

# BOTS EN WIKIPEDIA

## INTRODUCCIÓN A *pywikipediabot*

Emilio José Rodríguez Posada  
Noelia Sales Montes

Universidad de Cádiz

2010

# QUÉ DEBERÍAS SABER

- Algo de **Python**
- Algo sobre **edición de wikis** (MediaWiki)
- Funcionamiento de una **comunidad** wiki
  - **Consenso** antes de hacer miles de cambios
  - Sin consenso quizás haya que deshacerlo todo
    - Doble trabajo
- Sopesar si el bot será rentable
  - Hay tareas que no pueden automatizarse bien
  - ¿Tasa aciertos >>>> tasa fallos? Entonces vale

# ¿QUÉ ES UN BOT?

- Un **programa** que hace muchos cambios, que sería tedioso hacer a mano
- Pueden ser:
  - Totalmente **automáticos**
    - Aún así conviene verificar los cambios
  - **Pidan confirmación** antes de actuar
    - Son más lentos pero más seguros
    - Ideales para pocos cambios (<1000) y que éstos sean peligrosos de automatizar

# ¿QUÉ ES *pywikipedia*bot?

- Es un ***framework***
  - Conjunto de funciones que permiten modificar páginas MediaWiki
- Ha alcanzado **madurez**
  - Es el principal *framework* que se usa en Wikipedia
- Está hecho en **Python**
- Es **software libre**

# INSTALANDO *pywikipediabot*

Supongo que estamos en Linux:

- Web: <http://pywikipediabot.sourceforge.net/>
- Instalación:  
> `svn co http://svn.wikimedia.org/svn\`  
`root/pywikipedia/trunk/pywikipedia`
- Preferencias:
  - Crear *user-config.py*
    - Ejecutar  
> `python interwiki.py` (y rellenar los datos)
  - Primera ejecución:
    - Ahora sí  
> `python interwiki.py A`

# FUNCIONES BÁSICAS (I)

## Primeras funciones...

- `wikipedia.Site()`: **Constructor de sitio**  
`site = wikipedia.Site("es", "wikipedia")`
- `wikipedia.Page()`: **Constructor de página**  
`page = wikipedia.Page(site, u"Universidad de Cádiz")`
- `page.title()`: **Devuelve el título** de la página  
`wtitle = page.title()`
- `page.get()`: **Devuelve el contenido** de la página  
`wtext = page.get()`
- `page.put()`: **Modifica** el contenido  
`page.put(newtext, u"BOT - Arreglando lo que sea...")`

# PRIMER BOT

- **Crear un objeto página para el artículo “Universidad de Cádiz”, imprimir por pantalla su título y su contenido.**
- Pistas:
  - Para usar las funciones hay que importarlas  
`import wikipedia`
  - Usar constructor de sitio y de página
  - Capturar título en una variable (`page.title()`)
  - Capturar texto en una variable (`page.get()`)
  - Imprimir ambas variables por consola

# PRIMER BOT

- **Crear un objeto página para el artículo “Universidad de Cádiz”, imprimir por pantalla su título y su contenido.**
- Pistas:
  - Para usar las funciones hay que importarlalas  

```
import wikipedia
```
  - Usar constructor de sitio y de página
  - Capturar título en una variable (`page.title()`)
  - Capturar texto en una variable (`page.get()`)
  - Imprimir ambas variables por consola

# PRIMER BOT

```
# -*- coding: utf-8 -*-  
import wikipedia  
  
site = wikipedia.Site("es", "wikipedia")  
page = wikipedia.Page(site, u"Universidad de Cádiz")  
wtitle = page.title()  
wtext = page.get()  
wikipedia.output(u"Título: %s" % wtitle)  
wikipedia.output(u"Texto: %s" % wtext)
```

# FUNCIONES BÁSICAS (II)

## Más funciones básicas...

- `page.exists()` : Comprueba si existe el artículo al que “page” apunta
- `page.isRedirectPage()` : Comprueba si es redirección
- `page.isDisambig()` : Comprueba si es desambiguación

# SEGUNDO BOT

- **Cargar la página “Universidad de Cadiz” (ojo, sin acento), devolver su título, comprobar si es una redirección, y mostrar su contenido.**
- Pistas:
  - Las redirecciones son “páginas especiales”.
  - Su contenido se carga con `page.get()`, pero pasándole la variable `get_redirect=True`.

# SEGUNDO BOT

- **Cargar la página “Universidad de Cadiz” (ojo, sin acento), devolver su título, comprobar si es una redirección, y mostrar su contenido.**
- Pistas:
  - Las redirecciones son “páginas especiales”.
  - Su contenido se carga con `page.get()`, pero pasándole la variable `get_redirect=True`.

# SEGUNDO BOT

```
# -*- coding: utf-8 -*-
import wikipedia

site = wikipedia.Site("es", "wikipedia")
page = wikipedia.Page(site, u"Universidad de Cadiz")
wtitle = page.title()
esredirect = page.isRedirectPage()
wtext = page.get(get_redirect=True)
wikipedia.output(u"Título: %s" % wtitle)
wikipedia.output(u"Es redirección? %s" % esredirect)
wikipedia.output(u"Texto: %s" % wtext)
```

# FUNCIONES BÁSICAS (III)

- `page.interwiki()`: Devuelve los **interwikis**
- `page.categories()`: Devuelve las **categorías**
- `page.linkedPages()`: Devuelve los **[[enlaces]]**
- `page.imagelinks()`: Devuelve las **imágenes**

Todas estas funciones devuelven listas de objetos *Page*

# TERCER BOT

- Para “Universidad de Cádiz”, mostrar los títulos de las categorías, el número de enlaces y número de interwikis.
- Pistas:
  - Es posible recorrer con un *bucle for*  
`page.categories()` ¡Es una lista!
  - La longitud de una lista puede saberse con:  
`len(lista)`

# TERCER BOT

- Para “Universidad de Cádiz”, mostrar los títulos de las categorías, el número de enlaces y número de interwikis.
- Pistas:
  - Es posible recorrer con un *bucle for*  
`page.categories()` ¡Es una lista!
  - La longitud de una lista puede saberse con:  
`len(lista)`

# TERCER BOT

```
# -*- coding: utf-8 -*-
import wikipedia

site = wikipedia.Site("es", "wikipedia")
page = wikipedia.Page(site, u"Universidad de Cádiz")
wtitle = page.title()
wikipedia.output(u"Título: %s" % wtitle)
for cat in page.categories():
    wikipedia.output(cat.title())
wikipedia.output(u"No. enlaces: %d" %
                 len(page.linkedPages()))
wikipedia.output(u"No. interwikis: %d" %
                 len(page.interwiki()))
```

# PAGEGENERATORS

Quizás nos interese recorrer todos los artículos de Wikipedia o una parte...

Lo conseguimos con los **pagegenerators**:

- Importamos el módulo *pagegenerators*:

```
import pagegenerators
```

- Hacemos:

```
gen = pagegenerators.AllpagesPageGenerator(start=u"A")
```

- Si no queremos analizar redirecciones, le pasamos el parámetro:

```
includeredirects=False
```

- Precargamos el pagegenerator que hemos creado:

```
preload = pagegenerators.PreloadingGenerator(gen)
```

- **Calcular la media en KiloBytes de los primeros 100 artículos, a partir del comienzo “Be”.**
- Pistas:
  - Crear un pagegenerator pasándole el parámetro `start=u“Be”`
  - Evitamos cargar redirecciones
  - Ir acumulando los tamaños de los artículos:  
`len(page.get())`

# CUARTO BOT

- **Calcular la media en KiloBytes de los primeros 100 artículos, a partir del comienzo “Be”.**
- Pistas:
  - Crear un pagegenerator pasándole el parámetro `start=u“Be”`
  - Evitamos cargar redirecciones
  - Ir acumulando los tamaños de los artículos:  
`len(page.get())`

# CUARTO BOT

```
# -*- coding: utf-8 -*-
import wikipedia, pagegenerators

gen = pagegenerators.AllpagesPageGenerator(start=u"Be",
                                           includedirects=False)
preload = pagegenerators.PreloadingGenerator(gen)
media = 0.0
cont = 0
limite = 100
for page in preload:
    if cont == limite:
        break
    media += len(page.get())
    cont += 1
media = (media/1024)/limite
wikipedia.output(u"La media es: %.2f KB" % media)
```

# QUINTO BOT

- **Hacer un corrector ortográfico: que cambie ortográfico “avion” por “avión” y otras faltas que se os ocurran.**
- Pistas:
  - Usar una estructura dictionary para almacenar los pares de valores:  
`faltas={u"avion":u"avión", ...}`
  - Recorrer el diccionario con un bucle:  
`for clave, valor in faltas.items():`
  - Haciendo la sustitución en el texto así:  
`re.sub(clave, valor, textoarticulo)`
  - ¡Importar módulo re!

# QUINTO BOT

- **Hacer un corrector ortográfico: que cambie ortográfico “avion” por “avión” y otras faltas que se os ocurran.**
- Pistas:
  - Usar una estructura dictionary para almacenar los pares de valores:  
`faltas={u“avion”:u“avión”, ...}`
  - Recorrer el diccionario con un bucle:  
`for clave, valor in faltas.items():`
  - Haciendo la sustitución en el texto así:  
`re.sub(clave, valor, textoarticulo)`
  - ¡Importar módulo re!

# QUINTO BOT

```
# -*- coding: utf-8 -*-
import wikipedia, re

faltas = {u"avion":u"avión", u"dificil":u"difícil"}
page = wikipedia.Page(wikipedia.Site("es", "wikipedia"),
                      u"Wikipedia:Zona de pruebas")
newtext = page.get()
for clave, valor in faltas.items():
    newtext = re.sub(clave, valor, newtext)
page.put(newtext, u"Corrigiendo algunas faltas...")
```

- Cosas que **podríamos mejorar** del corrector ortográfico:
  - Que muestre qué palabras va a corregir, y si no hay falsos positivos, pulsar enter y guardar (bot **semiautomático**)
  - Recorrer toda la wiki con un **pagegenerator**, empezando por la A
  - Mejorar nuestro **diccionario de faltas**, y cargarlo desde un fichero externo

# EJEMPLOS DE BOTS FRECUENTES

Algunos **robots comunes**:

- Interwikis
- Traslado de categorías
- Correctores ortográficos
- Desambiguadores

- **Antivandalismo**
  - En la española: AVBOT
  - En la inglesa: ClueBot
- **Creación** de artículos (botopedia)
  - Útiles (municipios): Rambot creó 30.000 municipios de EEUU en la inglesa
  - Dudosos: asteroides...
- **Actualización** de datos
  - Cada año hay que actualizar cifras de población según el INE

# BUENAS PRÁCTICAS (PARA FINALIZAR)

- **Revisar** los cambios
  - Casos no contemplados en el código provocan ediciones erróneas
  - El usuario que lanza el bot es el único responsable de lo que hace su bot
- **Pedir permiso** y alcanzar consenso
  - BotFlag: evita que los cambios salgan en la página de cambios recientes y la inunden

¿Dudas?  
¿Reflexiones?

Gracias por su atención

# ESTA PRESENTACIÓN ES LIBRE

Copyright 2009 Emilio José Rodríguez Posada

Este trabajo se publica bajo la siguiente licencia:  
Creative Commons Attribution-Share Alike License

<http://creativecommons.org/licenses/by-sa/2.5/>

Usted es libre de:

- copiar, distribuir y comunicar públicamente la obra
- hacer obras derivadas

Bajo las condiciones siguientes:

**ATRIBUCIÓN** Debes reconocer la autoría de la obra en los términos especificados por el propio autor o licenciante.

**LICENCIAMIENTO RECÍPROCO** Si alteras, transformas o creas una obra a partir de esta obra, solo podrás distribuir la obra resultante bajo una licencia igual a ésta.

