Web Services

14 de Noviembre 2016

¿QUÉ SON?

Conjunto de aplicaciones cliente-servidor que se comunican a través de la WEB mediante un protocolo diseñado para ello.

La interacción es entre un **servidor** y una **aplicación**, por lo que se necesita un **protocolo** para poder interpretar la información intercambiada.

HTTP

- Protocolo **más común** utilizado por las arquitecturas de Web Services.
- Cliente hace una solicitud y servidor responde con la información solicitada.
- Es un protocolo **sin estados**: cada transacción es independiente de las otras.
- Funciona con la definición de **métodos o verbos** que indican la acción a desarrollar.

HTTP

- **GET:** **pide** una representación del recurso especificado (json, HTML, XML).
- **POST:** **crea** un nuevo recurso con los datos enviados.
- **DELETE:** **elimina** el recurso especificado.

- Existen más!...

Estructura

REQUEST = VERBO + URI

```
import request

url = ‘https://api.trello.com/1’
response = request.request(‘GET’, url)
dict = response.json()
```

```
import request

url = ‘https://api.trello.com/1’
response = request.get(url)
dict = response.json()
```

Lo que necesitamos en python...

`import requests`

- Para **hacer solicitudes** a un web service.

`import flask`

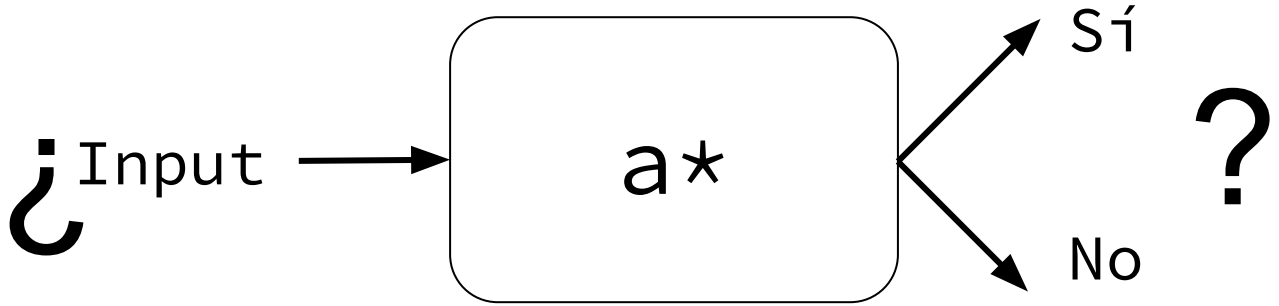
- Para **generar** un web service.
- Es un web micro-framework.
- Existen más librerías como esta.

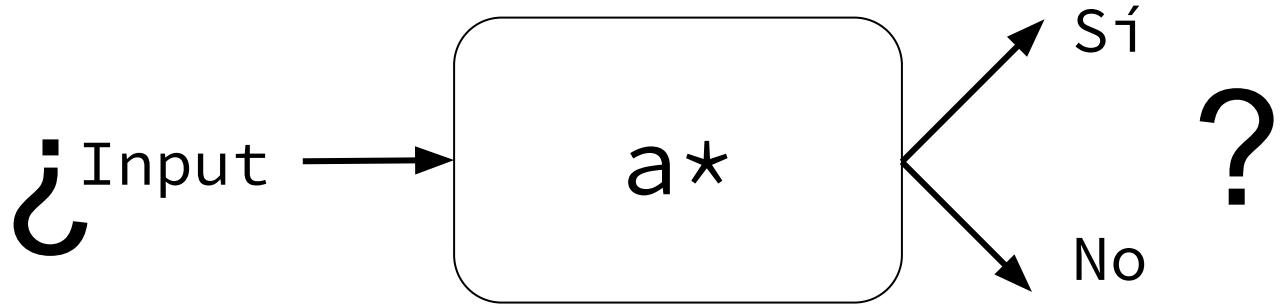
Expresiones Regulares

¡Regex!

¿Qué son?

Podemos pensar las expresiones regulares como máquinas a las que se les entrega una palabra y responden ‘Sí’ o ‘No’.


¿Input
a*
Sí
No
?



Sí: ‘a’, ‘aa’, ‘aaa…’, …
No: ‘ab’, ‘abaa…’, …

¿Por qué quiero crear estas máquinas?

Es mucho lo que hay que aprender, y solo se puede aprender practicando.

`http://regexr.com/`

[ab]

Expresión regular que acepta a todas las palabras de largo 1 que utilicen sólo las letras en {a, b}.

Ejemplos:

‘a’, ‘b’

[ab]?

Expresión regular que acepta a todas las palabras de largo 1 que utilicen sólo las letras en {a, b}, o ninguna de ellas.

Ejemplos:

‘a’, ‘b’, ‘’

[ab]+

Expresión regular que acepta a todas las palabras de largo 0 o más que utilicen sólo las letras en {a, b}.

Ejemplos:

‘a’, ‘b’, ‘aa’, ‘bb’, ‘ab’, ‘ba’, ‘abaa...’, ‘baba...’

$[ab]^*$

Expresión regular que acepta a todas las palabras de largo 0 o más que utilicen sólo las letras en {a, b}.

Ejemplos:

‘a’, ‘b’, ‘aa’, ‘bb’, ‘ab’, ‘ba’, ‘abaa...’, ‘baba...’, ‘’

¡También acepta la palabra vacía!

(foo|bar)

Expresión regular que acepta a todas las palabras que sean `foo` o `bar`.

Ejemplos:

foo, bar

Expresión regular que acepta a todas las palabras de largo 1 que utilicen cualquier letra conocida por la humanidad.

Ejemplos:

‘ß’, ‘ɬ’, ’¶’, ’→’, ’ø’

```
(?:[a-z0-9!#$%&'*+/=?^_`{|}~-]+(?:\.[a-z0-9!#$%&'*+/=?^_`{|}~-]+)*|"(?:[\x01-\x08\x0
b\x0c\x0e-\x1f\x21\x23-\x5b\x5d-\x7f]|\\[\x01-\x09\x0b\x0c\x0e-\x7f])*")@(?:(?:[a-z0-
9](?:[a-z0-9-]*[a-z0-9])?\.)+[a-z0-9](?:[a-z0-9-]*[a-z0-9])?|\[(?:(?:25[0-5]|2[0-4][0-9]|[01]
?[0-9][0-9]?)\.){3}(?:25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?|[a-z0-9-]*[a-z0-9]:(?:[\x01-\x08
\x0b\x0c\x0e-\x1f\x21-\x5a\x53-\x7f]|\\[\x01-\x09\x0b\x0c\x0e-\x7f])+)\])
```

Expresión regular que acepta a todos las palabras equivalentes a correos electrónicos válidos según el estándar RFC 5322.