Sesión web scraping

Sergi Pons

Objetivo Principal

Proporcionar las herramientas esenciales para realizar web scraping de forma autónoma y superar las barreras iniciales de la programación.

Objetivos Específicos

1. **Familiarización con el Entorno de Desarrollo**
    - Configuración del entorno de trabajo
    - Introducción a Jupyter Notebook
    - Gestión de paquetes y dependencias
2. **Dominio de Herramientas Básicas**
    - Jupyter Notebook como entorno interactivo
    - Pandas para manipulación de datos
    - Requests y BeautifulSoup para procesamiento HTML
3. **Habilidades Prácticas**
    - Extracción de datos estructurados
    - Automatización de procesos de recolección de datos
    - Análisis básico de datos web

Estructura de la sesión

- Web scraping y sus implicaciones
- Qué es Python
- Principales librerías de Python para Scraping
- Consideraciones antes de hacer scraping
- Proceso general del web scraping
- Entornos de desarrollo
- Comandos principales y shortcuts
- Recursos y materiales (fuentes públicas)
- Git y creación de entorno
- Primeros pasos de programación con python

Web Scraping

Definición

- Técnica de extracción automatizada de datos de sitios web
- Proceso de recolección de información estructurada de páginas web
- Transformación de datos no estructurados en datos estructurados

Usos Comunes

- Investigación de mercado
- Agregación de noticias
- Análisis de sentimientos
- Machine Learning y análisis de datos

Técnicas Principales

Parsing HTML

- Análisis del código fuente HTML
- Uso de selectores CSS y XPath
- Extracción de datos mediante DOM (Document Object Modelling)

APIs

- Uso de APIs públicas (INE)
- REST APIs (Google maps, spotify…)
- API Wrappers (biblioteca que simplifica tareas en la interacción con la API, como Tweepy)

Python es un lenguaje de programación interpretado que fue creado por Guido van Rossum en 1991 y que persigue una sintaxis que favorezca un código legible.

Python es también un programa ejecutable (intérprete) - lee el código de archivos .py y lo ejecuta.

Características principales

- Fácil de aprender
- Con una gran cantidad de librerías disponibles
- Interactivo
- Multiplataforma
- De código libre
- Interpretado
- Orientado a Objetos

fuente: Sesión web scraping Carles Illa (2018)

Herramientas Principales

Librerías Python

- **Requests**
  - Peticiones HTTP
  - Manejo de sesiones
  - Headers personalizados
- **Beautiful Soup 4**
  - Parser HTML/XML
  - Fácil de usar
  - Ideal para páginas estáticas
- **Scrapy**
  - Framework open source
  - Escalable
  - Automatiza la navegación entre páginas
- **Selenium**
  - Puede usarse en páginas que usan JavaScript
  - Interacción como clics o rellenados como si fuera usuario real

Aspectos Legales

- Términos de servicio
- Robots.txt (pensado especialmente para crawlers)
- Derechos de autor
- Políticas de privacidad

```
User-agent: *
Disallow: /bc/

User-agent: Twitterbot
Allow: /bc/*

User-agent: facebookexternalhit
Allow: /bc/*

User-agent: GPTBot
Disallow: /

User-agent: ClaudeBot
Disallow: /

User-agent: anthropic-ai
Disallow: /

User-agent: Claude-Web
Disallow: /
```

1. Para todos los bots (*):

   ❌ No pueden acceder a la sección `/bc/` (ejemplo: contenido exclusivo o privado).

2. Bots de redes sociales (Twitterbot, Facebook):

   ✅ Sí pueden acceder a `/bc/*` (probablemente para compartir enlaces en sus plataformas).

3. Bots de inteligencia artificial (GPTBot, ClaudeBot...):

   ❌ Bloqueados totalmente (no pueden rastrear ninguna parte del sitio).

Librerías y frameworks populares

- **Requests**, HTTP for Humans
- **Scrapy**, a Fast and Powerful Scraping and Web Crawling Framework
- NumPy, the fundamental package for scientific computing with Python
- SciPy, a library of algorithms and mathematical tools for python
- **Pandas**, high-performance, easy-to-use data structures and data analysis tools for Python
- Matplotlib, python data visualization
- SQLAlchemy, the Database Toolkit for Python
- **Jupyter Notebook**, an interactive command-line terminal for Python
- Django, a high-level Python Web framework
- NLTK, a leading platform for building Python programs to work with human language data

fuente: Sesión web scraping Carles Illa (2018)

Proceso general de Web Scraping

1. Obtención del contenido HTML de la página web
2. Extracción de la información (parsing)
3. Transformación
4. Almacenamiento
5. Ir a otra página web y repetir el proceso (crawling)

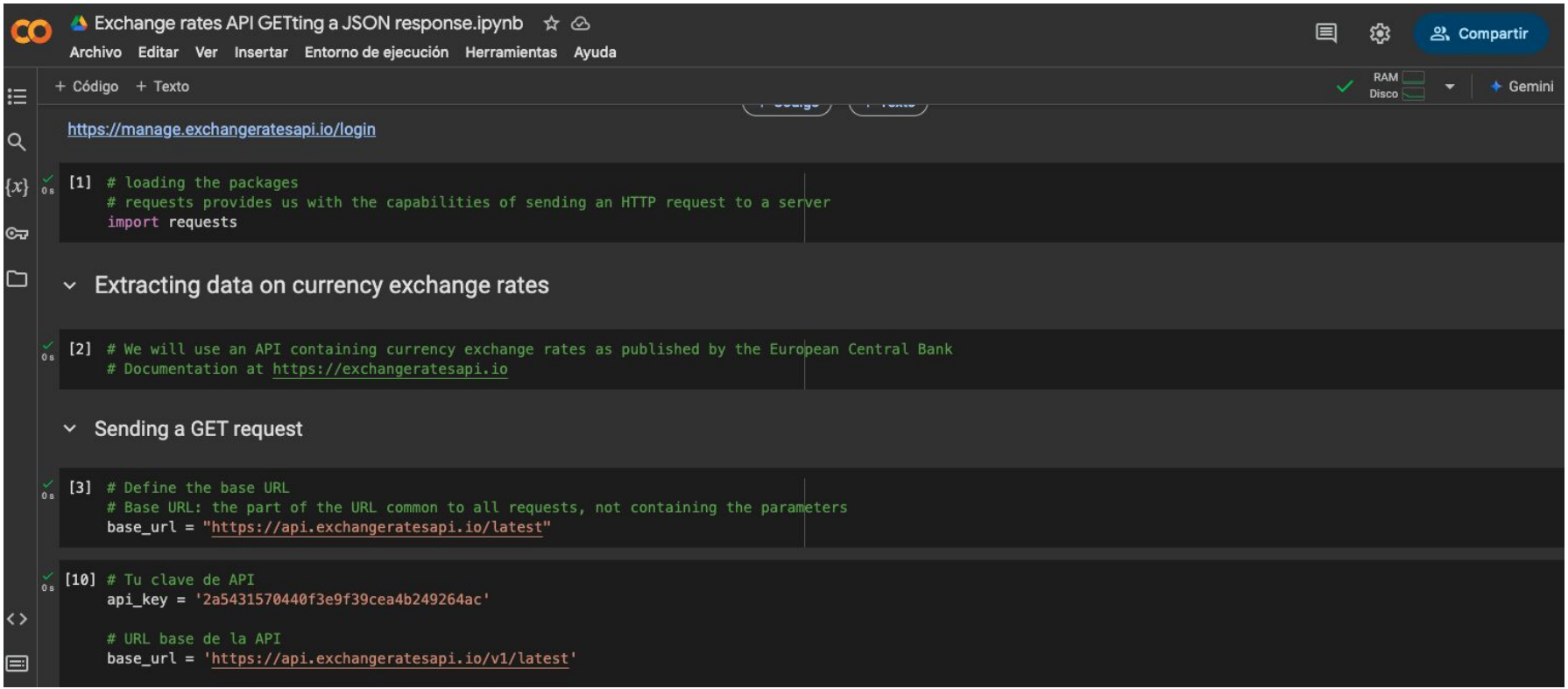
fuente: Sesión web scraping Carles Illa (2018)

Almacenamiento y conexión de datos

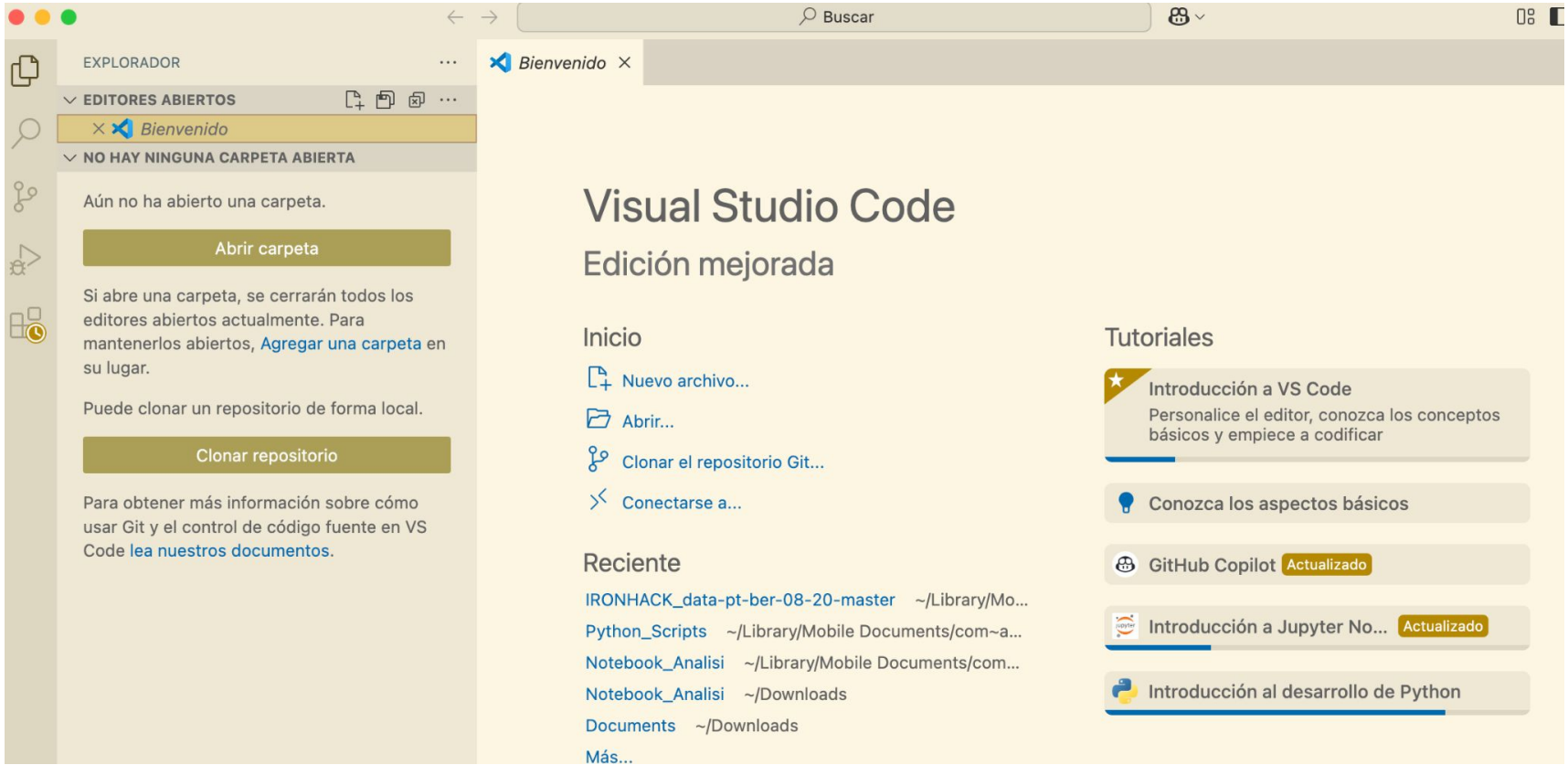
1. Formatos de almacenamiento
    - CSV (Comma Separated Values) para datos de texto.
    - JSON (JavaScript Object Notation) para estructuras complejas.

2. Creación de una API: Facilitar el acceso a los datos.
3. Repositorios de datos: Kaggle, UCI Machine Learning Repository, Github, Data World

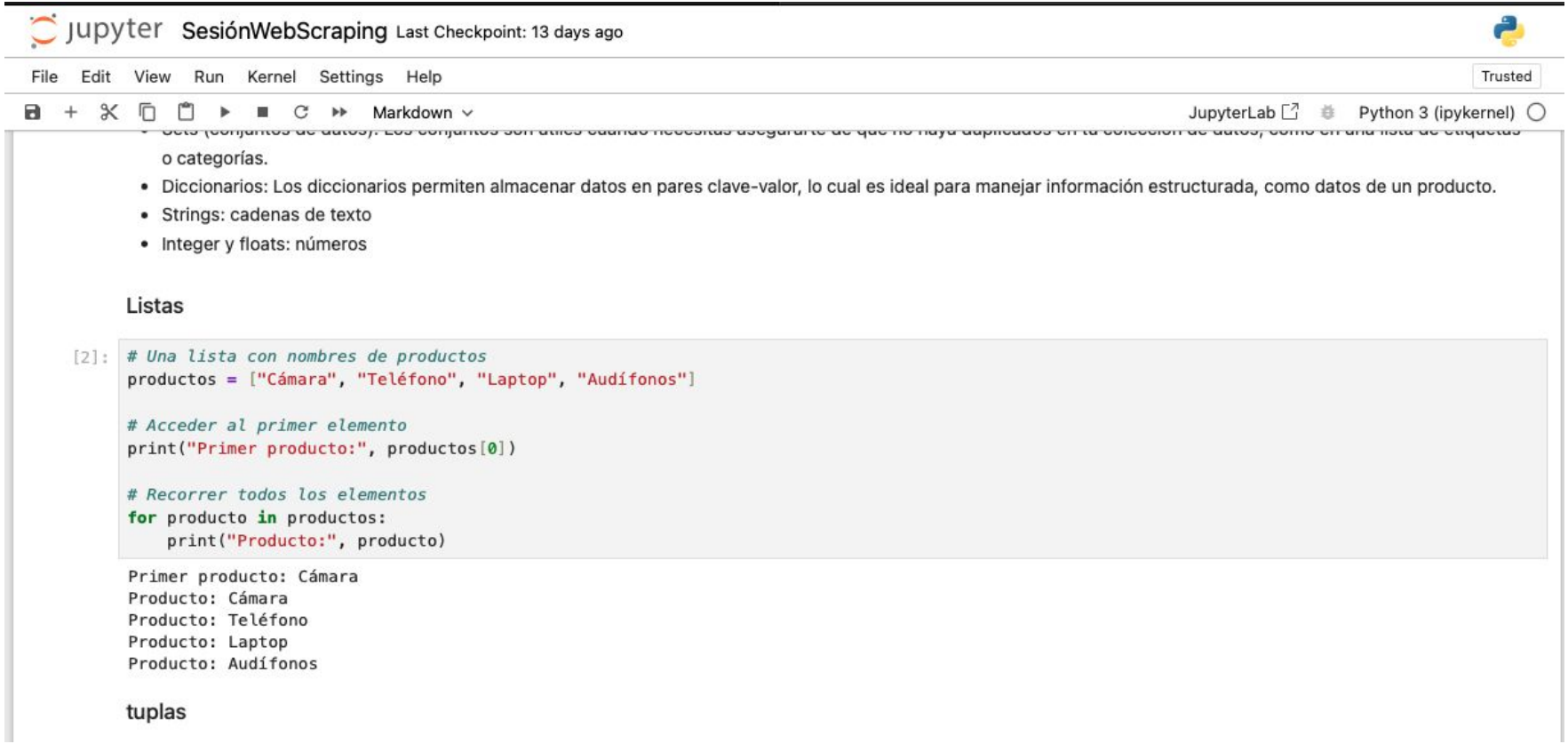
Google Colab



Visual Studio Code



Jupyter Notebook



Comandos terminal:

`python --version` → Muestra la versión de Python instalada.

`conda list` → Muestra los paquetes instalados en el entorno activo.

`pip list` → Muestra una lista de paquetes instalados con `pip`.

`pip install nombre_paquete` → Instala un paquete con `pip`.

`pip install --upgrade pip` → Actualiza un paquete con `pip`.

`pip freeze > requirements.txt` → Guarda la lista de paquetes instalados en un archivo `requirements.txt`.

`pip install -r requirements.txt` → Instala paquetes desde un archivo `requirements.txt`.

`which python` o **`where python`** (en Windows) → Muestra la ruta del intérprete de Python activo.

`which jupyter` → Muestra la ruta donde está instalado Jupyter.

`conda clean --all`

Trucos/shortcuts:

`Ctrl + Enter` → Ejecuta la celda sin moverse a la siguiente.

`Alt + Enter` → Ejecuta la celda y crea una nueva debajo.

`A` → Inserta una celda arriba.

`B` → Inserta una celda abajo.

`M` → Convierte la celda en Markdown.

`Y` → Convierte la celda en código.

`D D` → Elimina la celda seleccionada.

`Z` → Deshace la eliminación de una celda.

`Shift + M` → Fusiona la celda actual con la siguiente.

`Ctrl + S` → Guarda el notebook.

Tipos de estructuras de datos para manejar la información

- Listas: son estructuras de datos en Python que permiten almacenar múltiples elementos en un orden específico. Son útiles para manejar datos como listas de productos, títulos de artículos, o enlaces.
- Tuplas: Las tuplas son como listas, pero no se pueden modificar después de su creación. Son útiles para representar datos que no deben cambiar, como coordenadas o fechas.
- Sets (conjuntos de datos): Los conjuntos son útiles cuando necesitas asegurarte de que no haya duplicados en tu colección de datos, como en una lista de etiquetas o categorías.
- Diccionarios: Los diccionarios permiten almacenar datos en pares clave-valor, lo cual es ideal para manejar información estructurada, como datos de un producto.
- Strings: cadenas de texto.
- Integer y floats: números enteros y decimales.

Recursos

Kaggle: https://www.kaggle.com/datasets

Repositorios github:

https://github.com/alabarga/pandas-workshop

https://github.com/alabarga/DataScienceWorkshop-BS/tree/master

https://github.com/Pierian-Data/Complete-Python-3-Bootcamp

https://www.w3schools.com/python/default.asp