**Free IP Geolocation API**

* **AUTORES: Álvaro Cañada Lorenzo y Enrique Sánchez Vicente 2º DAM.**
* **URL**: <https://freegeoip.app/>
* **Descripción de la API con vuestras palabras. ¿Qué información ofrece?**

**Free IP Geolocation** es una **API de gelocalización de IP gratuita** para desarrolladores que hace uso de una base de datos de direcciones IP que están asociadas a ciudades, además de la capacidad ofrecer al usuario información relevante como la zona horaria, la latitud y la longitud. Está basada en JSON que permitiendo a los desarrolladores recuperar datos de manera fácil sobre una dirección IP. Permite realizar 15 000 solicitudes por hora a través de su apikey.

* **Breve descripción de uso de la API. ¿Cómo se utiliza?**

**String apikey = "XXXXXXXXXXXXXX";**

**String url = "https://api.freegeoip.app/json/8.8.8.8?apikey=" + apikey;**

Se pasa como parámetro implícito en la URL, la IP que se quiere buscar, en este caso 8.8.8.8.

Se pasa como parámetro de la consulta, la key de la api.

El JSON de retorno sería de esta forma:

**{**

**"ip": "90.161.180.197",**

**"country\_code": "ES",**

**"country\_name": "Spain",**

**"region\_code": "MD",**

**"region\_name": "Madrid",**

**"city": "Madrid",**

**"zip\_code": "28046",**

**"time\_zone": "Europe/Madrid",**

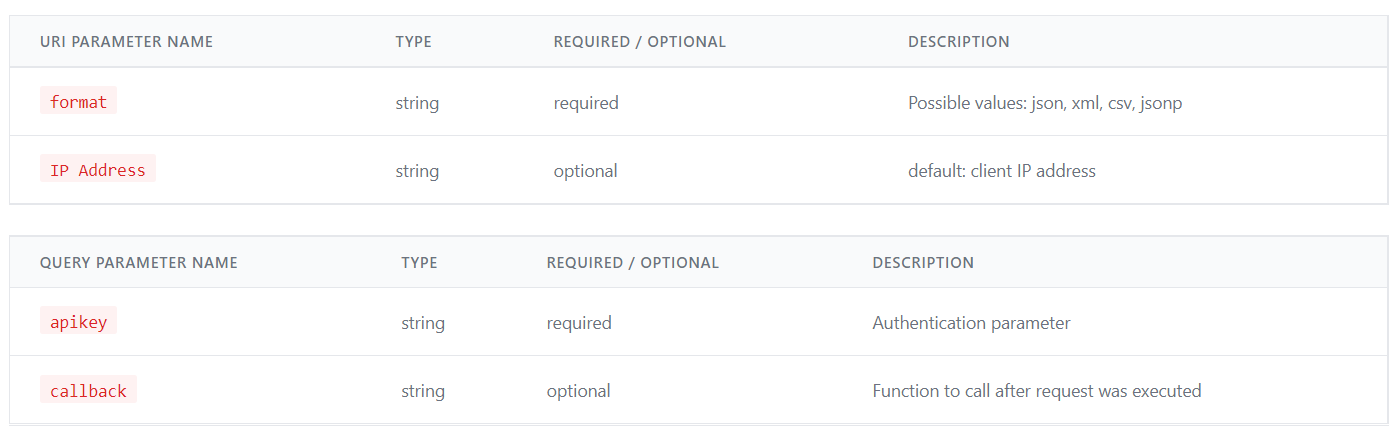
**"latitude": 40.4163,**

**"longitude": -3.6934,**

**"metro\_code": 0**

**}**

**Lista del total de parámetros que acepta la consulta:**

****

* **¿Qué va a hacer vuestra APP? Mínimo debe tener 2 servicios diferentes. Describe su uso.**

Cómo prototipo de una App dedicada a un RedTeam de ciberseguridad nuestra app ofrecerá varios servicios:

**SERVICIO 1:**

* Nuestra app accederá a la IP del usuario, y consultara a la api su ubicación para mostrársela.

**SERVICIO 2:**

* Además tendrá la opción de realizar un ataque DDOS a una IP personalizada desde su app y mostrarle el resultado de los datos más relevantes como geo localización, código postal, y número de peticiones por segundo lograda a realizar.

**CODIFICACIÓN: Uso de retrofit.**

**DISEÑO DE IMAGEN PARA EL LAYOUT DE NUESTRA APP PARA PIXEL 4:**



**Para este ejercicio usaremos la librería de Retrofit2, por lo que debemos realizar una configuración previa.**

* Primero incluimos las implementaciones de librerías en el fichero build.gradle, en el apartado dependencies.

implementation 'com.squareup.retrofit2:retrofit:2.9.0'  
implementation 'com.squareup.retrofit2:converter-gson:2.9.0'  
implementation 'com.google.code.gson:gson:2.8.9'

* Segundo creamos una clase API, en la cual vamos a configurar la conexión con el servidor.

En la clase API, incluimos la constante **BASE\_URL = “https://URL\_DE\_LA\_API . com/”;**

En la **clase API,** realizamos un patrón singleton para que solo se pueda crear una sola instancia de un objeto retrofit en la app de esta forma:

//Singleton  
private static Retrofit *retrofit*=null;  
  
public static Retrofit getAPI(){  
 if(*retrofit*==null){  
 *retrofit* = new Retrofit.Builder().baseUrl(*BASE\_URL*).addConverterFactory(GsonConverterFactory.*create*()).build();  
 }  
 return *retrofit*;  
}

* A continuación, creamos una interfaz **IPServices**, en la cual configuraremos nuestro envío a la API según los datos que queramos recibir. En esta interfaz agruparemos los servicios que vamos a utilizar.

//Para obtener informacion de una IP especifica.  
@GET("AQUI VA LA IP TARGET")  
Call<Atacado> getInfo(@Query("apikey") String apikey);

**PARA EVITAR FIREWALL DE COMPAÑIAS TELEFÓNICAS:**

Usaremos un proxy-list para variar de servidor y dirección IP y así evitar ser detectados por los proveedores de servicio.

Usaremos los proxy-list gratuitos: <https://free-proxy-list.net/>

(Estos se comprueban y actualizan cada 10 minutos).

Ejemplo:

**URLConnection.openConnection():**

**URL weburl = new URL (URL\_STRING);**

**Proxy webProxy = nuevo Proxy (Proxy.Type.HTTP, new InetSocketAddress (“127.0.0.1”, 8080));**

**HttpURLConnection webProxyConnection = (HttpURLConnection) weburl.openConnection (webProxy);**