Informe PTI

Lab 5

Autores

Marc Cabezas

Antonio Arellano

1. Familiarización con el entorno

Se ha configurado la máquina, instalando Go, CURL y GIT para poder descargar correctamente los archivos y poder ejecutar los ficheros .go.

Se han ejecutado estos comandos:

**sudo** apt-get install curl

**sudo** apt-get install git

wget https://storage.googleapis.**com**/golang/go1.7.1.linux-amd64.tar.gz

**sudo** tar -C /usr/local -xzf go1.7.1.linux-amd64.tar.gz

sudo **chmod** -R 777 /usr/**local**/go

export PATH=$PATH:/usr/local/go/bin

**mkdir** $HOME/**go**

**mkdir** $HOME/**go**/src

**export** GOPATH=**$HOME**/go

git clone https://github.**com**/YOUR\_GITHUB\_USER/pti\_golang.git

**cd** $HOME/**go**/src/pti\_golang

**mkdir** hello

**cd** hello

wget https://gitlab.fib.upc.edu/pti/pti/raw/master/p5\_goREST/src/hello/hello.**go**

**go** install pti\_golang/hello

$HOME/**go**/bin/hello

Una vez estaba todo instalado se han realizado pruebas para comprobar que todo se había instalado correctamente.

Se han realizado más pruebas con varios ejemplos:

|  |
| --- |
|  |
| Figura 1. Salida realizada por el ejemplo hello.go |

Después se han realizado ejemplos con un Endpoint que nosotros mismos hemos creado y que nos devolvía un valor estatico.

2. Caso práctico

En esta sección se explicarán las soluciones adoptadas para los problemas propuestos, luego éstas soluciones nos ayudarán a elaborar un conclusión.

2.1.Webserver.go

Se nos pedía crear un programa con el cual se pudiera hacer diferentes acciones respecto a unos endpoints.

Las acciones son las siguientes:

**-Guardar datos mediente una llamada al Endpoint**

Con un endpoint poder guardar los nuevos alquileres que se pasen por parametros:

|  |
| --- |
|  |
| Figura 2. Salida por terminal al realizar correctamente el alta de un alquiler. |

|  |
| --- |
|  |
| Figura 3. Fichero donde se almacena toda la información de los alquileres. |

**-Obtener una lista con todos los alquileres**

Con este Endpoint podemos obtener todos los alquileres que hemos realizado.

|  |
| --- |
|  |
| Figura 4. Lista de todos los alquileres realizados. |

3. Conclusiones

Las conclusiones se vuelven a estructurar de la siguiente manera, primero comentaremos las conclusiones que hemos sacado del punto 1 (Familiarización con el entorno), luego las del punto 2 (Caso práctico) y finalmente a partir de las conclusiones anteriores, intentaremos sacar una conclusión definitiva de las tecnologías con las que hemos trabajado, así como puntos a favor y en contra o posibles escenarios en donde éstas tecnologías serían de gran utilidad.

3.1 Conclusiones de la familiarización con el entorno

La instalación de los paquetes necesarios no ha dado problemas. En la prueba de los ejemplos no ha habido ningún problema. Un punto positivo es el entorno de desarrollo, Go es un nuevo idioma que puede resultar muy últi tener unas nociones básicas.

3.2 Conclusiones del caso práctico

No ha sido difícil entender cómo poder generar un Endpoint en el que poder cargar los datos que son necesarios para ir creando instancias e irlas guardando en un fichero .csv.

Tampoco ha sido muy costoso realizar otra llamada que nos devolviese todos los alquileres guardados y que nos lo mostrase en un formato JSON.

Podemos concluir que Go es una herramienta especialmente útil y rápido para desplegar entornos en los que serán necesarios llamadas API.

3.3 Conclusiones finales

Básandonos en las anteriores conclusiones , podemos argumentar que Go es un gran idioma de programación útil como ya hemos visto en entornos de BackEnd y demás.