# ¿Qué es OAuth 2.0?

OAuth 2.0 es un protocolo de autorización que permite que aplicaciones de terceros accedan a recursos protegidos en nombre de un usuario, sin que la aplicación tenga acceso directo a las credenciales del usuario. Es ampliamente utilizado para delegar acceso seguro en aplicaciones web, móviles y APIs.

## Principales Conceptos de OAuth 2.0

### Resource Owner (Propietario del Recurso):

Es el usuario que otorga acceso a sus recursos protegidos (como datos de perfil o fotos).

### Client (Cliente):

Es la aplicación que solicita acceso a los recursos del usuario. Puede ser una aplicación web, móvil o servidor backend.

### Authorization Server (Servidor de Autorización):

Es el servidor que autentica al usuario y emite tokens de acceso al cliente.

### Resource Server (Servidor de Recursos):

Es el servidor que aloja los recursos protegidos (como una API). Valida los tokens y responde a las solicitudes del cliente.

## Flujos de OAuth 2.0

OAuth 2.0 tiene diferentes flujos (o 'grants') que varían según el tipo de cliente y la seguridad requerida. Los principales son:

* Authorization Code Grant: Uso típico en aplicaciones web con servidor backend. Involucra un código de autorización que se intercambia por un token de acceso.  
  Implicit Grant: Uso típico en aplicaciones SPA (Single Page Applications) sin servidor backend. El token se emite directamente.  
  Client Credentials Grant: Uso típico para comunicación entre servicios o aplicaciones sin usuarios finales. El cliente solicita un token utilizando sus propias credenciales.  
  Password Grant (Password Credentials): Uso típico en aplicaciones móviles o clientes de confianza. El cliente recibe el token tras proporcionar directamente las credenciales del usuario al servidor.

## Tokens en OAuth 2.0

### Access Token (Token de Acceso):

Es un identificador que el cliente usa para acceder a los recursos protegidos. Tiene una vida útil limitada.

### Refresh Token (Token de Renovación):

Se usa para obtener nuevos tokens de acceso sin requerir que el usuario vuelva a autenticarse.

## Seguridad en OAuth 2.0

Para garantizar la seguridad en el uso de OAuth 2.0, se deben seguir buenas prácticas como el uso obligatorio de HTTPS, la configuración de scopes para definir permisos específicos, y la emisión de tokens con corta duración.

## Ejemplo Básico de Flujo Authorization Code

1. El usuario quiere permitir que una aplicación acceda a sus fotos alojadas en un servidor.  
2. La aplicación redirige al usuario al Servidor de Autorización.  
3. El usuario inicia sesión y concede acceso.  
4. El Servidor de Autorización redirige de nuevo al cliente con un código de autorización.  
5. La aplicación intercambia el código por un token de acceso.  
6. La aplicación usa el token de acceso para solicitar recursos protegidos.