Universidad Gerardo Barrios

Comisión de Acreditación de la Calidad de la Educación Superior
UNIVERSIDAD GERARDO BARRIOS (UGB)
ACREDITADA
2024-2029

PARCIAL II, C2

ASIGNATURA:

PROGRAMACIÓN COMPUTACIONAL III

DOCENTE:

WILIAN ALEXIS MONTES GIRÓN

ESTUDIANTES:

CARMEN XIOMARA HERNANDEZ CASTILLO

CESIA MADAI AVALOS DIAZ

FECHA DE ENTREGA

24 DE OCTUBRE DE 2025

Contenido

Introducción

La librería requests es una de las herramientas más utilizadas en Python para realizar solicitudes HTTP de manera sencilla, eficiente y legible. Su objetivo principal es facilitar la comunicación entre aplicaciones y servidores web, permitiendo enviar y recibir datos a través del protocolo HTTP (HyperText Transfer Protocol), el mismo que utilizan los navegadores para acceder a sitios web.

Antes de la existencia de requets, los programadores debían utilizar módulos integrados como urllib o http.client, los cuales eran más complejos y menos intuitivos. La librería requets, creada por Kenenth Reitz en 2011, simplificó este proceso al ofrecer una interfaz amigable y de alto nivel, basada en el principio de que “hacer cosas simples debe ser simple y las complejas deben ser posibles”.

Con requets, es posible realizar diferentes tipos de solicitudes, tales como GET o POST, y otras, para interactuar con APIs (Interfaces de Programación de aplicaciones) , sitios web o servicios en línea.

¿En qué consiste la librería y para qué se usa?

La librería requests consiste en un conjunto de herramientas que permiten realizar solicitudes HTTP desde Python de forma sencilla, legible y eficiente. Su función principal es facilitar la comunicación entre aplicaciones y servidores web, permitiendo enviar y recibir información sin necesidad de un navegador.

Se usa principalmente para interactuar con APIs, descargar o enviar datos, automatizar procesos en línea y probar servicios web. Gracias a su diseño intuitivo, requests es ideal tanto para principiantes como para desarrolladores avanzados que buscan integrar servicios externos dentro de sus programas.

Funciones más relevantes y utilizadas de la librería

A continuación, se describen algunas de las funciones y métodos más usados de la librería requests:

1. **requests.get(url, params=None, **kwargs)**

Permite obtener datos de un servidor mediante una solicitud HTTP GET.

Es la forma más común de acceder a información pública en una API.

respuesta = requests.get("https://api.github.com/users")

2. **requests.post(url, data=None, json=None, **kwargs)**

Se utiliza para enviar datos al servidor, por ejemplo, al enviar un formulario o crear un nuevo registro.

datos = {'usuario': 'Madai', 'clave': '1234'}

respuesta = requests.post("https://example.com/login", data=datos)

3. **requests.put(url, data=None, **kwargs)**

Permite actualizar información existente en el servidor.

actualizar = {'nombre': 'Madai Avalos'}

requests.put("https://api.ejemplo.com/usuario/1", data=actualizar)

4. **requests.delete(url, **kwargs)**

Envía una solicitud HTTP DELETE para eliminar un recurso en el servidor.

requests.delete("https://api.ejemplo.com/usuario/1")

5. **response.json()**

Convierte automáticamente el contenido de una respuesta en formato JSON a un diccionario de Python, facilitando su manejo.

respuesta = requests.get("https://api.github.com")

datos = respuesta.json()

print(datos)

¿Cómo usamos Requets en la vida real?

1. **Conexión con APIs públicas**
   Se utiliza para obtener información de servicios como **OpenWeatherMap** (clima), **NASA**, **GitHub** o **Twitter**, consumiendo sus datos en tiempo real.

2. **Automatización de tareas web**
   Permite descargar archivos, llenar formularios o recopilar datos de distintas páginas sin intervención manual.

3. **Pruebas de sistemas y monitoreo**
   Los desarrolladores usan `requests` para comprobar si un sitio o una API responde correctamente, verificando códigos de estado como 200 (éxito) o 404 (no encontrado).

4. **Integración entre aplicaciones**
   Facilita la comunicación entre diferentes sistemas, por ejemplo, una aplicación interna que necesita enviar información a un servicio externo.

5. **Análisis de datos y aprendizaje automático**
   Es común en proyectos de **data science**, donde se descargan conjuntos de datos desde fuentes en línea antes de analizarlos con librerías como **pandas** o **NumPy**.

Conclusión

La librería `requests` representa uno de los avances más importantes dentro del ecosistema de Python en cuanto a conectividad y comunicación con servicios web. Su diseño simple, legible y a la vez poderoso ha permitido que millones de desarrolladores puedan interactuar con la web de forma directa, sin tener que enfrentarse a la complejidad de los protocolos de red tradicionales. Gracias a su sintaxis clara y a la amplia variedad de funciones que ofrece, `requests` se ha convertido en una herramienta esencial tanto para proyectos personales como profesionales.

En el contexto actual, donde el intercambio de información a través de Internet es parte fundamental de casi todos los sistemas digitales, el dominio de esta librería resulta indispensable. Permite acceder a datos en tiempo real, consumir APIs de grandes plataformas, automatizar procesos en línea y conectar múltiples aplicaciones en un entorno cada vez más interconectado. Además, su compatibilidad con formatos modernos como JSON y su soporte para la autenticación y seguridad mediante HTTPS garantizan que las comunicaciones sean confiables y seguras.

La versatilidad de `requests` la ha consolidado como una pieza clave en ámbitos como el desarrollo web, la ciencia de datos, la inteligencia artificial y la integración de servicios empresariales. Su popularidad no solo radica en su facilidad de uso, sino también en su estabilidad y la gran comunidad que la respalda, lo que asegura su constante actualización y mejora.

En conclusión, la librería `requests` no solo simplifica el trabajo del programador, sino que también amplía las posibilidades de lo que puede lograrse con Python en el mundo digital. Comprender y aplicar sus capacidades abre las puertas a un sinfín de oportunidades tecnológicas, desde el desarrollo de aplicaciones inteligentes hasta la construcción de sistemas conectados globalmente.