ACCESO AL API DE FORTIMING - OAUTH 2

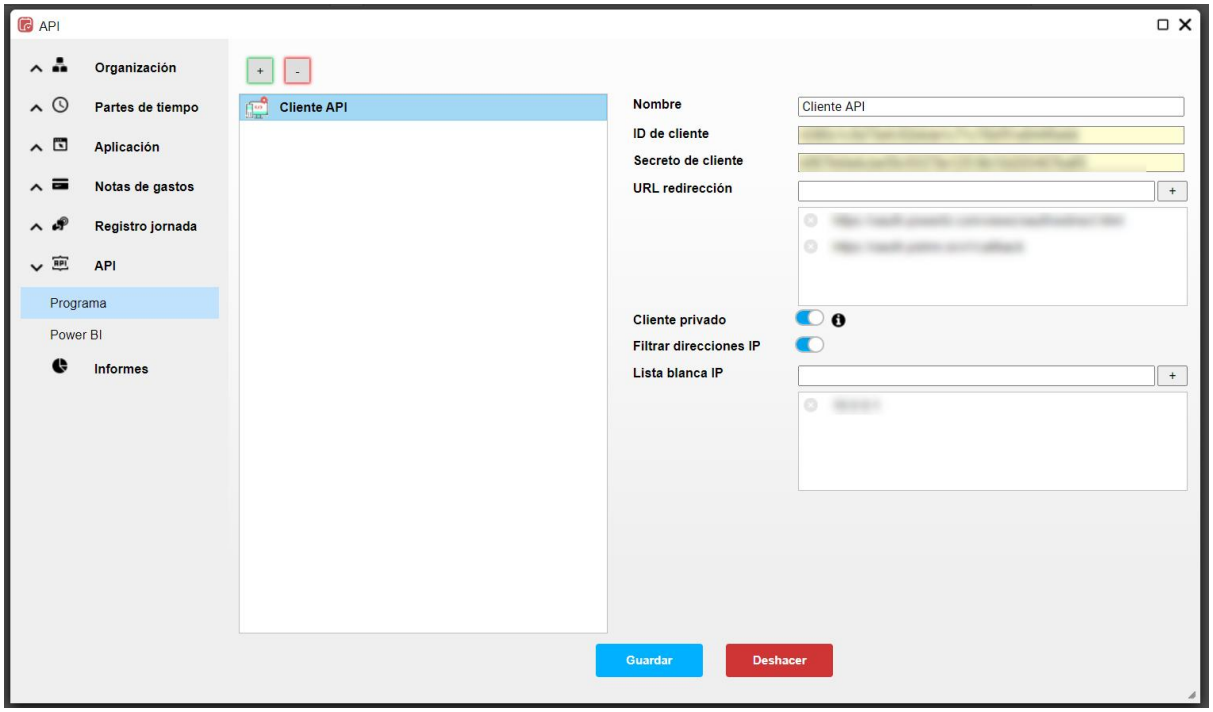
Introducción

El API pretende proporcionar acceso controlado a los datos gestionados por el servicio de ForTiming. Open Authorization 2 (OAuth 2 en adelante) es un estándar abierto que permite flujos simples de autorización para aplicaciones de escritorio, móviles, web y servidores.

Este estándar propone varias opciones a ajustar libremente por los desarrolladores según las necesidades del sistema. Este documento describe esas particularidades de implementación de OAuth 2 para proporcionar acceso al sistema ForTiming a través de su API.

Registro del programa cliente

Para poder obtener las credenciales de acceso siempre va a ser necesario registrar la aplicación que vaya a acceder. Se ha definido un módulo API que debe haber sido solicitado y activado previamente y que habilita el panel de administración de aplicaciones de terceros a aquellos usuarios que tengan el permiso correspondiente activado. Este panel únicamente está disponible en la versión web de la aplicación en la dirección https://gqc.fortiming.com/ForTimingWeb/app. Ver siguiente imagen.



Se describen a continuación los campos que aparecen.

- **Nombre**: Es un nombre descriptivo que se mostrará al usuario cuando se solicite su autorización.
- **ID de cliente**: Es un código público que identifica de manera única al programa. Es el valor que habrá que indicar cuando se pida el parámetro *client_id*. Se dice que es público porque se puede mostrar su valor a terceros sin riesgos de seguridad.
- **Secreto de cliente**: Es un código privado que se usa en algunos flujos de autorización y que conviene proteger. Por eso no conviene que se use en aplicaciones nativas ni en aplicaciones web de una sola carga (SPA - Single Page Applicaion). Es el valor que habrá que indicar cuando se pida el parámetro *client_secret*.
- **URL redirección**: Son las URI permitidas a las que se podrá mandar códigos de autorización y tokens. Es imprescindible registrar las que se vayan a usar como parte de la validación del programa.
- **Cliente privado:** Indica que el programa cliente es de uso privado y por tanto el secreto de cliente permanecerá seguro. Cuando se activa esta opción, se habilita el flujo 'Credenciales de cliente' para permitir la comunicación entre servidores (Machine to Machine). Permite generar sesiones sin un usuario vinculado.
- **Lista blanca de IP**: Son las direcciones IP desde las que se va permitir la conexión del programa registrado. Es opcional, si no se define ninguna dirección esta restricción quedará deshabilitada. No forma parte del estándar de OAuth 2.

FLUJOS DE VALIDACIÓN

El estándar OAuth 2 contemplan varios flujos de autorización de los cuales se han implementado los siguientes:

- **Código de autorización**
- **Implícito**
- **Implícito PKCE (Proof Key for Code Exchange)**
- **Credenciales de cliente**

A continuación, se describe detalladamente cómo aplicar cada flujo.

Flujo de validación usando un código de autorización

Este flujo es el más seguro y usado. Utiliza el *client_secret* por lo que es necesario garantizar que no se pueda obtener por terceros. Es por eso que este flujo requiere que el programa cliente disponga

de un servidor que proteja el secreto del programa cliente. Es habitual que este flujo se use en páginas web generadas dinámicamente por un servidor.

Para comenzar el proceso de autorización se debe hacer la siguiente llamada:

```
GET:https://gq.fortiming.com/ForTimingWeb/auth?response_type=code&client_id=CLIENT_ID&redirect_uri=REDIRECT_URI&scope=read_organizacion%20read_tiempos&state=89ABC
```

Descripción de la llamada y sus parámetros:

- Es una llamada **GET**
- **response_type** debe indicar el valor ***code*** para pedir un código de autorización.
- **client_id** es el identificador público de la aplicación. Se obtiene al registrar el programa.
- **redirect_uri** es la dirección a la que se redirigirá el 'User-Agent' (normalmente un navegador web) cuando el usuario autorice o no el acceso al programa. Esta dirección debe estar registrada por motivos de seguridad.
- **scope** son los niveles de acceso solicitados. En este caso se solicita la lectura de datos de la organización y la lectura de partes de tiempo. Coincide con el scope por defecto así que si no se indica se aplicará este automáticamente. Sin embargo si se indican valores incorrectos se producirá el error *unsupported_scope*.
- **state** es un parámetro opcional. Se recomienda que lo use el programa cliente para hacer un seguimiento del proceso. Suele ser un valor aleatorio que se usa para validar la respuesta del servidor de autorización ya que este mismo valor se mandará de vuelta en la respuesta de la autorización.

Se mostrará una web para autorizar el acceso al programa cliente. Una vez el usuario haya autorizado o no al programa, se redirigirá al '*User-Agent*' a la URI indicada en *redirect_uri*.

```
GET https://app-cliente.com/cb?code=AUTH_CODE_HERE&state=1234zyx
```

En esta redirección, si se ha autorizado el acceso, se mandarán en el *query string* el código de autorización y el *state* indicado en la llamada inicial. Si no se ha autorizado el acceso no vendrá el parámetro *cod*e.

El servidor que soporta la aplicación debe recoger estos parámetros y después de identificar y validar el *state* debe realizar la siguiente llamada al servidor de autorización para obtener un token de acceso.

```
POST:https://gq.fortiming.com/ForTimingWeb/token?grant_type=authorization_code&code=AUTH_CODE_HERE&redirect_uri=REDIRECT_URI&client_id=CLIENT_ID&client_secret=CLIENT_SECRET
```

Descripción de la llamada y sus parámetros:

- Es una llamada **POST**
- **grant_type** indica que se quiere autorizar un código para obtener un token de acceso al indicar el valor ***authorization_code.***
- **code** indica el código de autorización que nos han entregado en el paso anterior.
- **redirect_uri** deberá indicar la misma dirección usada para obtener el código de autorización. Solo se usa como elemento adicional de seguridad.
- **client_id** es el identificador público de programa.
- **client_secret** es el secreto del programa cliente obtenido al registrar la aplicación.

La respuesta que se espera del servidor es un objeto json.
Si la validación ha sido positiva el objeto json tendrá los siguientes valores:

```
HTTP/1.1 200 Ok
Content-Type: application/json;charset=UTF-8
{
	"access_token":"TOKEN",
	"token_type": "Bearer",
	"expires_in": 1600,
	"refresh_token": "REFRESH",
	"scope": "SCOPE",
	"url_base": "https://gq.fortiming.com",
	"user_info": {
				"id_personal": 0,
				"email": "user@mailserver.com"
			}
}
```

Descripción de las propiedades del objeto:

- **access_token** El token de acceso al API.
- **token_type** El tipo de token siempre va a ser *bearer* (token portadora).
- **expires_in** Tiempo de caducidad del token expresado en segundos.
- **refresh_token** Código de refresco de sesión para aumentar su tiempo de vida o si la sesión ya se ha cerrado generar nuevas sesiones.
- **scope** El nivel de acceso concedido. Podría no siempre coincidir con el solicitado.
- **url_base** Es la dirección base del servidor API y raíz de los endpoint. Esta URL puede variar entre distintas empresas.
- **user_info** Información sobre el usuario que ha realizado la autorización.

Si se ha producido algún error también se proporcionará un objeto json con información sobre el error.
El objeto tendrá los siguientes campos:

```
HTTP/1.1 400 Bad Request                    Content-Type:
application/json;charset=UTF-8
{
      "error":" ERROR",
      "error_description": "DESCRIPTION",
      "error_uri": "URI"
}
```

Descripción de las propiedades del objeto:

- **error** Indica el nombre del error.
- **error_description** Da una descripción más detallada del error.
- **error_uri** Es opcional y si se indica propone una dirección de ayuda para solucionar el error.

En este punto el programa cliente ya dispone del token y puede emplearlo para hacer llamadas al API. Para que el API reconozca el token se debe incluir en la cabecera de llamada de la siguiente manera:

```
curl --request GET \
      --url https://gq.fortimiming/ForTimingWeb/api/departamentos \
      --header 'authorization: Bearer ACCESS_TOKEN' \
      --header 'content-type: application/json'
```

Flujo de validación implícito

Se denomina implícito porque se obtiene el token de acceso directamente tras autorizar el acceso al programa.

Este flujo de validación está pensado para programas nativos y aplicaciones web que no pueden hacer uso de un servidor que proteja el secreto de cliente.

Antes de continuar conviene aclarar que el estándar OAuth 2 ha publicado una variación sobre este flujo denominado PKCE (Proof Key for Code Exchange) para hacerlo más seguro y aunque este método sigue siendo válido se recomienda usar la nueva especificación PKCE descrita en la siguiente sección.

Para comenzar el flujo de autorización se debe hacer la siguiente llamada al servidor:

```
GET:https://gq.fortiming.com/ForTimingWeb/auth?response_type=token&client_
id=CLIENT_ID&redirect_uri=REDIRECT_URI&scope=read_organizacion%20read_tiem
pos&state=89ABC
```

Descripción de la llamada y sus parámetros:

- Es una llamada **GET**
- **response_type** debe indicar ***token*** solicitando así directamente el token de acceso.
- **client_id** es el identificador público de la aplicación. Se obtiene al registrar el programa.
- **redirect_uri** es la dirección a la que se redirigirá el 'User-Agent' (normalmente un navegador web) cuando el usuario autorice o no el acceso al programa. Esta dirección debe estar registrada por motivos de seguridad.
- **scope** son los niveles de acceso solicitados. En este caso se solicita la lectura de datos de la organización y la lectura de partes de tiempo. Coincide con el scope por defecto así que si no se indica se aplicará este automáticamente. Sin embargo si se indican valores incorrectos se producirá el error *unsupported_scope*.
- **state** es un parámetro opcional. Se recomienda que lo use el programa cliente para hacer un seguimiento del proceso. Suele ser un valor aleatorio que se usa para validar la respuesta del servidor de autorización ya que este mismo valor se mandará de vuelta en la respuesta de la autorización.

Se mostrará una web para autorizar el acceso al programa cliente. Una vez el usuario haya autorizado o no al programa, se redirigirá al '*User-Agent*' a la URI indicada en *redirect_uri*.

```
GET https://app-cliente.com/cb#token=AUTH_CODE_HERE&state=1234zyx
```

En esta redirección se manda el código de autorización y el state indicado en la llamada inicial, pero es importante fijarse que el token viene como un fragmento de la URI (usando el carácter '#') y no como parámetro de la *query string*. Los fragmentos sólo son útiles para el 'User-Agent' y por eso no se mandan al servidor evitando así que pueda ser interceptado. El programa cliente debe ser capaz de extraer el token del fragmento. Con el token ya podrá realizar llamadas al API de la siguiente manera.

```
curl --request GET \
 --url https://gq.fortiming.com/ForTimingWeb/api/departamentos \
 --header 'authorization: Bearer ACCESS_TOKEN' \
 --header 'content-type: application/json'
```

Una característica del flujo de autorización implícito es que no se proporciona un código refresco (refresh_token) por lo que cuando se cierre la sesión será necesario volver a autorizar el programa cliente para que tenga acceso al API.

Flujo de validación implícito PKCE (Proof Key for Code Exchange)

Este flujo de validación se parece al flujo de autorización ya que obtiene inicialmente un código de autorización. La diferencia es que no se usa el *client_secret* ya que no se puede proteger, se añade una protección adicional mandando un reto al servidor de autorización y al igual que en el flujo de

validación implícito anterior a esta especificación no se proporciona un código de refresco (*refresh_token*).

El reto es un texto aleatorio formado por los caracteres A-Z, a-z, 0-9 y los símbolos de puntuación -._~ con una longitud entre 43 y 128 caracteres.

Se inicia el proceso de validación haciendo la siguiente llamada:

```
GET:https://gq.fortiming.com/ForTimingWeb/auth?response_type=code&client_id=CLIENT_ID&redirect_uri=REDIRECT_URI&scope=read_organizacion%20read_tiempos&state=89ABC&code_challenge=CODE_CHALLENGE&code_challenge_method=S256
```

Descripción de la llamada y sus parámetros:

- Es una llamada **GET**
- **response_type** debe indicar el valor ***code*** para pedir un código de autorización.
- **client_id** es el identificador público de la aplicación. Se obtiene al registrar el programa.
- **redirect_uri** es la dirección a la que se redirigirá el 'User-Agent' (normalmente un navegador web) cuando el usuario autorice o no el acceso al programa. Esta dirección debe estar registrada por motivos de seguridad.
- **scope** son los niveles de acceso solicitados. En este caso se solicita la lectura de datos de la organización y la lectura de partes de tiempo. Coincide con el scope por defecto así que si no se indica se aplicará este automáticamente. Sin embargo si se indican valores incorrectos se producirá el error *unsupported_scope*.
- **state** es un parámetro opcional. Se recomienda que lo use el programa cliente para hacer un seguimiento del proceso. Suele ser un valor aleatorio que se usa para validar la respuesta del servidor de autorización ya que este mismo valor se mandará de vuelta en la respuesta de la autorización.
- **code_challenge** es el texto del reto cifrado y codificado en base64. El reto es un texto aleatorio que manda el programa cliente en el proceso de validación y que se usa para que el servidor valide la obtención del token resolviendo el reto al final del proceso.
- **code_challenge_method** indica el método de cifrado. La especificación contempla los métodos ***S256*** que hace referencia al algoritmo SHA-256 y ***plain*** que indica que el reto se manda sin cifrar.

Se mostrará una web para autorizar el acceso al programa cliente. Una vez el usuario haya autorizado o no al programa, se redirigirá al '*User-Agent*' a la URI indicada en *redirect_uri*.

```
GET https://app-cliente.com/cb#code=AUTH_CODE_HERE&state=1234zyx
```

En esta redirección se manda el código de autorización y el *state* indicado en la llamada inicial. Al igual que en el flujo implícito anterior a esta especificación, el código viene como un fragmento de la URI (usando el carácter '#') y no como parámetro de la *query string*. Los fragmentos sólo son útiles para el

'User-Agent' y por eso no se mandan al servidor evitando así que pueda ser interceptado. El programa cliente debe ser capaz de extraer el código del fragmento y el valor de *state*. Deberá comprobar el state para asegurarse que no es una redirección malintencionada y podrá terminar solicitando el token realizando la siguiente llamada.

```
POST:https://gq.fortiming.com/ForTimingWeb/token?grant_type=authorization_code&code=AUTH_CODE_HERE&redirect_uri=REDIRECT_URI&client_id=CLIENT_ID&challenge=CHALLENGE_TEXT
```

Descripción de la llamada y sus parámetros:

- Es una llamada **POST** que realiza el programa cliente.
- **grant_type** indica que se quiere autorizar un código para obtener un token de acceso al indicar el valor ***authorization_code.***
- **code** indica el código de autorización que nos han entregado en el paso anterior.
- **redirect_uri** deberá indicar la misma dirección usada para obtener el código de autorización. Solo se usa como elemento adicional de seguridad.
- **client_id** es el identificador público de programa.
- **challenge** es el texto original del reto sin cifrar y codificado en base64.

La respuesta que se espera del servidor es un objeto json.
Si la validación ha sido positiva el objeto json tendrá los siguiente valores:

```
HTTP/1.1 200 Ok
Content-Type: application/json;charset=UTF-8
{
	"access_token": "TOKEN",
	"token_type": "Bearer",
	"expires_in": 1600,
	"scope": "SCOPE",
	"url_base": "https://gq.fortiming.com",
	"user_info": {
			"id_personal": 0,
			"email": "user@mailserver.com"
		}
}
```

Descripción de las propiedades del objeto:

- **access_token** El token de acceso al API.
- **token_type** El tipo de token siempre va a ser *bearer* (token portadora).
- **expires_in** Tiempo de caducidad del token expresado en segundos.
- **scope** El nivel de acceso concedido. Podría no siempre coincidir con el solicitado.
- **url_base** Es la dirección base del servidor API y raíz de los endpoint.

- **user_info** Información sobre el usuario que ha realizado la autorización.

Si se ha producido algún error también se proporcionará un objeto json con información sobre el error. El objeto tendrá los siguientes campos:

```
HTTP/1.1 400 Bad Request
Content-Type: application/json;charset=UTF-8
{
      "error":" ERROR",
      "error_description": "DESCRIPTION",
      "error_uri": "URI"
}
```

Descripción de las propiedades del objeto:

- **error** Indica el nombre del error.
- **error_description** Da una descripción más detallada del error.
- **error_uri** Es opcional y si se indica proponer una dirección de ayuda para solucionar el error.

En este punto el programa cliente ya dispone del token y puede emplearlo para hacer llamadas al API. Para que el API reconozca el token se debe incluir en la cabecera de llamada de la siguiente manera:

```
curl --request GET \
 --url https://gq.fortimiming/ForTimingWeb/api/departamentos \
 --header 'authorization: Bearer ACCESS_TOKEN' \
 --header 'content-type: application/json'
```

Este flujo de validación tampoco proporciona un código de refresco y cuando caduque la sesión, será necesario volver a autorizar el programa cliente.

Flujo de validación usando credenciales de cliente

En este flujo de validación no interviene la acción del usuario. Es un flujo diseñado específicamente para la comunicación entre servidores. El servidor deberá ser de uso privado para mantener oculto el secreto de cliente. Este proceso inicia una sesión en la que no se identifica a ningún usuario de la empresa de manera concreta y concede acceso de lectura a todos los datos del esquema de empresa como si se hubiera accedido con la cuenta de un usuario superadministrador.

Para comenzar el proceso de autorización se solicita directamente un token de acceso realizando una llamada siguiendo el siguiente esquema:

```
POST:https://gq.fortiming.com/ForTimingWeb/token?grant_type=client_credentials&client_id=CLIENT_ID&client_secret=CLIENT_SECRET&scope=read_organization%20read_tiempos
```

Descripción de la llamada y sus parámetros:

- Es una llamada **POST**
- **grant_type** es necesario indicar el valor ***client_credentials*** para solicitar directamente un token de acceso para el programa cliente.
- **client_id** es el identificador público de programa.
- **client_secret** es el secreto del programa cliente obtenido al registrar la aplicación.
- **scope** son los niveles de acceso solicitados. En este caso se solicita la lectura de datos de la organización y la lectura de partes de tiempo. Coincide con el scope por defecto así que si no se indica se aplicará este automáticamente. Sin embargo si se indican valores incorrectos se producirá el error *unsupported_scope*.

La respuesta que se espera del servidor es un objeto json.

Si la validación ha sido positiva el objeto json tendrá los siguientes valores:

```
HTTP/1.1 200 Ok
Content-Type: application/json;charset=UTF-8
{
      "access_token":"TOKEN",
      "token_type": "Bearer",
      "expires_in": 1600,
      "scope": "SCOPE",
      "url_base": "https://gq.fortiming.com"
}
```

Descripción de las propiedades del objeto:

- **access_token** El token de acceso al API.
- **token_type** El tipo de token siempre va a ser *bearer* (token portadora).
- **expires_in** Tiempo de caducidad del token expresado en segundos.
- **scope** El nivel de acceso concedido. Podría no siempre coincidir con el solicitado.
- **url_base** Es la dirección base del servidor API y raíz de los endpoint.

Si se ha producido algún error también se proporcionará un objeto json con información sobre el error. El objeto tendrá los siguientes campos:

```
HTTP/1.1 400 Bad Request
Content-Type: application/json;charset=UTF-8
```

```
{                                                    "error":" ERROR",
      "error_description": "DESCRIPTION",
      "error_uri": "URI"
}
```

Descripción de las propiedades del objeto:

- **error** Indica el nombre del error.
- **error_description** Da una descripción más detallada del error.
- **error_uri** Es opcional y si se indica propone una dirección de ayuda para solucionar el error.

En este punto el programa cliente ya dispone del token y puede emplearlo para hacer llamadas al API. Para que el API reconozca el token se debe incluir en la cabecera de llamada de la siguiente manera:

```
curl --request GET \
      --url https://gq.fortimiming/ForTimingWeb/api/departamentos \
      --header 'authorization: Bearer ACCESS_TOKEN' \
      --header 'content-type: application/json'
```

Cierre de sesión

La sesión se cerrará automáticamente transcurridos 30 minutos después de su inicio o de su último refresco. Se aconseja por seguridad que el cliente cierre la sesión una vez terminado su uso. Para cerrar la sesión de forma voluntaria hay que realizar la siguiente llamada que tendrá distintos parámetros en función del tipo de autorización que se quiera revocar.

- **Flujo con código de autorización**
  En este tipo de autorización hay que usar el client_secret.

```
POST:https://gq.fortiming.com/ForTimingWeb/revoke_token?token=TOKEN&client
_id=CLIENT_ID&client_secret=CLIENT_SECRET
```

- **Flujos implícito e implícito PKCE**
  Al ser un cliente público en el que no se puede proteger el *client_secret* este no se usará. La llamada será como la siguiente.

```
POST:https://gq.fortiming.com/ForTimingWeb/revoke_token?token=TOKEN&client
_id=CLIENT_ID
```

Si el cierre de sesión ha sido correcto la respuesta al cliente será un código 200.

```
HTTP/1.1 200 Ok
```

Por el contrario, si se ha producido algún error se proporcionará un objeto json con información sobre el error. El objeto tendrá los siguientes campos:

```
HTTP/1.1 400 Bad Request
Content-Type: application/json;charset=UTF-8
{
      "error":" ERROR",
      "error_description": "DESCRIPTION",
      "error_uri": "URI"
}
```

Descripción de las propiedades del objeto:

- **error** Indica el nombre del error.
- **error_description** Da una descripción más detallada del error.
- **error_uri** Es opcional y si se indica propone una dirección de ayuda para solucionar el error.

Sesiones

Características de las sesiones del API:

- Tienen un tiempo de vida de 30 minutos.
- Se permite prolongar el tiempo de vida si se solicita un refresco de la autorización antes de que la sesión haya caducado si se dispone de un *refresh_token*.
- Si la sesión ha caducado, con el *refresh_token* se puede generar una nueva sesión sin necesidad de volver a autorizar al programa.

Códigos y autorizaciones

Conviene tener presente los siguientes aspectos en los flujos de autorización.

- Los códigos de autorización son de un solo uso.
- Los códigos de autorización caducan al de 10 minutos si no se han usado.
- Los *refresh_token* de las autorizaciones no caducan si las autorizaciones se utilizan, pero se eliminarán si no se utilizan en menos de 30 días.
- La longitud de los códigos de autorización, *refresh_token* y *token* puede ser entre 40 y 64 caracteres de largo.

Códigos de error en el proceso de autorización

Estos son los errores que puede generar el servidor de autorización si se detectan problemas en el proceso.

- **invalid_request** Faltan parámetros o son incorrectos.
- **unsupported_grant_type** Tipo de solicitud no soportado.
- **unsupported_scope** Rol de acceso no soportado.
- **invalid_client** Código de cliente incorrecto.
- **invalid_grant** Códigos caducados o incorrectos.
- **unauthorized_client** Cliente sin el permiso de acceso solicitado.
- **internal_error** Error interno inesperado.
- **license_error** Error de licencia por cuenta deshabilitada, bloqueada o fin de servicio.

Códigos de respuesta en las llamadas al API

En las llamadas a los endpoints del API de podrán generar los siguientes códigos de respuesta:

- **200 OK** Solicitud aceptada. La respuesta contiene el resultado.
- **400 BAD REQUEST** La solicitud enviada es incorrecta.
- **403 FORBIDDEN** El cliente ha intentado acceder a un recurso sin contar con el nivel de permiso necesario, sin el token de acceso, con un token espirado, o desde una IP no incluida en lista blanca.
- **404 NOT FOUND** El recurso concreto al que se ha llamado no existe.
- **500 INTERNAL SERVER ERROR** Error inesperado en el servidor.

Scope definidos

- **read_organizacion** Leer datos de organización.
- **write_organizacion** Escribir datos de organización
- **read_tiempos** Leer datos de partes de tiempo.
- **write_tiempos** Escribir datos de partes de tiempo
- **read_gastos** Leer datos de notas de gastos
- **write_gastos** Escribir datos de notas de gastos
- **read_fichaje** Leer datos de registro de jornada
- **write_fichaje** Escribir datos de registro de jornada

Consideraciones adicionales de seguridad

- No se obliga, pero sí se recomienda que las redirecciones de retorno estén protegidas mediante TLS.

Referencias

Recursos y especificaciones del protocolo OAuth
https://oauth.net/
Especificación OAuth 2.0 para Bearer Tokens
https://tools.ietf.org/html/rfc6750
Documentación del API
https://gq.fortiming.com/swagger-ui/