Álvaro Gala Martínez  
Adrián Pérez Fernández  
Andrés Casado García  
Daniel Montero García

PRACTICA 2 PYTHON

PARSING DE HTTP GET Y POST REQUEST

# INTRODUCCION

La practica 2 se basa en implementar el parseo de las request Get y Post mediante el uso de diccionarios, listas y strings.

Para ello se ha realiza una limpieza de los strings recibidos, luego mediante un bucle For, se itera línea por línea, para crear el diccionario.

# ¿QUÉ SIGNIFICA PARSEAR?

Cuando hablamos de parsear se trata de transformar una entrada de texto en una estructura de datos que es apropiada para ser procesada. Se puede realizar este parseo desde Strings o JSONS, que podemos convertir en Objetos, Diccionarios, Listas… Existen varias herramientas que facilitan el parseo a Objetos, como por ejemplo jsonschema2pojo que crea diferentes clases en Java a partir de un JSON. Haciendo que el parseo sea más rápido.

# CÓDIGO

Ahora vamos a explicar cómo hemos realizado el código para realizar el parseo. Esta función nos devuelve un diccionario con la petición que le hemos pasado en String.

Para la prueba de la Práctica 2 únicamente se le pasa a la llamada de la función String hardcodeado, es decir que no se realiza ninguna request y por lo tanto no varía según la entrada la obtención del diccionario.

Para ello creamos un diccionario y limpiamos el String.

def requests(request):  
 dictionary = dict()  
 strip = re.sub('[\r]', '', request)  
 strip = re.split('[\n]', strip)  
 strip = list(filter(''.\_\_ne\_\_, strip))

Tras esto iteramos línea por línea del string limpio, mediante un bucle for, y vamos creando el diccionario según sea una petición POST o GET.

if line.\_\_contains\_\_('GET'):  
 pal = line.split(' ')  
 dictionary['Method'] = pal[0]  
 dictionary['URL'] = pal[1]  
 dictionary['Version'] = pal[2]  
 val = pal[1].split('=', 1)  
 list\_message = {'message': val[1]}  
  
elif line.\_\_contains\_\_('POST'):  
 pal = line.split(' ')  
 dictionary['Method'] = pal[0]  
 dictionary['URL'] = pal[1]  
 dictionary['Version'] = pal[2]

Esto lo hacemos para obtener si se trata de una request POST o GET, ya que hay parte de la estructura que es diferente en cada uno.

De forma general el diccionario se forma con las siguientes claves.

elif line.\_\_contains\_\_('HTTP'):  
 val = line.split('/', 1)  
 dictionary[val[0].strip()] = val[1].strip()  
  
elif line.\_\_contains\_\_('message'):  
 val = line.split('=', 1)  
 list\_message = {val[0]: val[1]}  
 dictionary['Params'] = list\_message

else:  
 val = line.split(':', 1)  
 dictionary[val[0].strip()] = val[1].strip()

Si la línea contiene “HTTP”, el Split será “/”, en el caso de que contenga “message”(POST) el separador será “=”, el resto de líneas el separador será “:”

Esto dentro del bucle va formando clave: valor de nuestro diccionario.

Después únicamente imprimimos por pantalla el valor del diccionario. Esto se ha hecho con un cast a JSON, para darle el indentado y que se vea de una forma visualmente más sencilla.

print(json.dumps(dictionary, indent=2))

Para poder usar este diccionario en las siguientes prácticas, se ha incluido todo en una función.

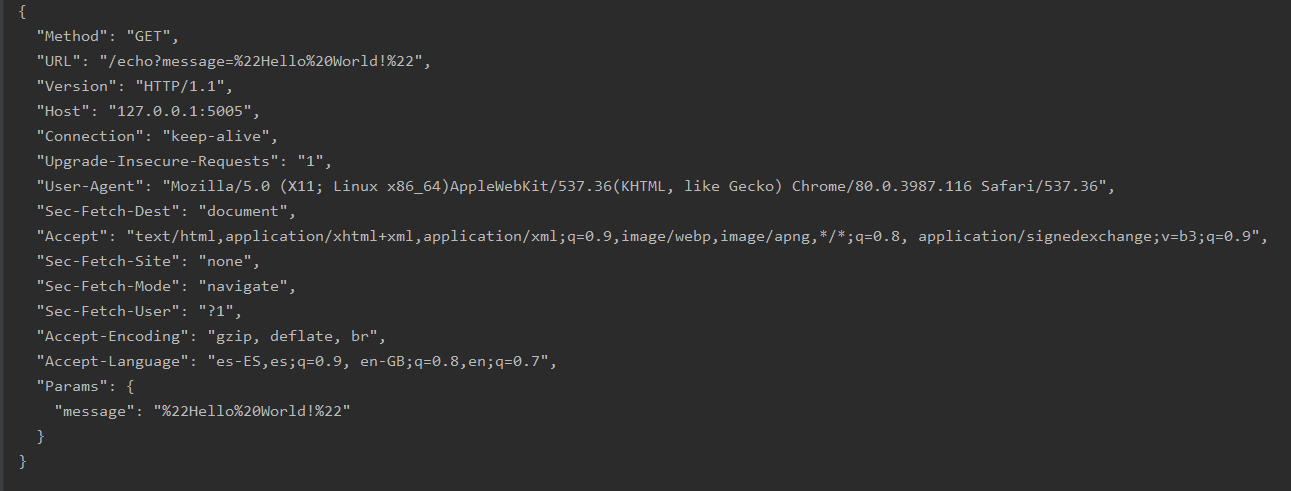
def requests(request):

Está función espera como argumento un String y devuelve el diccionario.

return dictionary

# SALIDAS

# GET



# POST

