

INFORME LIBRERÍA “PyPiston” – MARKEL SETIEN Y UNAI DURAN

El objetivo de esta tarea consiste en crear una librería funcional de “PyPi” de acuerdo a todos los conocimientos obtenidos durante estos cuatro años académicos de programación, especialmente aquellos aprendidos durante este último curso, como, por ejemplo, programación orientada a objetos y tratamiento de errores y excepciones.

En este caso, se propone una librería dirigida a la búsqueda y recomendación de coches satisfaciendo unos requisitos establecidos por el usuario. Además, se permite la opción de realizar una reserva del vehículo elegido.

Como ya se ha mencionado, la programación orientada a objetos era un requisito fundamental a la hora de desarrollar la librería por lo que sobre ésta se cimenta la propuesta ofrecida.

De forma adicional, y para poder realizar la recomendación de coches, se ha hecho uso de una API utilizada para realizar búsqueda de vehículos de acuerdo a unos parámetros de entrada (<https://api-ninjas.com/>).

Además, durante el proceso de desarrollo se ha utilizado la herramienta *GitHub* como repositorio de archivos. El link del repositorio utilizado: <https://github.com/unaiduran/MarkelUnaiProgra.git>

A continuación, se explica paso a paso y punto por punto el funcionamiento de la aplicación, así como su configuración y estructura primitiva. Con el fin de facilitar la comprensión, se informa que los paréntesis indicarán el tipo de dato en caso de los atributos y el tipo de *input* y *output* (*input* -> *output*) en caso de los métodos.

Clase “Cliente”

En primer lugar, se genera una clase llamada “Cliente”, que representa un cliente con información básica. Esta clase está formada por los siguientes atributos (no precisan explicación):

- *edad* (*int*)
- *mayoria_edad* (*bool*)
- *nombre* (*str*)
- *residencia* (*str*)
- *permiso* (*list*)

Y la constituyen los siguientes tres métodos:

- *comprobacion_edad* (-> *bool*): Comprueba si el cliente es mayor de edad, devuelve e inserta *True* en el atributo *mayoria_edad* en caso de que lo sea, *False* en caso contrario.
- *añadir_permiso* (*str* ->): Añade un permiso a la lista (atributo) de permisos de conducción del cliente.
- *show_cliente*: Muestra los datos del cliente en consola.

Clase “ClienteExt”

Esta segunda clase representa una versión extendida de cliente con versión adicional.

La forman los siguientes atributos:

- *cliente (Cliente)*: Clase *Cliente* heredada.
- *vigencia_permiso (str)*: La vigencia del permiso de conducción del cliente.
- *necesidades_especiales (bool)*: Indica si el cliente tiene necesidades especiales a considerar a la hora de conducir y realizar la reserva de un vehículo.
- *compromiso_medioambiental (str)*: El nivel del compromiso mediambiental del cliente.
- *nivel_cliente (str)*: Nivel del cliente.

Y los siguientes métodos:

- *es_vigente (-> bool)*: Comprueba si el permiso del cliente está vigente (haciendo uso de la fecha actual y el atributo *vigencia_permiso*).
- *añadir_necesidad_especial*: Añade necesidades especiales al cliente. Establece atributo *necesidades_especiales* a *True*.
- *show_clientext*: Muestra los datos del cliente extendido en consola.

Clase “Coche”

Esta clase representa un coche deseado por el usuario. Es decir, aquí se guardan las características deseadas por el cliente para considerar en la recomendación de su coche.

Está formada por los siguientes atributos:

- *make (str)*: La marca del coche.
- *Model (str)*: El modelo del coche.
- *fuel_type (str)*: El tipo de combustible del coche (gasolina, diesel).
- *Year (str)*: El año del coche.
- *Transmission (str)*: El tipo de transmisión del coche.
- *Cylinders (str)*: El número de cilindros del motor.

Y los siguientes métodos:

- *buscar_coche (list ->)*: Realiza una búsqueda de coches en la API de acuerdo a la información especificada. En este método se incorpora un *try...except* que comprueba la respuesta devuelta por la API.
- *mostrar_inform_busqueda_coche*: Muestra los resultados de la búsqueda en forma de tabla.

Clase “Reserva”

Para finalizar, se define la clase *Reserva* que representa una reserva de un coche realizada por un cliente, de ahí el hecho de que herede información de las clases *ClienteExt* y *Coche*.

Está formada por los siguientes atributos:

- *num_resultado_coche_reservado (int)*: El número de resultado de la búsqueda de coche a reservar.

Y los siguientes métodos:

- *realizar_reserva (int ->)*: Realiza una reserva de un coche en función del número de resultado de búsqueda. En este método se realizan dos *try...except*: El primero comprueba que el dato introducido sea un entero válido; el segundo, en cambio, comprueba que el número de resultado insertado realmente exista entre los resultados de búsqueda ofrecidos. Por otro

lado, y para poder realizar la reserva, se comprueba que el cliente disponga de permiso de conducción tipo *B*, que sea mayor de edad y que su permiso de conducción esté vigente.

Funcionamiento cronológico esperado de la librería

En primer lugar, se espera la creación de un objeto *Cliente* con la información de la clase requerida. Después, se espera la creación del objeto *ClienteExt* en el que ultimar detalles necesarios sobre el objeto previamente creado.

Para continuar, el cliente deberá crear un objeto *Coche* en el que especificará los atributos deseados para su búsqueda. Después, éste llevará a cabo la búsqueda y la selección del coche deseado haciendo uso de los métodos relativos a esta clase.

Para finalizar, y en caso de que el cliente lo decida, realizará la reserva de uno de los vehículos previamente mostrados mediante la creación del objeto *Reserva* y su método correspondiente.

SPHINX

Para crear la documentación de la librería se ha utilizado la herramienta SPHINX. A continuación, se adjuntan algunas capturas de pantalla que muestran y confirman el proceso seguido hasta la creación de ésta.

```
(base) PS C:\Users\unai\Documents\BData\cuarto_curso\programacion\MarkelUnaiProgra> sphinx-quickstart
Welcome to the Sphinx 7.2.6 quickstart utility.

Please enter values for the following settings (just press Enter to
accept a default value, if one is given in brackets).

Selected root path: .

You have two options for placing the build directory for Sphinx output.
Either, you use a directory "_build" within the root path, or you separate
"source" and "build" directories within the root path.
> Separate source and build directories (y/n) [n]: n

The project name will occur in several places in the built documentation.
> Project name: PistonPy
> Author name(s): MarkelUnai
> Project release []: V0

If the documents are to be written in a language other than English,
you can select a language here by its language code. Sphinx will then
translate text that it generates into that language.

For a list of supported codes, see
https://www.sphinx-doc.org/en/master/usage/configuration.html#confval-language.
> Project language [en]: es

Creating file C:\Users\unai\Documents\BData\cuarto_curso\programacion\MarkelUnaiProgra\conf.py.
Creating file C:\Users\unai\Documents\BData\cuarto_curso\programacion\MarkelUnaiProgra\index.rst.
Creating file C:\Users\unai\Documents\BData\cuarto_curso\programacion\MarkelUnaiProgra\Makefile.
Creating file C:\Users\unai\Documents\BData\cuarto_curso\programacion\MarkelUnaiProgra\make.bat.

Finished: An initial directory structure has been created.

You should now populate your master file C:\Users\unai\Documents\BData\cuarto_curso\programacion\MarkelUnaiProgra\index.rst and create other documentation
source files. Use the Makefile to build the docs, like so:

    make builder

where "builder" is one of the supported builders, e.g. html, latex or linkcheck.

(base) PS C:\Users\unai\Documents\BData\cuarto_curso\programacion\MarkelUnaiProgra>
```

El resultado:

PyPiston

¡Bienvenido a la documentación de PyPiston!

/ ¡Bienvenido a la documentación de PyPiston!

[View page source](#)

¡Bienvenido a la documentación de PyPiston!

La librería proporciona una interfaz para interactuar con una API de búsqueda de información sobre coches, gestionando datos de clientes, coches y reservas.

Cliente

La clase Cliente representa clientes con información básica

```
class PyPiston.Cliente(edad, nombre, residencia, permiso=[]) [source]
```

Representa un cliente con información básica.

Representa un cliente con información básica.

```
__init__(edad, nombre, residencia, permiso=[]) [source]
```

Inicializa una instancia de la clase Cliente.

Parameters:

- edad** – La edad del cliente.
- nombre** – El nombre del cliente.
- residencia** – La residencia del cliente.
- permiso** – Una lista de permisos del cliente (opcional). Inicializa una instancia de la clase Cliente.

```
comprobacion_edad() [source]
```

Subida de librería a *Pypi*

Para finalizar y cumplir el último de los quehaceres de la tarea, se subió la librería a *Pypi*, como lo corroboran las siguientes evidencias.

```
(base) C:\Users\unai\Documents\BData\cuarto_curso\programacion\MarkelUnaiProgra>python setup.py sdist bdist_wheel
running sdist
running egg_info
writing PyPiston.egg-info\PKG-INFO
writing dependency links to PyPiston.egg-info\dependency_links.txt
writing requirements to PyPiston.egg-info\requires.txt
writing top-level names to PyPiston.egg-info\top_level.txt
reading manifest file 'PyPiston.egg-info\SOURCES.txt'
writing manifest file 'PyPiston.egg-info\SOURCES.txt'
running check
creating PyPiston-0.1.4
creating PyPiston-0.1.4\PyPiston.egg-info
copying files to PyPiston-0.1.4...
copying README.md -> PyPiston-0.1.4
copying setup.py -> PyPiston-0.1.4
copying PyPiston.egg-info\PKG-INFO -> PyPiston-0.1.4\PyPiston.egg-info

(base) C:\Users\unai\Documents\BData\cuarto_curso\programacion\MarkelUnaiProgra>py -m pip install --upgrade twine
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: twine in c:\users\unai\appdata\roaming\python\python310\site-packages (4.0.2)
Requirement already satisfied: importlib-metadata>=3.6 in c:\programdata\anaconda3\lib\site-packages (from twine) (4.3)
Requirement already satisfied: keyring>=15.1 in c:\programdata\anaconda3\lib\site-packages (from twine) (23.4.0)
Requirement already satisfied: pkginfo>=1.8.1 in c:\programdata\anaconda3\lib\site-packages (from twine) (1.8.3)
```

En primer lugar, como se recomienda, se subió la librería a *testPypi*, una vertiente separada de *pypi* sobre la que asegurarse del correcto funcionamiento del programa antes de implementarlo definitivamente a *pypi*.

```
(base) C:\Users\unai\Documents\BData\cuarto_curso\programacion\MarkelUnaiProgra>py -m twine upload --repository testpypi dist/*
Uploading distributions to https://test.pypi.org/legacy/
Enter your username:
```

A la hora de subirlo a la página se obtuvo un error; debido a que la autenticación de doble factor estaba activada, se debía usar la API token, en vez de el usuario y la contraseña, para subir la librería a la página.

```
(base) C:\Users\unai\Documents\BDData\cuarto_curso\programacion\MarkelUnaiProgra>python -m twine upload --repository testpypi dist/*
Uploading distributions to https://test.pypi.org/legacy/
Enter your username: MarkelUnai
Enter your password:
Uploading PyPiston-0.1.0-py3-none-any.whl
100% ----- 4.4/4.4 kB • 00:00 • ?
WARNING Error during upload. Retry with the --verbose option for more details.
ERROR HTTPError: 401 Unauthorized from https://test.pypi.org/legacy/
User MarkelUnai has two factor auth enabled, an API Token or Trusted Publisher must be used to upload in place of password.
```

Create API token

Ficha para «PyPiston»

Permisos: Cargar paquetes

Ámbito: La cuenta completa (todos los proyectos)

```
pypi-AgEsdGVzdC5weXBpLm9yZWlkMWVjY2U0MGQtZjQ5MC00MDA2LTlkMTMtMmRkMmE5OTdiZDE1AAIqWzMsIjlkZDZiOTdkLTdlMGMtNDA1MC04Njc3LTE2ODg5ZGEzOTQzMjJdAAAGIPzhwKYRH1qGFh1yGwkkWIGPtknbsJKaiXceXpcUXv9t
```

Por motivos de seguridad, esta ficha aparecerá solo una vez. **Cópiela ahora.**

Copiar ficha


Quitar ficha


```
(base) C:\Users\unai\Documents\BDData\cuarto_curso\programacion\MarkelUnaiProgra>py -m twine upload --repository testpypi --verbose --username __token__ dist/*
Uploading distributions to https://test.pypi.org/legacy/
INFO dist\PyPiston-0.1.0-py3-none-any.whl (1.2 KB)
INFO dist\PyPiston-0.1.0.tar.gz (1.9 KB)
INFO username set by command options
INFO Querying keyring for password
Enter your password:
INFO username: __token__
INFO password: <hidden>
Uploading PyPiston-0.1.0-py3-none-any.whl
100% ----- 4.4/4.4 kB • 00:00 • ?
INFO Response from https://test.pypi.org/legacy/:
200 OK
Uploading PyPiston-0.1.0.tar.gz
100% ----- 4.7/4.7 kB • 00:00 • ?
INFO Response from https://test.pypi.org/legacy/:
200 OK

View at:
https://test.pypi.org/project/PyPiston/0.1.0/
```

Ahora sí, se ha subido correctamente a testpypi.

Una vez comprobado que todo funciona correctamente en testpy, se puede subir a pypi. Al tener la autenticación de doble factor activada, se necesita también crear el API Token.



 MarkelUnai

Su cuenta

Sus proyectos

Sus organizaciones

Configuración de cuenta

Publicando

Create API token

Ficha para «PyPiston»

Permisos: Cargar paquetes
Ámbito: La cuenta completa (todos los proyectos)

pypi-AgE1cHlwaS5vcmcCJDZkZmIxODZlLTRjZDMtNDY5Ni1iZTF1LTE5NDI4ZWZlMGJlNwACKlszLCJjYThkMjUxYi0zMzc3LTQyNjktYTYSNy02NWY1NmIwM2UwYWQ1XQAABiCdtA6iFwaONC5d5492ncZlx29X0CuUfyahlsPQrC0aLA


Por motivos de seguridad, esta ficha aparecerá solo una vez. Cópiala ahora.


Copiar ficha

Quitar ficha

```
(base) C:\Users\unai\Documents\BDData\cuarto_curso\programacion\MarkelUnaiProgra>py -m twine upload --verbose --username
__token__ -password pypi-AgE1cHlwaS5vcmcCJDZkZmIxODZlLTRjZDMtNDY5Ni1iZTF1LTE5NDI4ZWZlMGJlNwACKlszLCJjYThkMjUxYi0zMzc3LT
QyNjktYTYSNy02NWY1NmIwM2UwYWQ1XQAABiCdtA6iFwaONC5d5492ncZlx29X0CuUfyahlsPQrC0aLA dist/*
Uploading distributions to https://upload.pypi.org/legacy/
INFO dist\PyPiston-0.1.4-py3-none-any.whl (2.0 KB)
INFO dist\PyPiston-0.1.4.tar.gz (2.0 KB)
INFO username set by command options
INFO password set by command options
INFO username: __token__
INFO password: <hidden>
Uploading PyPiston-0.1.4-py3-none-any.whl
100% ----- 7.4/7.4 kB • 00:00 • ?
INFO Response from https://upload.pypi.org/legacy/:
200 OK
Uploading PyPiston-0.1.4.tar.gz
100% ----- 7.0/7.0 kB • 00:00 • ?
INFO Response from https://upload.pypi.org/legacy/:
200 OK

View at:
https://pypi.org/project/PyPiston/0.1.4/
```



 MarkelUnai

PyPiston 0.1.4

✓

Versión más reciente

pip install PyPiston

Publicación: half a minute ago

La librería proporciona una interfaz para interactuar con una API de búsqueda de información sobre coches, gestionando datos de clientes, coches y reservas.

Gestionar proyecto

Navegación

Descripción de proyecto

Histórico de versiones

Archivos de descarga

Descripción de proyecto

Librería PyPiston

Esta librería Python facilita la interacción con una API de búsqueda de información sobre coches, permitiendo gestionar datos de clientes, coches y reservas. Aquí se presenta un resumen de las clases clave y sus funciones.

Ya está la librería en *pypi*.

Líneas futuras

Como se puede observar en el código de la librería, se almacenan datos sobre el cliente del estilo “nivel de cliente”, “compromiso medioambiental”, etc. de los que todavía no se hace uso. Aun así, se pretende, en futuras versiones de la librería, implementar estas funcionalidades teniendo esa información en cuenta en la búsqueda de vehículos.