DB2 et construire dynamiquement une page HTML
  - Nécessite que l'extension DB2 pour Python soit installée
- Créer les fichiers suivants dans l'IFS
  - Sample.html

    Contient un formulaire permettant la saisie d'une instruction SQL
  - Sample3.py

    Contient le code Python d'exécution de l'instruction et de génération du code HTML en retour

Exemple 3

- sample.html

```
<!DOCTYPE HTML>
<html lang="en-US">
	<head>
		<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
		<title>Python for i Sample</title>
	</head>
<style>
input {
	height:30px;
	border:#ccc solid 1px
}
input[type="text"] {
	width:500px
}
input[type="submit"] {
	margin:1em;
	width:120px
}
</style>
<body>
	<form name="input" action="query" method="post">
		<div>SQL Command </div>
		<input type="text" name="sql" placeholder="SELECT * FROM ..."/>
		<input type="submit" value="Query"/>
	</form>
	<form name="input" action="cmd" method="post">
		<div>CL Command </div>
		<input type="text" name="cl" placeholder="WRKSYSSTS"/>
		<input type="submit" value="Run"/>
	</form>
</body>
</html>
```

Exemple 3

sample3.py

```
from bottle import request, get, post, static_file, route, run
import ibm_db

port_number=9090                ← Votre port
host_location='10.2.0.1'        ← Votre IP
file_name='sample.html'
db_name = '*LOCAL'
username = 'XX'                 ← Votre profil
password = 'XX'                 ← Votre mot de passe

@route('/sample')
def sample():
    return static_file(file_name, root='/home/NB/python')          ← Votre répertoire IFS

@route('/query', method='POST')
def query_ibm_db():
    html_result = ""
    # html table styling
    html_result += "<style>table, th, td {border: 1px solid black;border-collapse: collapse;}"
    html_result += "th, td {padding: 5px;text-align: left;width: 200px;}tr:nth-child(even) {background-
color: #f3f3f3;}"
    html_result += "tr:nth-child(odd) {background-color:#fff;}th {background-color: #008ABF;color:
white}</style>«

conn = ibm_db.connect(db_name, username, password)
```

Exemple 3

- sample3.py - suite

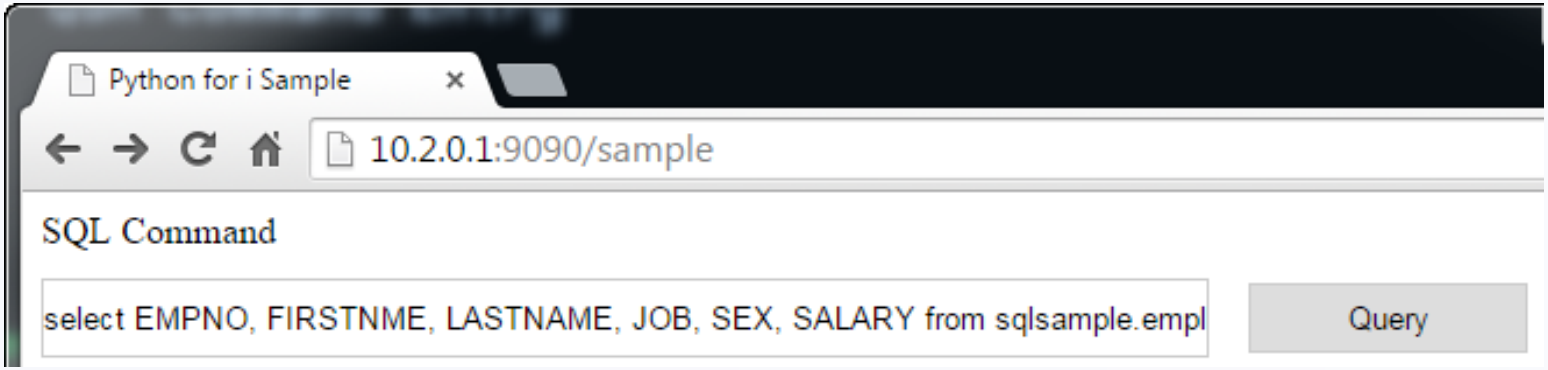
```
    if conn:
      statement = request.forms.get('sql')
      result_set = ibm_db.exec_immediate(conn, statement)

      num_columns = ibm_db.num_fields (result_set)

      html_result += "<table><tr>"
      # create table headers from column names
      for num in range(0, num_columns):
        html_result += "<th>" + ibm_db.field_name(result_set, num) + "</th>"

      html_result += "</tr>"
      # insert results into table
      while (ibm_db.fetch_row(result_set)):
        html_result += "<tr>"
        for num in range(0, num_columns):
          result = ibm_db.result(result_set, num)
          html_result += "<td>" + str(result) + "</td>"
        html_result += "</tr>"

      html_result += "</table>"
      ibm_db.close(conn)
      return html_result

    else:
      return "<p>Connection Failed.</p>"

run(host=host_location, port=port_number, debug=True)
```

Exemple 3

- Exécuter le programme Python

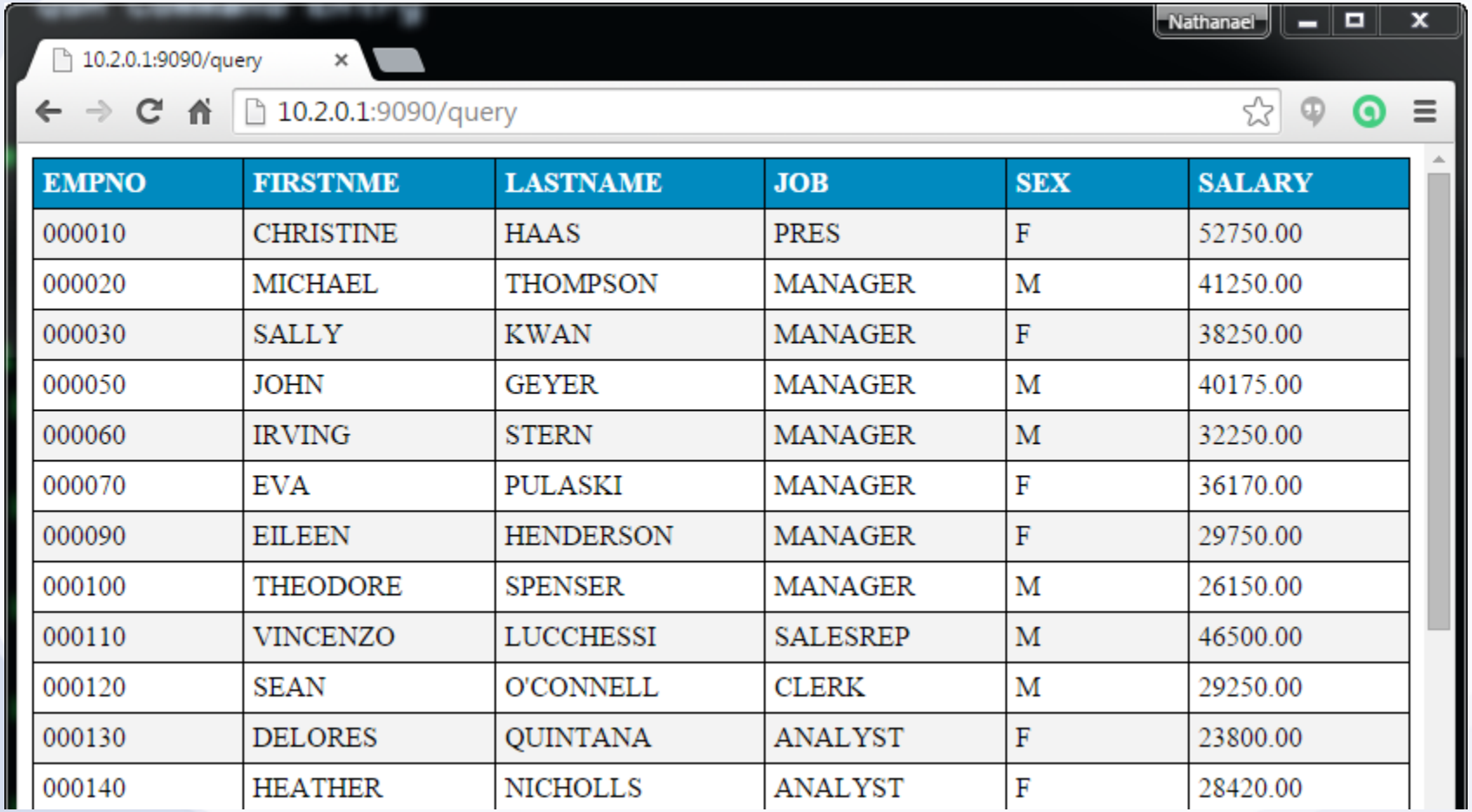
```
> python3 sample3.py
  Bottle v0.12.8 server starting up (using WSGIRefServer())...
  Listening on http://10.2.0.1:9090/
  Hit Ctrl-C to quit.
```

- Navigateur



Exemple 3

- Résultat



EMPNO	FIRSTNME	LASTNAME	JOB	SEX	SALARY
000010	CHRISTINE	HAAS	PRES	F	52750.00
000020	MICHAEL	THOMPSON	MANAGER	M	41250.00
000030	SALLY	KWAN	MANAGER	F	38250.00
000050	JOHN	GEYER	MANAGER	M	40175.00
000060	IRVING	STERN	MANAGER	M	32250.00
000070	EVA	PULASKI	MANAGER	F	36170.00
000090	EILEEN	HENDERSON	MANAGER	F	29750.00
000100	THEODORE	SPENSER	MANAGER	M	26150.00
000110	VINCENZO	LUCCHESSI	SALESREP	M	46500.00
000120	SEAN	O'CONNELL	CLERK	M	29250.00
000130	DELORES	QUINTANA	ANALYST	F	23800.00
000140	HEATHER	NICHOLLS	ANALYST	F	28420.00

Exemple 4

- Exécuter des commandes, accéder aux objets natifs IBM i
  - Nécessite d'avoir installé l'option `itoolkit`
- On ajoute une section de code à notre fichier Python déjà existant
  - Copié ici en sample4.py pour être complété

Exemple 4

- Code à ajouter
  - En début de fichier

```
from itoolkit import *
from itoolkit.lib.ilibcall import *     #for local jobs
```

  - Permet d'indiquer que l'on utilise l'extension itoolkit

Exemple 4

- En fin de fichier

```
@route('/cmd', method='POST')
def cmd_toolkit():

    html_result = "<pre><p>"

    cl_statement = request.forms.get('cl')

    itool = iToolKit()

    # xmlservice
    itransport = iLibCall()

    itool.add(iCmd5250(cl_statement,cl_statement))
    itool.call(itransport)
```

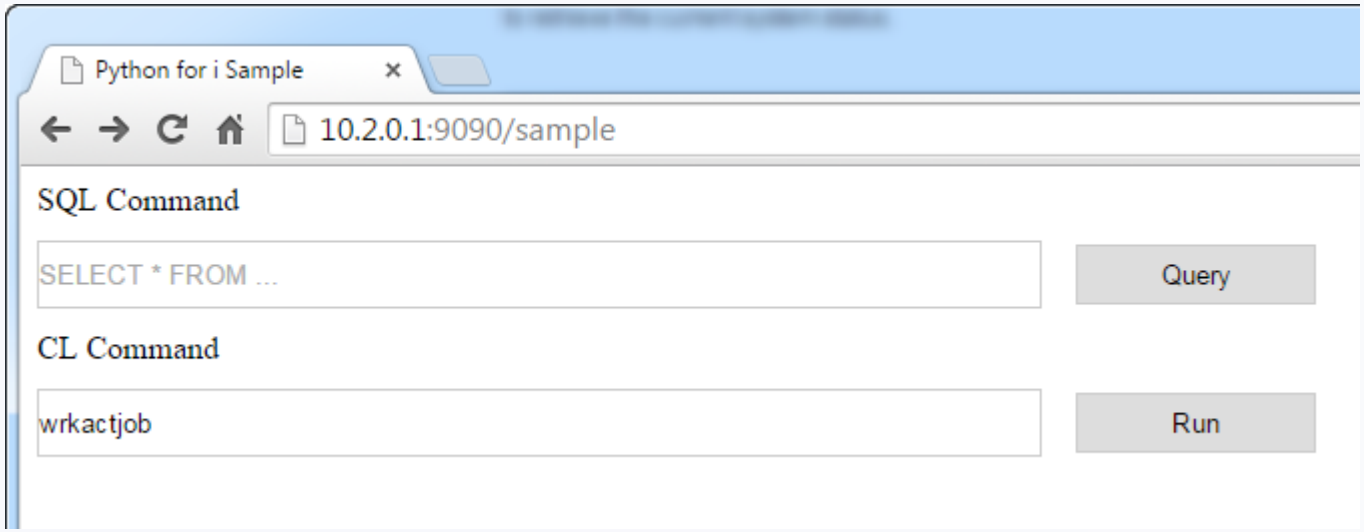
Exemple 4

- En fin de fichier - suite

```
# results from list
output_list = itool.list_out()
for output_outer in output_list:
    for output_inner in output_outer:
        html_result += output_inner

# results from dict
"""
output_dict = itool.dict_out(cl_statement)
if 'error' in output_dict:
    html_result += (output_dict['error'])
    exit()
else:
    html_result += (output_dict[cl_statement])
"""

html_result += "</p></pre>"

return html_result
```

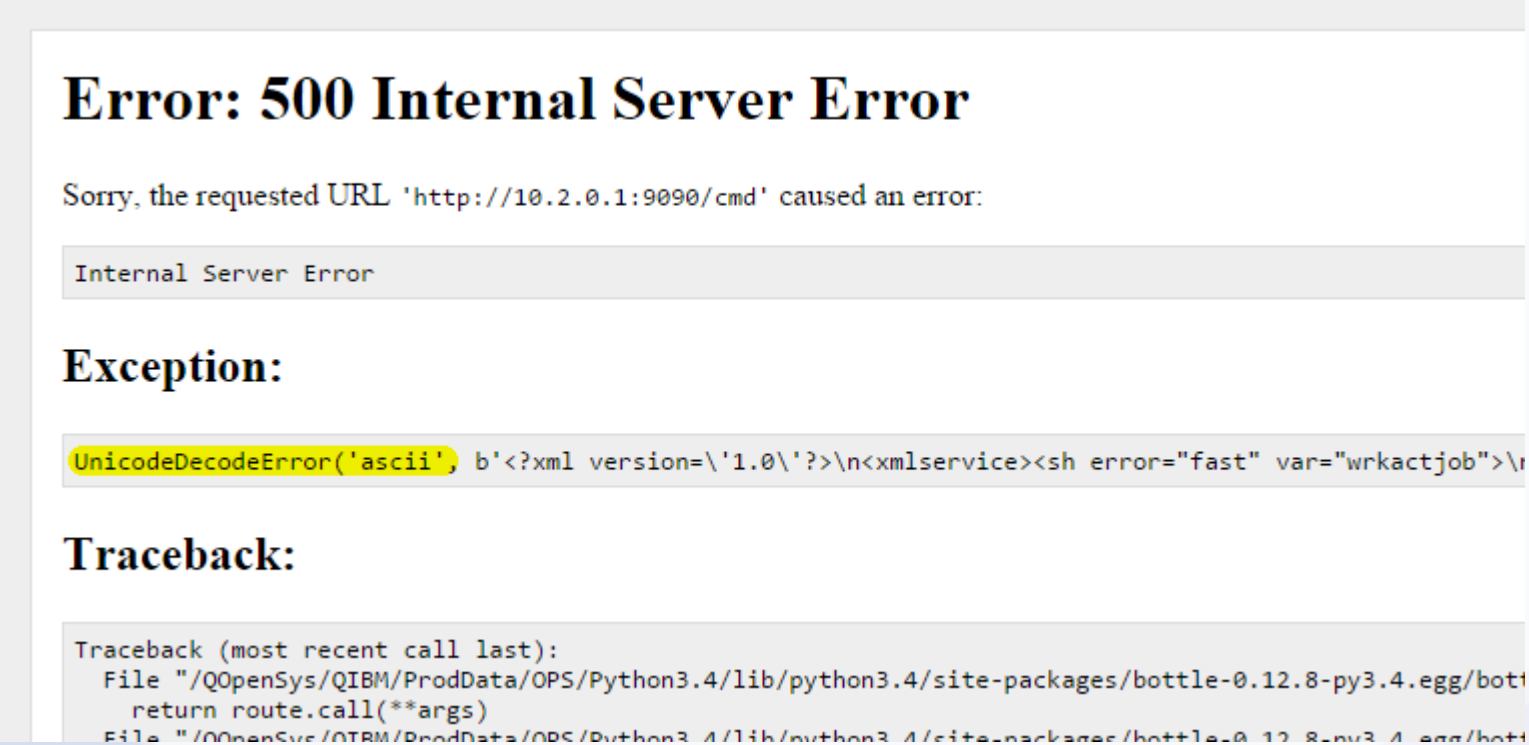
Exemple 4

- Lancer le programme Python
  - python3 sample4.py
- Navigateur



Exemple 4

- Résultat
  - Problème d'encodage des caractères
  - Python est ici utilisé par un utilisateur disposant d'un CCSID 1147 (France €)



Error: 500 Internal Server Error

Sorry, the requested URL `'http://10.2.0.1:9090/cmd'` caused an error:

```
Internal Server Error
```

Exception:

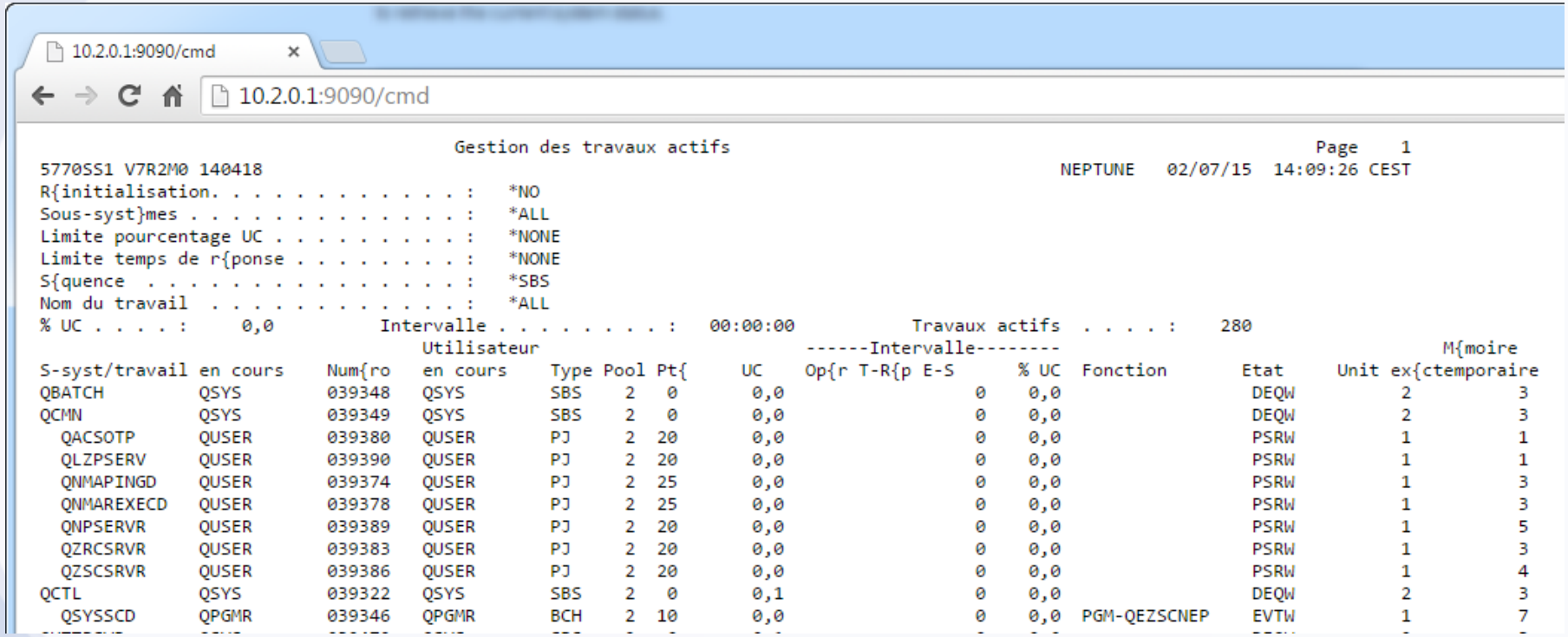
```
UnicodeDecodeError('ascii', b'<?xml version=\'1.0\'?>\n<xmlservice><sh error="fast" var="wrkactjob">\
```

Traceback:

```
Traceback (most recent call last):
  File "/QOpenSys/QIBM/ProdData/OPS/Python3.4/lib/python3.4/site-packages/bottle-0.12.8-py3.4.egg/bott
    return route.call(**args)
```

Exemple 4

- En utilisant un profil avec un CCSID 37



10.2.0.1:9090/cmd

10.2.0.1:9090/cmd

```
                                Gestion des travaux actifs                                                  Page    1
5770SS1 V7R2M0 140418                                                                    NEPTUNE   02/07/15  14:09:26 CEST
R{initialisation. . . . . . . . . . . . :   *NO
Sous-syst}mes . . . . . . . . . . . . . :   *ALL
Limite pourcentage UC . . . . . . . . . :   *NONE
Limite temps de r{ponse . . . . . . . . :   *NONE
S{quence  . . . . . . . . . . . . . . . :   *SBS
Nom du travail  . . . . . . . . . . . . :   *ALL
% UC . . . . :     0,0          Intervalle . . . . . . . . :   00:00:00          Travaux actifs  . . . . :    280
                                Utilisateur                                ------Intervalle--------                                M{moire
S-syst/travail en cours     Num{ro   en cours     Type Pool Pt{     UC     Op{r T-R{p E-S      % UC  Fonction        Etat      Unit ex{ctemporaire
QBATCH         QSYS         039348   QSYS         SBS     2   0       0,0                    0    0,0                    DEQW         2           3
QCMN           QSYS         039349   QSYS         SBS     2   0       0,0                    0    0,0                    DEQW         2           3
  QACSOTP      QUSER        039380   QUSER        PJ      2  20       0,0                    0    0,0                    PSRW         1           1
  QLZPSERV     QUSER        039390   QUSER        PJ      2  20       0,0                    0    0,0                    PSRW         1           1
  QNMAPINGD    QUSER        039374   QUSER        PJ      2  25       0,0                    0    0,0                    PSRW         1           3
  QNMAREXECD   QUSER        039378   QUSER        PJ      2  25       0,0                    0    0,0                    PSRW         1           3
  QNPSERVR     QUSER        039389   QUSER        PJ      2  20       0,0                    0    0,0                    PSRW         1           5
  QZRCSRVR     QUSER        039383   QUSER        PJ      2  20       0,0                    0    0,0                    PSRW         1           3
  QZSCSRVR     QUSER        039386   QUSER        PJ      2  20       0,0                    0    0,0                    PSRW         1           4
QCTL           QSYS         039322   QSYS         SBS     2   0       0,1                    0    0,0                    DEQW         2           3
  QSYSSCD      QPGMR        039346   QPGMR        BCH     2  10       0,0                    0    0,0  PGM-QEZSCNEP      EVTW         1           7
```

- Il subsiste quelques problèmes de caractères accentués
  - A régler dans l'utilisation du itoolkit