Alfred

Versión 0.0.1.dev6

22 de mayo de 2019

Contents:

Me alegra que hayas decidido aprender a programar en este maravilloso nuevo lenguaje de programación.

Advertencia: Actualmente Alfred está en fase de desarrollo, por lo que es posible que gran parte de esta guía no cubra el procedimiento a seguir en tu entorno.

En **Mayo de 2019**, todo lo aquí descrito sólamente ha sido probado en la distribución de `Linux Mint` con la versión de `Python 3.6.7`.

Aquí te dejo una lista con los contenidos que podrás encontrar en esa guía:

CAPÍTULO 1

Instalación

Alfred ha sido diseñado apoyándose en el lenguaje de programación **Python3**. Es posible que en algunos Sistemas Operativos salten errores de dependencias incumplidas si Python no está instalado en el sistema.

Además, como Alfred aún está en fase de desarrollo, los archivos standalone y los módulos ejecutables no tienen ninguna garantía de funcionar correctamente en cualquier sistema. En caso de encontrar una incompatibilidad, puedes reportarla directamente en la sección de errores del repositorio de GitHub.

1.1 Linux

Si estás corriendo Linux, tienes varias formas de conseguir a `Alfred`.

La primera es *descargar el archivo* correspondiente a tu distribución y arquitectura desde repositorio de GitHub. Aquí podrás encontrar todas las versiones junto a los cambios realizados respecto a las anteriores versiones.

Una vez descagado, asegúrate de añadir la ruta del archivo a tu `PATH`. Para ello puedes crear una nueva ruta en la que poner tu binario, por ejemplo `/usr/local/alfred/bin`. Para poder acceder a él desde cualquier parte de tu terminal, puedes añadir la siguiente línea al archivo `$HOME/.profile` o al archivo `/etc/profile` (en caso de que desees una instalación para todos los usuarios del sistema):

```
export PATH=$PATH:/usr/local/alfred/bin
```

También puedes instalar a `Alfred` mediante `pip`. Asegúrate de tener instalado **Python3** junto a su correspondiente versión de **pip**:

```
python --version
pip --version
```

Una vez comprobado, sólamente tienes que ejecutar el siguiente comando:

```
pip install alfred-lang
```

1.2 Probando la instalación

Desde tu terminal, prueba a poner el siguiente comando:

Si te aparece el mensaje de ayuda, significa que la instalación se ha realizado con éxito.

Puedes probar tus habilidades con este lenguaje de programación mediante el modo interactivo o ejecutando tus programas:

```
alfred -i
alfred --interactive

alfred holamundo.alf
```

CAPÍTULO 2

Lenguaje Alfred

A pesar de que Alfred es un lenguaje bastante flexible estructuralmente, sigue ciertas directrices a la hora de interpretar el código fuente de los programas.

Además, por convenio, todos los archivos que incluyan código fuente escrito en este lenguaje de programación deberán tener la extensión `.alf`.

2.1 Alfred

El archivo que contenga el flujo principal del programa ha de comenzar obligatoriamente con la palabra reservada `Alfred`. Además, esta palabra ha de ser única en todo el código fuente.

Cualquier otra instrucción que preceda a dicha palabra reservada generará un error de sintaxis parecido a este:

```
[] (Línea: 1) Sintaxis inválida: ...
```

De este modo, si se desea generar un programa `.alf` vacío, el contenido será similar a este:

```
Alfred.
```

2.2 Comentarios

Como casi cualquier otro lenguaje de programación, Alfred acepta la inclusión de comentarios en el código. Aunque la semántica sea más que expresiva, en muchas ocasiones es recomendable, por no decir necesario, el uso de anotaciones.

Para ello, se disponen de los símbolos `(` y `)`, los cuales encapsularán cualquier contenido que será ignorado a la hora de procesar el código.

```
(Los comentarios pueden ir precediendo a la palabra reservada Alfred)

Alfred. (También pueden ir intercalados entre instrucciones) Di "Buenos días, Sr.␣
↪Wayne".
```

(continué en la próxima página)

(proviene de la página anterior)

```
(Además, Alfred acepta comentarios
multilínea sin la necesidad de
caracteres especiales adicionales)
```

Advertencia: Hay que tener cuidado al comentar código. La ausencia de un paréntesis puede convertir tus instrucciones en simples anotaciones.

```
Alfred.
Di "Esto no es un comentario".
(Hay que intentar no olvidarse de cerrar los comentarios
Di "Esto es parte del comentario"
(Alfred ignorará cualquier código si éste es tratado como un comentario)
Di "Esto tampoco es parte de un comentario".
```

La ausencia de un paréntesis, es decir, si la cantidad total de pares de paréntesis es impar, Alfred generará una excepción similar a esta:

```
[] Caracter inválido (1,~184): (
```

2.3 Instrucciones

2.3.1 Di

La instrucción `Di` permite mostrar un texto por pantalla. Cualquier argumento que se le pase será propiamente tratado para que se pueda mostrar por la salida estándar del sistema. Además, un nuevo salto de línea será añadido tras evaluar la instrucción.

```
Alfred. Di "Encantado de conocerte, Batman".
```

Resultado:

```
Encantado de conocerte, Batman
```

Atención: Los caracteres escapados tales como `\n`, `\r` o `\t` **son tratados de manera literal**, por lo que si deseas que se muestren por pantalla, sólamente has de usarlos como si de un editor de textos común se tratase.

```
Alfred. Di "\n no funciona como un salto de línea,
pero este mensaje va a ser multilínea. Además,
si deseas tabular algo (\t), has de hacerlo      de esta manera".
```

Resultado:

```
\n no funciona como un salto de línea,
pero este mensaje va a ser multilínea. Además,
si deseas tabular algo (\t), has de hacerlo      de esta manera
```

2.3.2 Escribe

De la misma manera que la instrucción `Di`, `Escribe` permite mostrar textos por la salida estándar del sistema. La única diferencia es que no se añade una salto de línea al evaluar la instrucción.

Esto es muy útil a la hora de concatenar textos.

```
Alfred. Escribe "Hola ", escribe "Mundo" y di "!".
```

Resultado:

```
Hola Mundo!
```

2.3.3 Pregunta

La instrucción `Pregunta` permite interactuar con la entrada estándar del sistema. Esta instrucción hace uso de un parámetro **opcional**, el cual corresponde al texto que va a ser mostrado antes de realizar la interacción con el teclado.

```
Alfred. Pregunta.
```

```
Alfred,
escribe "¿Cómo te llamas? " y pregunta.
(es similar a...)
Pregunta "¿Cómo te llamas? ".
```

Advertencia: Es posible que en algunos sistemas, teclas como el tabulador, el retorno o las flechas no sean correctamente tratadas y den como resultado a entradas similares a `^[[D` o a `^D`

El resultado es guardado en una variable especial propia de Alfred, llamada **variable temporal**, la cual solamente es accesible mediante la instrucción `Guardalo en`.

2.3.4 Guardalo en

La instrucción `Guardalo en` permite obtener el resultado de comandos tales como `Pregunta`, que hayan hecho uso de la **variable temporal** característica de Alfred.

`Guardalo en` toma como parámetro el nombre de la variable en la que se quiera almacenar la información perteneciente a la **variable temporal**.

```
Alfred. Pregunta "¿Cómo te llamas? ", guardalo en nombre,
escribe "Encantado de conocerte, " y di nombre.
```

Resultado:

```
¿Como te llamas? Bruce
Encantado de conocerte, Bruce
```

CAPÍTULO 3

Contribuir

Si deseas contribuir en el desarrollo de Alfred, ¡no dudes en hacerlo!. Alfred es un lenguaje de programación de código abierto y cualquier ayuda por parte de la comunidad es bienvenida.

3.1 Herramientas necesarias

Para poder contribuir, es necesario tener algunas herramientas instaladas en el sistema:

- python3
- pip3
- git
- make

Para disponer de un entorno controlado que no interfiera con el resto de tu sistema, hay disponibles herramientas como virtualenv que puedes usar:

Lista 1: Windows

```
pip install virtualenv

virtualenv venv
.\venv\Scripts\activate
# ... comandos a ejecutar ...
deactivate
```

Lista 2: Linux

```
pip install virtualenv

virtualenv venv
source venv/bin/activate
```

(continué en la próxima página)

(proviene de la página anterior)

```
# ... comandos a ejecutar ...
deactivate
```

3.2 Obtener el Código Fuente

Para descargar el código fuente de Alfred, puedes hacerlo mediante la utilidad de descarga directa que ofrece GitHub. Aún así, es posible que luego tengas que configurar el repositorio local para poder realizar **Pull Requets** (PR) o mantenerlo actualizado.

Es por ello que la forma recomendable de obtener el código fuente es mediante la herramienta de control de versiones `git`:

```
git clone https://github.com/CosasDePuma/Alfred
```

Para probar que la descarga se ha realizado con éxito, puedes poner el siguiente comando dentro de la carpeta `Alfred`:

```
make check
```

3.3 Compilar el Código Fuente

Antes de compilar el programa, es necesario instalar los requisitos del código fuente. Esto se realiza ejecutando los siguientes comandos:

```
pip install -r requirements.txt
pip install -r requirements-dev.txt
```

Alfred cuenta con varios archivos **Makefile** que proporcionan todas las instrucciones necesarias, independientemente del Sistema Operativo que estés usando. Para compilar, basta con escribir los comandos:

```
make build
make clean
```

Para instalar el módulo de manera local mediante `pip`, basta con poner el comando:

```
make install
```

Esto sobreescrirá cualquier versión anterior que hayas instalado desde el repositorio de `PyPI`. De la misma manera, se puede desinstalar usando el comando:

```
make uninstall
```

Advertencia: `pip3` ha de ser accesible de manera global mediante el comando `pip` para que las directivas `install` y `uninstall` funcionen correctamente.