DigitalHouse>

Certified Tech Developer
The Ultimate Degree

Infraestructura I

Objetivos

En el siguiente ejercicio vamos a realizar un script un poco más complejo. El lenguaje lo eligen ustedes (Bash o PowerShell).

¿Qué recibimos?

Un listado de web services o APIs libres que vamos a utilizar para el Ejercicio 2.

- Genderize
- Nationalize

Un arquivo hospedado en GitHub con una lista de 456 nombres en español:
https://raw.githubusercontent.com/olea/lemarios/master/nombres-propios-es.txt

API Genderize

Esta API (https://genderize.io) nos permite obtener un resultado en formato JSON con información relacionada con el género de un determinado nombre. Por ejemplo, para el nombre “Emilio”, el resultado que se obtiene es:

```
{
  "name":"emilio",
  "gender":"male",
  "probability":0.99,
  "count":25883
}
```

Las propiedades del objeto JSON que obtenemos son:
name: el nombre por el que consultamos es incluido en la respuesta.
gender: el género del nombre por el que consultamos.
probability: probabilidad en que la predicción de género sea correcta.
count: cantidad de veces que ese nombre ha sido consultado.

Para obtener el resultado previamente indicado, la llamada a ejecutar es una llamada tipo GET incluyendo el parámetro 'name' a la URL: https://api.genderize.io/. Por ejemplo: https://api.genderize.io/?name=emilio

> **Tip:** Para el ejercicio, la propiedad que nos interesa es la propiedad 'gender'.

> **Importante:** La API tiene un límite de 1000 llamadas diarias. De hacer uso de la misma más de 1000 (mil) veces en un mismo día, deberemos esperar 24 h para volver a utilizarla.

API Nationalize

Esta API (https://nationalize.io/) nos permite predecir la nacionalidad de una persona en función de su nombre. El resultado es entregado en formato JSON. Por ejemplo, para el nombre "Emilio", el resultado que se obtiene es:


```
{
  "name":"emilio",
  "country":
  [
    {
      "country_id":"ES",
      "probability":0.13849973696268725
    },
    {
      "country_id":"MZ",
      "probability":0.10571390688819501
    },
    {
      "country_id":"CL",
      "probability":0.08975185148274877
    }
  ]
}
```

Las propiedades del objeto JSON que obtenemos son:
name: el nombre por el que consultamos es incluido en la respuesta.
country: contiene un array de objetos por cada uno de los países en donde potencialmente se usa ese nombre, ordenado de mayor probabilidad a menor. Cada uno incluye las siguientes propiedades:

country_id: el código del país basado en la normativa ISO 3166.
probability: la probabilidad de que el nombre corresponda a una persona de ese país.

Para obtener el resultado previamente indicado, la llamada a ejecutar es una llamada tipo GET incluyendo el parámetro 'name' a la URL: https://api.nationalize.io/. Por ejemplo: https://api.nationalize.io/?name=emilio

> **Tip:** Para el ejercicio, la propiedad que nos interesa es la propiedad 'country_id' del objeto que representa al país con mayor probabilidad.

> **Importante:** la API tiene un límite de 1000 llamadas diarias. De hacer uso de la misma más de 1000 (mil) veces en un mismo día, deberemos esperar 24 h para volver a utilizarla.

Instrucciones

De forma individual, ejecutá los siguientes pasos:
Del archivo que contiene los 456 nombres, seleccionar aleatoriamente 5 nombres distintos que comiencen con "A", 5 nombres distintos que comienzan con "L" y 5 nombres distintos que no comienzan ni con "A", ni con "L".

Ejemplo de una lista válida:
Alonso
Alfonso
Alfredo
Agustín

Ananías
Lidia
Lino
Lorena
Lorenzo
Leonor
Montserrat
Patricio
Porfirio
Irene
Vicente

Luego, para cada uno de los nombres obtenidos del archivo, someterlos a ambas APIs (Genderize y Nationalize) e imprimir un mensaje en pantalla por cada nombre indicando género y nacionalidad.

Pista: en PowerShell considerá estas líneas, asegurate de entender qué hacen para poder incluirlas y/o adaptarlas a tu script. Reemplazá el texto en rojo con la dirección web del archivo.

```
$req = Invoke-WebRequest -Method Get -Uri <urlAlListado>
$nombresConA = $req.Content.split("`n") | Where-Object { $_ -like
"A*" }

1..5 | ForEach-Object {
  $random = Get-Random -Minimum 0 -Maximum $($nombresConA.count-1)
  $nombresConA[$random]
}
```

Pista: een **Bash** considerá estas líneas, asegurate de entender qué hacen para poder

incluirlas y/o adaptarlas a tu script. Reemplazá el texto en rojo con la dirección web del archivo.

```
req=`curl <urlAlListado> | shuf`

countA=0

for nombre in $req
do
    if [[ $nombre == A* ]] && [ $countA -le 4 ] ;
    then
        echo $nombre
        let "countA++"
    fi
done
```