DWES - OAuth

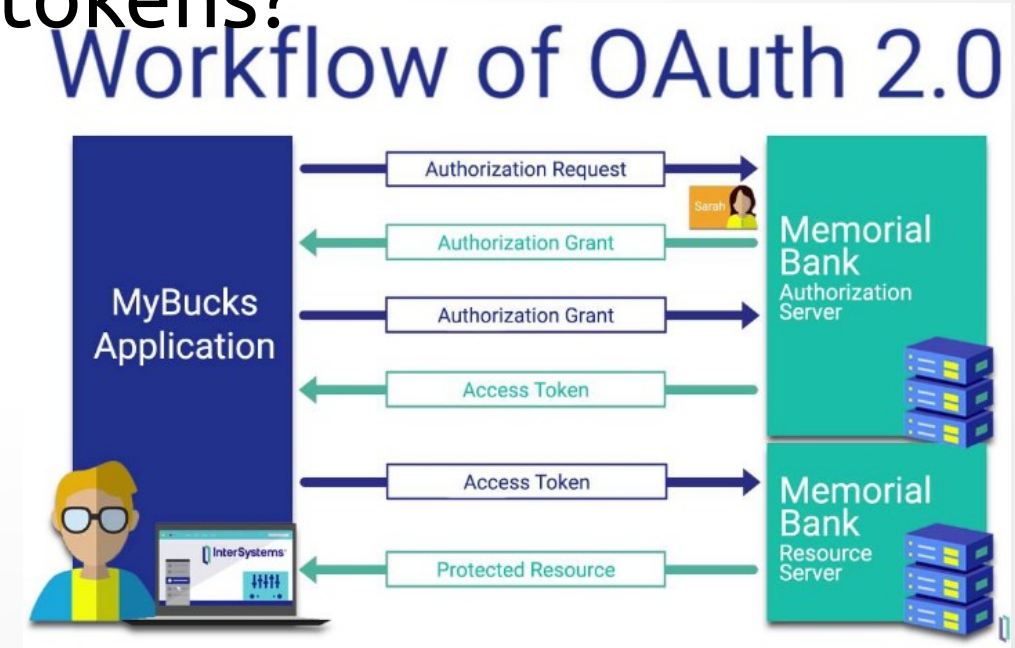
Jorge Dueñas Lerín
Profesor Formación Profesional
Comunidad Madrid

Índice

- Presentación OAuth
-

OAuth

- ¿Qué es?
  - Un estándar para autorización como servicio
- ¿Qué hace?
  - Define un protocolo de intercambio de tokens
- ¿Para qué sirven estos tokens?
  - Podemos acceder a recursos y ejecutar acciones



OAuth

- Sí, muy bien pero… ¿Para qué sirve?
  - List of OAuth providers
  - https://en.wikipedia.org/wiki/List_of_OAuth_providers

OAuth

- Situaciones
  - Quiero ver los amigos de un usuario en Facebook
  - Quiero publicar tweets a una determinada hora en nombre de un usuario
  - Quiero subir un vídeo a la cuenta de Youtube de un usuario desde mi app móvil
  - Quiero autentificar a mi usuario con algúna plataforma conocida: Google, Twitter, Facebook, etc.

OAuth

- ¿Qué ventajas tiene todo esto?
  - Tu contraseña no viaja a ninguna parte
  - Gran granularidad en cuanto a las acciones y la información
    - Puede hacer esto y esto pero no esto
    - Puedes acceder a mi email pero no a mis amigos
    - Etc.
  - Posibilidad de revocar acceso

Actores

- Actores - **OAuth**
  - Propietario del recurso
  - Cliente o aplicación
    - Ojo!
  - Servidor de autorización
    - Google, Facebook, Linkin, Github
  - Servidor de recursos
    - Dropbox, Drive, Otra app, etc.

Pueden ser
el mismo elemento

Actores

- Actores
  - Usuario
  - Aplicación
    - Móvil
    - SPA (Javascript)
    - Aplicación web
    - Proceso automático
  - Servidores
    - Autentificación
    - Recurso
    - Nuestra aplicación

- 4 Flujos de trabajo
  - Depende de la situación en la que estemos
  - Los nombres no son muy significativos
  - Antes de usar un flujo debemos dar de alta nuestra aplicación.

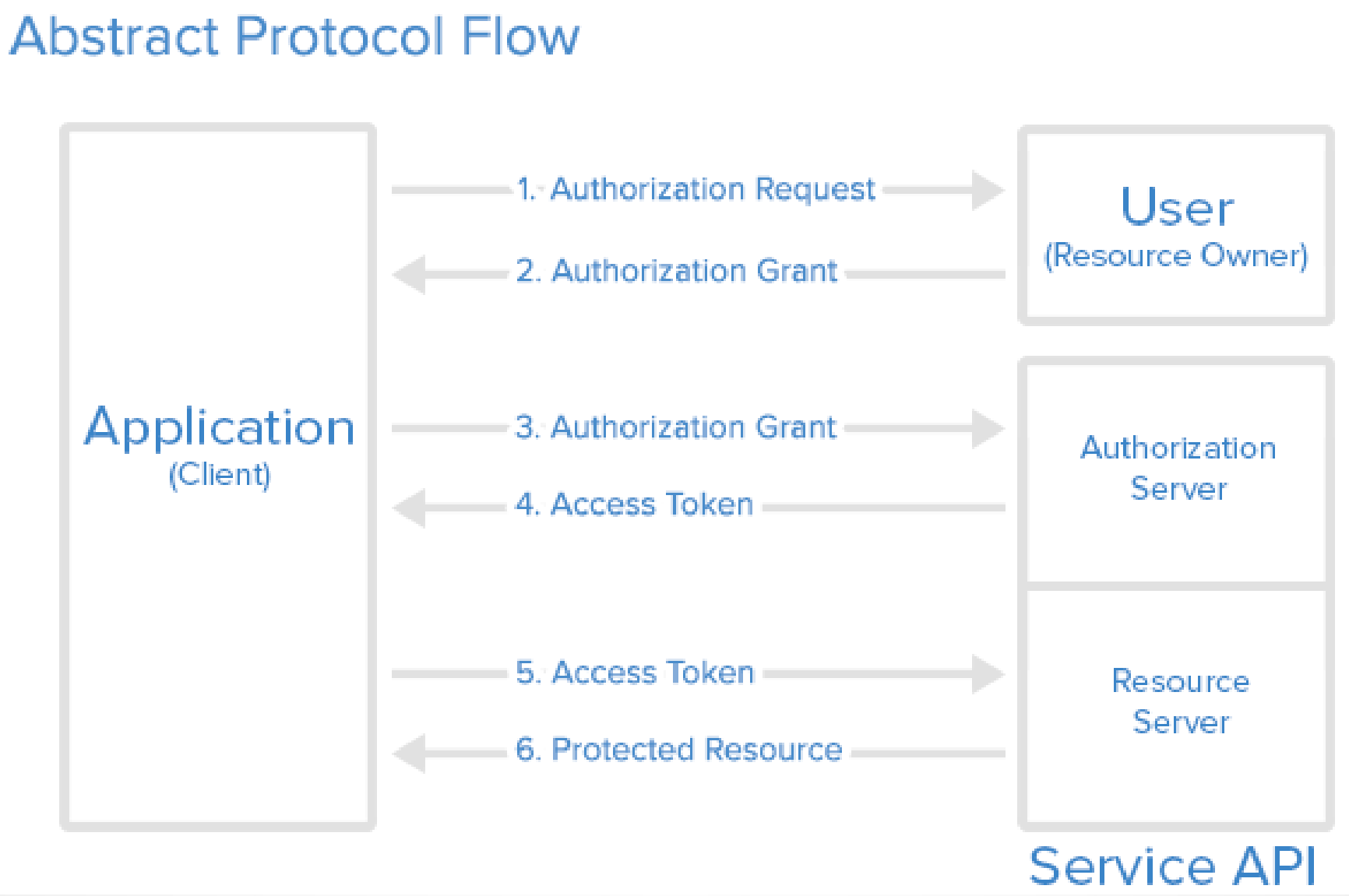
Canales de comunicación

- Dentro del estándar no se especifica cómo se envían los tokens
  - Método GET
    - Parámetros y token en la URL
  - Método POST
    - Datos de la petición
  - Cualquiera
    - Cabecera de procolo
      GET /algun/lugar/web/ HTTP/1.1
      Host: asdasdas.es
      ...
      Autentification: <ALGO_TOKEN>

Grant Access

- Tipo de autentificación - flujos
  - Credenciales de cliente
    - Este cliente es el termino Oauth
    - Acceso al API
  - Código de autorización
    - Servidor web, **cliente** web (JS), **cliente** app
      - OJO! con esta palabra cliente
  - Credenciales de propietario – Password
  - Implícito

Flujo genérico




Start
Is the Application the Resource Owner?
YES
Client Credentials Grant
NO
Is the Application a web app executing on the server?
YES
Authorization Code Grant
NO
Is the Application absolutely trusted with user credentials?
YES
Resource Owner Password Credentials Grant
NO
Is the Application a native app or a SPA?
SPA
Implicit Grant
NATIVE APP
Authorization Code Grant (PKCE)

Práctica

- Práctica:
  - Acceso al API desde una aplicación (Client credentials)
    - Proceso CRON
  - Acceso a información de un usuario (Authorization code)
- Vamos a
  - Montar un servidor Oauth
  - Usarlo desde clientes
  - Gestión de granjas:
    - Plataforma para supervisar las gallinas.

Entorno

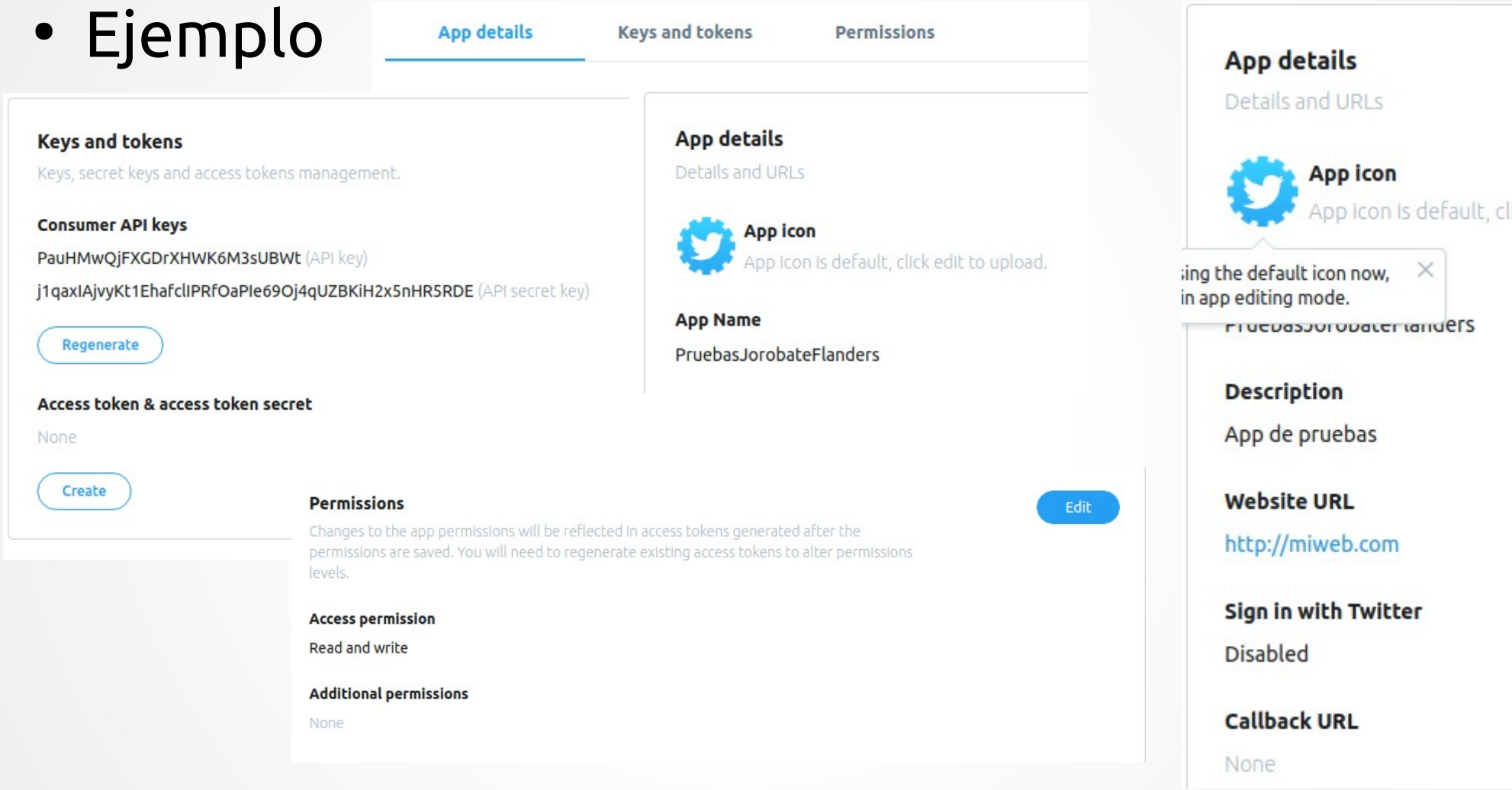
- Preparando el entorno
  - Instala en tu servidor composer:
    - Es un gestor de paquetes/librerías php
    - https://getcomposer.org/doc/00-intro.md
    - Sigue las instrucciones y ejecuta el comando composer, deberá funcionar y aparecer un listado de acciones
  - Descargar el código de los ejemplos
    - Colócalo en tu directorio web y configura los permisos

Primeros pasos

- Primero
  - Dar de alta mi cliente para usar un API
    - OJO! **Cliente**
  - No se especifica dentro del protocolo cómo hacer esto ni qué información pedir:
    - Nombre de la app, organización, URL de comunicación, etc.
  - Al terminar lo básico
    - CLIENT_ID (Elemento público)
    - CLIENT_SECRET (Elemento privado, máxima seguridad)
    - URL_CALLBACK (Redirección de comunicación)

Twitter

- Ejemplo

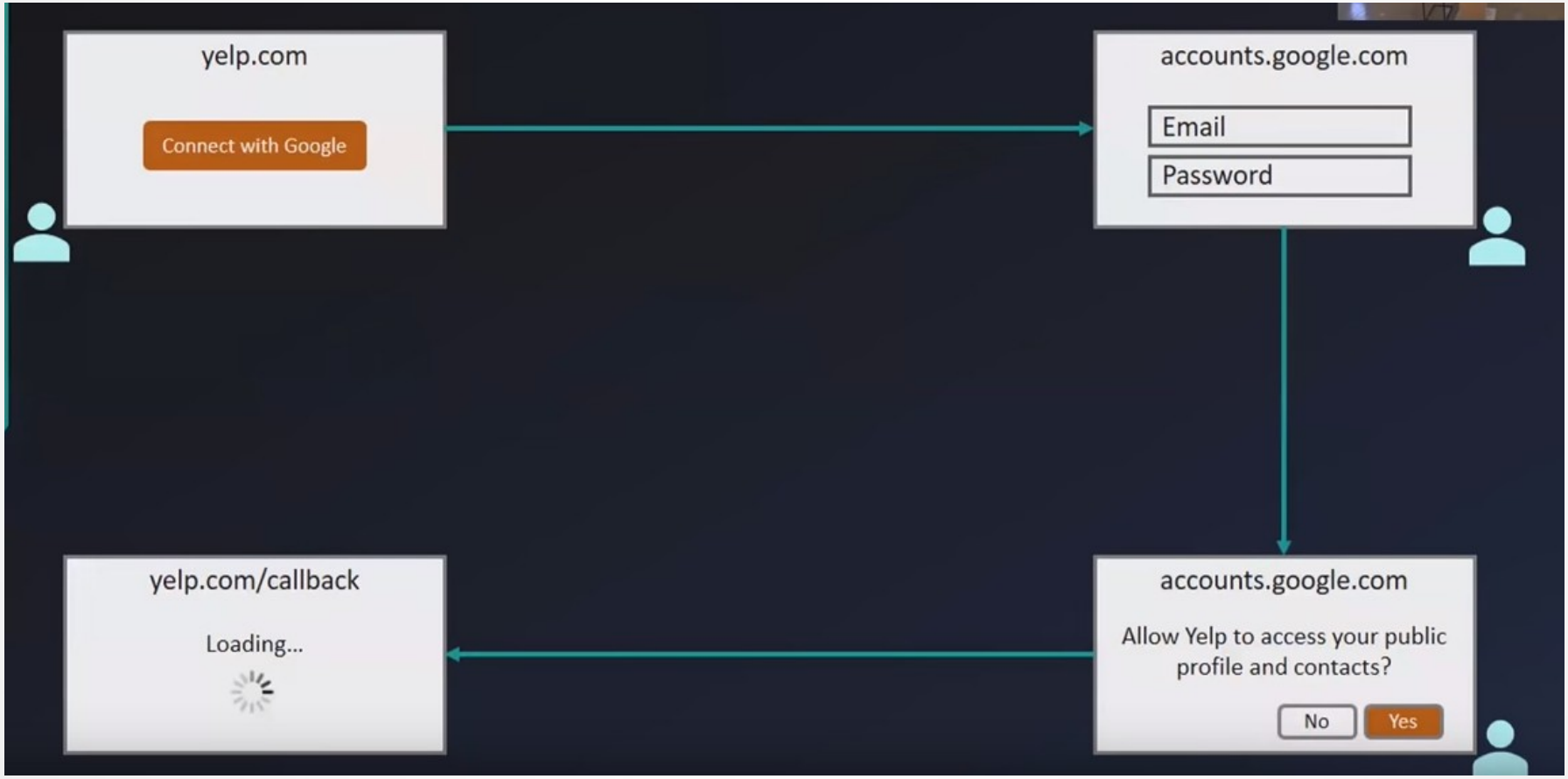


Tareas

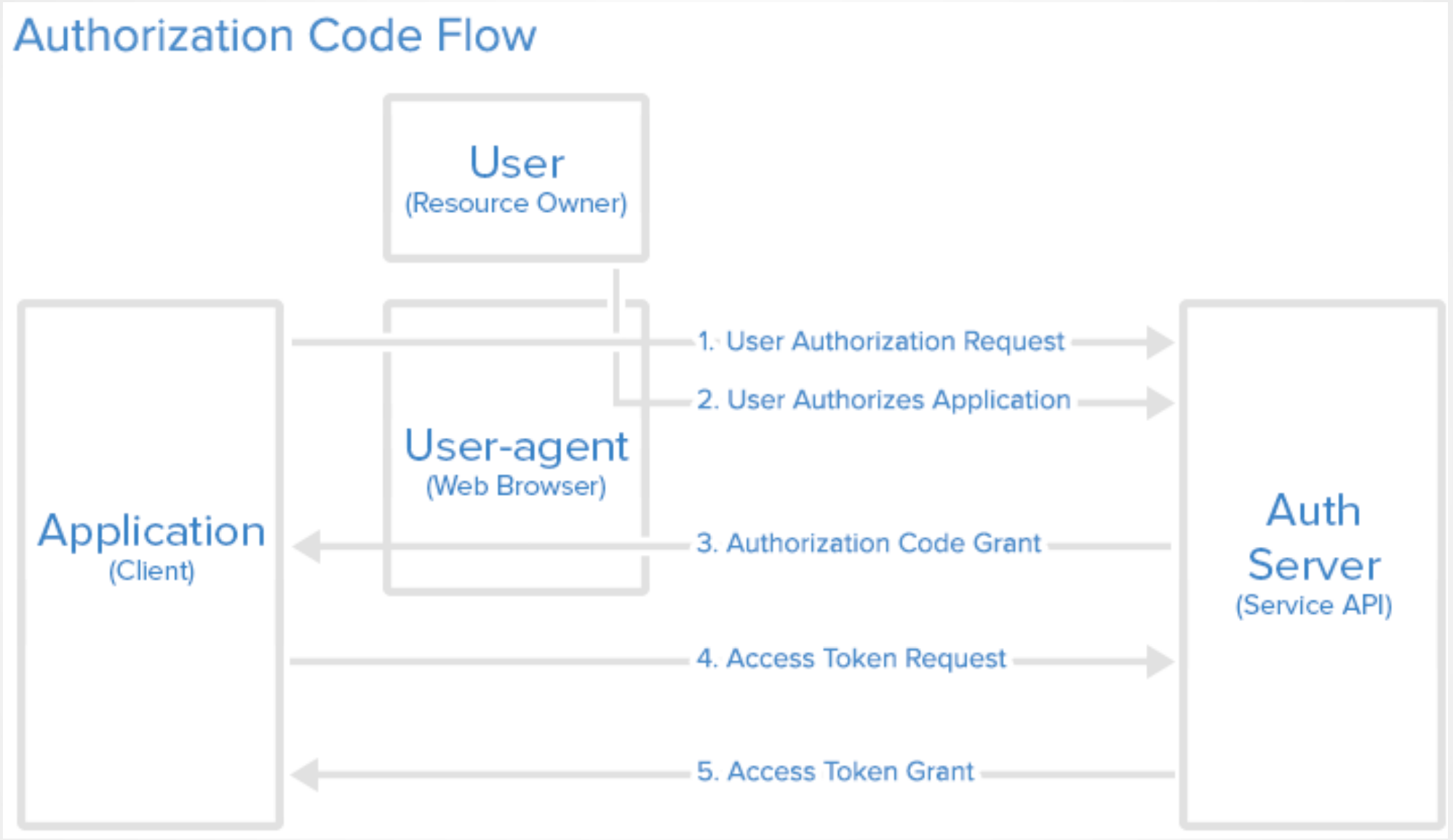
- Hacer peticiones y entender el acceso con client credentials
  - Explicación

Siguiente flujo

- Hacer web para usar auth code



Auth code


Authorization Code Flow
User
(Resource Owner)
User-agent
(Web Browser)
Application
(Client)
Auth
Server
(Service API)
1. User Authorization Request
2. User Authorizes Application
3. Authorization Code Grant
4. Access Token Request
5. Access Token Grant

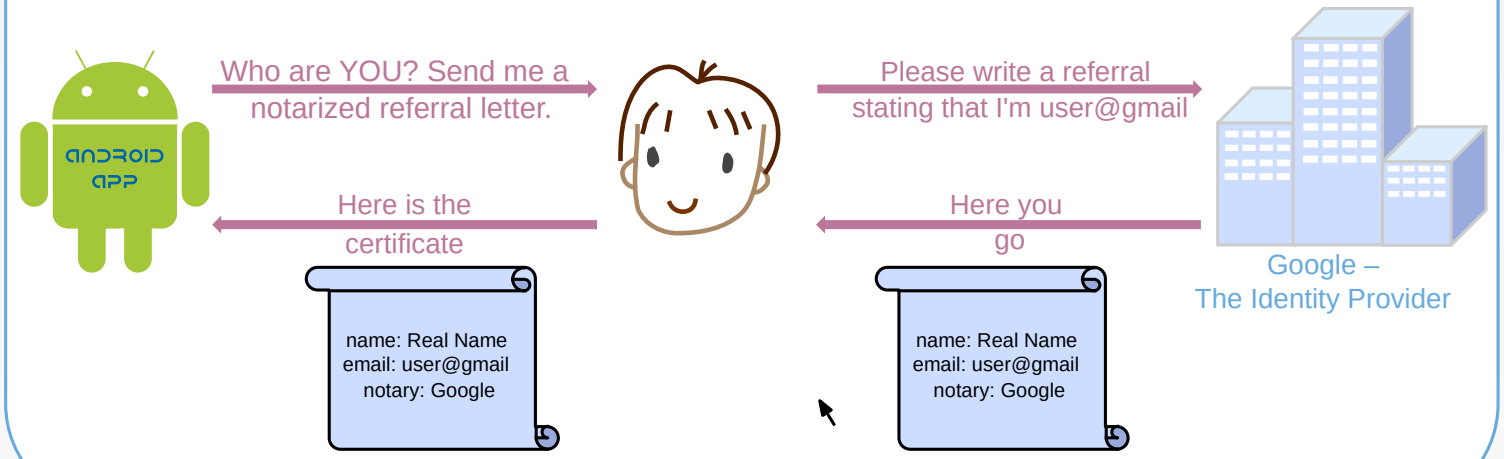
Auth code

- ¿Qué hacemos con la información?
  - {
    "access_token":"...",
    "expires_in":86400,
    "token_type":"Bearer",
    "scope":"eggs-count profile",
    "refresh_token":"…"
    }
  - Dependiendo de la duración de estos token nos merecerá la pena guardarlos en la base de datos
  - Cuando visitamos la página de autorización, si el usuario ya ha autorizado la app no vuelve a pedirle permiso.

Autentificación

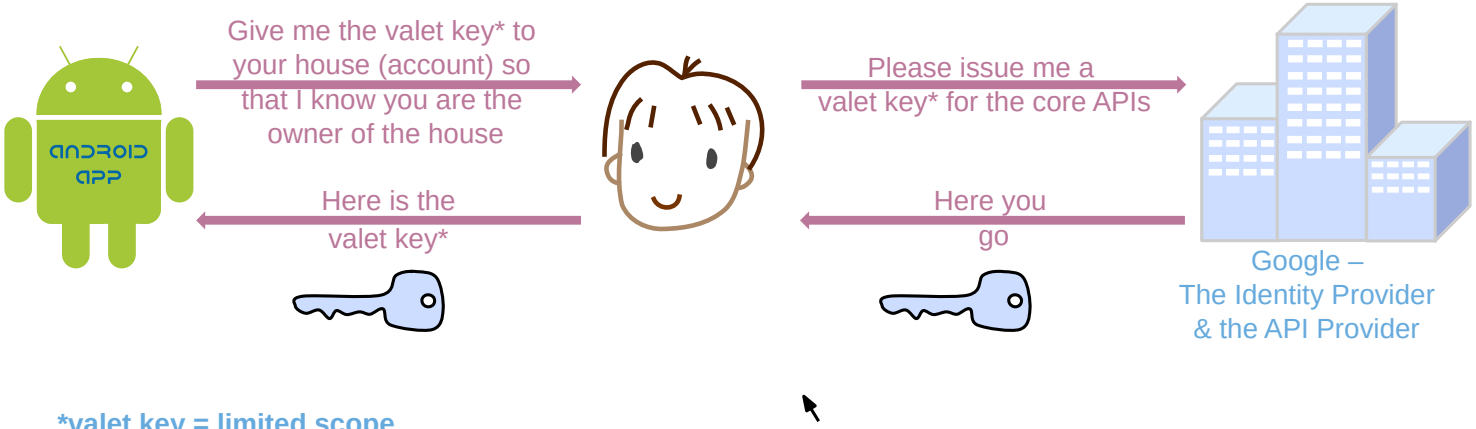
- Esto vale para consultar información pero… ¿Y para loguear a un usuario?
  - Si podemos acceder a su perfil consideramos que está logueado
  - Es distinto a OpenID pero nos sirve

OpenID Authentication



vs.

Pseudo-Authentication using OAuth



***valet key = limited scope OAuth Token**

adapted from a drawing by @_nat_en

- Para autentificar con OAuth debemos poder tener “usuarios” solo con el acceso a OAuth
- Usuarios
  id
  nombre
  email
  password ← ¿si un usuario no tiene password?
  token
  expiración
  token_renovacion

- ¿Cómo seguir?
  - Plantearse un reto en el mundo real
    - Acceder a los vídeos de un usuario de Youtube
    - Publicar en el muro de Facebook de un usuario
    - Etc.
  - Buscar librerías que no ayuden
  - Implementar la funcionalidad
  - Realizar en el proyecto el login con entidades externas