

Manual de API de Integradores

FACe, Punto General de
Entrada de Facturas
Electrónicas de la
Administración General
del Estado



TLP: CLEAR



Financiado por
la Unión Europea
NextGenerationEU



AUTOR	SGAD
ÁREA	Catálogos
PROYECTO	FACE, Punto General de Entrada de Facturas de la Administración General del Estado
NOMBRE DEL DOCUMENTO	Manual de API de Integradores

Control de Versiones del Documento

VERSIÓN	AUTOR	FECHA	DESCRIPCIÓN
2.0.0	Equipo FACE	19/11/2025	<ul style="list-style-type: none"> Cambio de plantilla Revisión y corrección de errores
1.0.0	Equipo FACE	03/1/2024	Versión inicial del documento

Índice

1	INTRODUCCIÓN	5
1.1	Objetivos	5
2	CONSIDERACIONES PREVIAS	6
2.1	Alta como integrador	6
2.2	Catálogo de APIs	6
2.3	Conceptos de autenticación	6
3	DESCRIPCIÓN DEL API	9
3.1	Grupo Integradores	9
3.1.1	Autorizados (/authorized)	9
3.1.1.1	Listado de Autorizados [GET]	9
3.1.1.2	Crear Autorizado [POST]	10
3.1.1.3	Eliminar Autorizados [DELETE]	12
3.1.2	Sistemas (/system)	14
3.1.2.1	Crear Sistema [POST]	14
3.1.2.2	Asociar Certificado al Sistema [POST]	16
3.1.2.3	Desasociar Certificado al Sistema [DELETE]	18
3.1.2.4	Editar Sistema [PUT]	19
3.1.2.5	Eliminar Sistema [DELETE]	21
3.1.3	Certificados (/certificate)	22
3.1.3.1	Asociar certificado al Integrador [POST]	22
3.1.3.2	Desasociar Certificado al Integrador [DELETE]	24
3.2	Grupo FAQ	26
3.2.1	FAQ (/faqs)	26
3.2.1.1	Recuperar el listado de FAQ del sistema [GET]	26
3.3	Grupo Noticias	27
3.3.1	News (/news)	27
3.3.1.1	Visualizar las noticias del portal [GET]	27
3.4	Grupo Notificaciones	29
3.4.1	Notificaciones (/notifications)	29
3.4.1.1	Visualizar las notificaciones del portal [GET]	29
3.5	Grupo Presentación	30
3.5.1	Slides (/slides)	30
3.5.1.1	Recuperar las diapositivas del portal [GET]	30

ANEXO I 32

1 INTRODUCCIÓN

1.1 Objetivos

El presente documento tiene como objetivo servir de manual de uso de las APIs expuestas por el Punto General de Entrada de Facturas Electrónicas de la AGE, conocido como FACE.

En el apartado 2 Consideraciones previas se tratarán aspectos relacionados con el alta como integrador dentro del sistema FACE y de seguridad como es el modelo de autenticación que expone el API.

2 CONSIDERACIONES PREVIAS

2.1 Alta como integrador

Para poder utilizar los servicios web REST de FAcE el usuario se tiene que dar de alta previamente en el **Portal de Integradores** cuyas URL son:

Servicios Estables	Producción
https://se-integradores-face.redsara.es/inicio	https://integradores.face.gob.es/inicio

En dicho portal podrá:

- Darse de alta como integrador
- Generar plataformas, tantas como se necesiten, y asociarles certificados (PEM)
- Gestionar autorizados.

Los certificados tendrán que ser los que se utilicen para firmar las peticiones SOAP para que FAcE pueda verificar quién está llamando a los servicios web y si tiene permisos para ello.

2.2 Catálogo de APIs

Puede encontrar las API en las siguientes rutas:

- Servicios estables: <https://se-api-face.redsara.es/integrators/doc>
- Producción: <https://api.face.gob.es/integrators/doc>

2.3 Conceptos de autenticación

Las invocaciones a la API están securizadas mediante tokens a través de un JWT firmado con JWS. El integrador deberá generar un token que tendrá que incluirse en la cabecera de las peticiones. El token no es más que la firma cifrada que permita al API identificar al usuario.

- **JWT (JSON Web Token):** El token en sí. Un JWT es un estándar abierto (RFC 7519, <https://datatracker.ietf.org/doc/html/rfc7519>) que define una forma compacta y segura de representar afirmaciones entre dos partes como un objeto JSON.

TLP: CLEAR

USO PÚBLICO

Comentado [PMB1]: Revisar

Comentado [EGD2R1]: Cambiado. Confirmado con el equipo y me han dado este texto, aunque está reformulado.

- **JWS (JSON Web Signature):** El mecanismo de firma. Es un estándar relacionado con JWT que define una forma de firmar digitalmente un mensaje JSON. La firma digital garantiza la integridad y autenticidad del mensaje.

El tiempo de validez del token será de 5 minutos. Una vez pasado este tiempo de validez, el servidor no permitirá más el acceso a recursos con dicho token y deberá generarse un nuevo token.

Un JWT firmado con JWS consta de tres partes codificadas en base 64 y separadas por puntos:

- **Header:** Un JSON que contiene información sobre el tipo de token (JWT), el algoritmo de firma utilizado y otros metadatos. En nuestro caso es un JSON con el tipo de token, su codificación (RSA Signature with SHA-256 (RS256) o HMAC with SHA-256 (HS256)) y una clave xc5 cuyo valor es un array con el que llamaremos PEM limpio, y que es el PEM, o parte publica, del certificado con el que se va a firmar en base 64, en una línea, sin espacios ni saltos de línea, ni los tags de -----BEGIN/END CERTIFICATE-----. Todo ello, codificado a base 64:

```
{  
  "typ": "JWT",  
  "alg": "RS256",  
  "xc5": [ "(PEM limpio )" ]  
}
```

- **Payload:** Un JSON con los claims o atributos, que son pares clave-valor que definen al token y contienen información sobre la entidad a la que pertenece el token. Todo ello, codificado a base 64:

```
{  
  "username": "(Hash o huella digital SHA1 del PEM limpio)",  
  "iat": "(fecha de creación del token en formato de tiempo UNIX)",  
  "exp": "(fecha de expiración del token en formato de tiempo UNIX.)"  
}
```

- **Signature:** Un valor calculado sobre el header y el payload. Se concatenan las representaciones base64 del header y payload, separadas por un punto. Se calcula el hash de la cadena concatenada utilizando el algoritmo SHA-256. Se utiliza la clave privada RSA para firmar el hash. Se codifica en base64 la firma resultante y el resultado es la signature.

Tiene más información, una herramienta de creación de token JWT, decodificadores y generadores de JWT, conversor de tiempo unix y de hash en las páginas:

- <https://jwt.io/>
- <https://jwt.one/https://tribestream.io/tools/jwt/>
- <https://www.jstoolset.com/jwt>
- https://time.is/es/Unix_time_converter
- <https://www.convertstring.com/es/Hash/SHA1>

Para mayor privacidad en la generación del JWT tienen bibliotecas de lenguajes de programación como:

- Node.js: jsonwebtoken o Passport.js
- PHP: Firebase PHP Admin SDK, Lcobucci JWT, Sodium PHP, RobRichards/jwt
- Python: PyJWT
- Java: Nimbus JOSE+JWT jjwt
- .NET: System.IdentityModel.Tokens.Jwt

En el [Anexo I](#) se muestra un ejemplo del código en java para generar el token.

3 DESCRIPCIÓN DEL API

3.1 Grupo Integradores

Los recursos ubicados bajo la ruta `/integrators` están relacionados con la gestión del integrador del usuario logado.

3.1.1 Autorizados (`/authorized`)

Los autorizados son personas físicas/jurídicas a los que un Integrador habilita para gestionar su perfil de integrador: plataformas, certificados y otros autorizados.

3.1.1.1 Listado de Autorizados [GET]

Devuelve la información del integrador logado.

- API Integrators

```
/v1/integrators/{identifier}/authorizeds
```

- Parámetros de path

- Identifier (string, obligatorio)

- Petición (text/plain)

```
{ }
```

- Response 201 (application/json)

- Body

```
{
  "items": [
    {
      "identifier": "string",
      "fullName": "string"
    }
  ]
}
```

- Response 401 (application/json)

- Body

```
{
  "code": "401",
  "message": "Unauthorized"
}
```

- Response 403 (application/json)

- Body

```
{
  "code": "403",
  "message": "Forbidden"
}
```

- Response 500 (application/json)

- Body

```
{
  "code": 500,
  "message": "Internal Server Error"
}
```

3.1.1.2 Crear Autorizado [POST]

Da de alta al usuario logado como un integrador.

- API integrators

```
/v1/integrators/{identifier}/authorizeds
```

- Parámetros de path

TLP: CLEAR

USO PÚBLICO

- Identifier (string, obligatorio)
- Parámetros de body
 - identifier (string, obligatorio)
 - integrator (string, obligatorio)
 - name (string, obligatorio)
 - surname1 (string, obligatorio)
 - surmane2 (string, obligatorio)
 - email (string, obligatorio)
 - conflicto (string, obligatorio)
 - message (string, obligatorio)

- Petición (text/plain)

```
{
  "integrator": "11111111H",
  "identifier": "00000000T",
  "name": "",
  "surname1": "",
  "surname2": "",
  "email": "",
  "conflict": {
    "message": ""
  }
}
```

- Response 201 (application/json)

- Body

```
{
  "identifier": "string",
  "fullName": "string",
}
```

- Response 400 (application/json)

- Body

```
{
  "errors": [
    "string"
  ],
  "code": "400",
  "message": "Bad Request"
}
```

- Response 401 (application/json)

- Body

```
{
  "code": "401",
  "message": "Unauthorized"
}
```

- Response 403 (application/json)

- Body

```
{
  "code": "403",
  "message": "Forbidden"
}
```

- Response 500 (application/json)

- Body

```
{
  "code": 500,
  "message": "Internal Server Error"
}
```

3.1.1.3 Eliminar Autorizados [DELETE]

Permite eliminar un autorizado de un integrador.

- API integrators

```
/v1/integrators/{identifier}/authorizeds/{authorized}
```

- Parámetros del path

- identifier (string, obligatorio)
- authorized (string, obligatorio)

- Petición (text/plain)

```
{  
  "identifier": "string",  
  "authorized": "string",  
}
```

- Response 204 (application/json)

- Body

```
{}
```

- Response 401 (application/json)

- Body

```
{  
  "code": "401",  
  "message": "Unauthorized"  
}
```

- Response 403 (application/json)

- Body

```
{  
  "code": "403",  
  "message": "Forbidden"  
}
```

- Response 500 (application/json)

- Body

```
{  
  "code": 500,  
  "message": "Internal Server Error"  
}
```

3.1.2 Sistemas (/system)

Los autorizados son personas físicas/jurídicas a los que un Integrador habilita para gestionar su perfil de integrador: plataformas, certificados y otros autorizados.

3.1.2.1 Crear Sistema [POST]

Dar de alta un sistema.

- API Integrators

/v1/integrators/{identifier}/systems

- Parámetros de body

- Integrator (string, obligatorio)
- Alias (string, obligatorio)
- name (string, obligatorio)
- canSend (booleano, opcional)
- canReceive (booleano, opcional)
- hash (string)

- Parámetros de path

- Identifier (string, obligatorio)

- Petición (text/plain)

```
{
  "integrator": "99999999R",
  "alias": "",
  "name": "pruebas_23",
  "canSend": true,
  "canReceive": true,
  "hash": "56485e81bfd5771f1e8c9bae981bd8bbdda115c5"
}
```

- Response 201 (application/json)

- Body

```
{
  "uuid": "67dae4108a2f3",
  "name": "pruebas_23",
  "certificates": [
    {
      "alias": "pruebas",
      "issuer": "FNMT-RCM",
      "serial":
"75078687968838737208550885083691084024",
      "hash":
"56485e81bfd5771f1e8c9bae981bd8bbdda115c5"
    }
  ],
  "administrations": [
    {
      "code": "string",
      "name": "string"
    }
  ],
  "canSend": true,
  "canReceive": true,
  "createdAt": "2025-03-19 16:34:40",
  "updatedAt": "2025-03-19 16:34:40"
}
```

- Response 400 (application/json)

- Body

```
{
  "errors": [
    "string"
  ],
  "code": "400",
  "message": "Bad Request"
}
```

- Response 401 (application/json)

- Body

```
{
  "code": "401",
  "message": "Unauthorized"
}
```

- Response 500 (application/json)

- Body

```
{
  "code": 500,
  "message": "Internal Server Error"
}
```

3.1.2.2 Asociar Certificado al Sistema [POST]

Agregar un certificado al sistema. Cada sistema puede tener más de un certificado asociado.

- API integrators

```
/v1/integrators/{identifier}/systems/{uuid}/certificates/{hash}
```

- Parámetros de path

- identifier (string, obligatorio)
- uuid (string, obligatorio)
- hash (string, obligatorio)

- Petición (text/plain)

```
{
  "identifier": "string",
  "name": "string",
  "surname1": "string",
  "surname2": "string",
  "email": "string"
}
```


- Response 201 (application/json)

- Body

```
{
  "uuid": "string",
  "name": "string",
  "certificates": [
    {
      "alias": "string",
      "issuer": "string",
      "serial": "string",
      "hash": "string"
    }
  ],
  "administrations": [
    {
      "code": "string",
      "name": "string"
    }
  ],
  "canSend": true,
  "canReceive": true,
  "createdAt": "2025-03-20T08:26:31.405Z",
  "updatedAt": "2025-03-20T08:26:31.405Z"
}
```

- Response 400 (application/json)

- Body

```
{
  "errors": [
    "string"
  ],
  "code": "400",
  "message": "Bad Request"
}
```

- Response 401 (application/json)

- Body

```
{
  "code": "401",
  "message": "Unauthorized"
}
```

- Response 500 (application/json)

- Body

```
{
  "code": 500,
  "message": "Internal Server Error"
}
```

3.1.2.3 Desasociar Certificado al Sistema [DELETE]

Permite eliminar un certificado de un sistema.

- API integrators

```
/v1/integrators/{identifier}/systems/{uuid}/certificates/{hash}
```

- Parámetros del path

- identifier (string, obligatorio)
- uuid (string, obligatorio)
- hash (string, obligatorio)

- Petición (text/plain)

```
{
  "identifier": "string",
  "authorized": "string",
}
```

- Response 204 (application/json)

- Body

```
{}
```

- Response 401 (application/json)

- Body

```
{
  "code": "401",
  "message": "Unauthorized"
}
```

- Response 403 (application/json)

- Body

```
{  
  "code": "403",  
  "message": "Forbidden"  
}
```

- Response 500 (application/json)

- Body

```
{  
  "code": 500,  
  "message": "Internal Server Error"  
}
```

3.1.2.4 Editar Sistema [PUT]

Actualizar la información de un sistema.

- API Integrators

```
/v1/integrators/{identifier}/systems/{uuid}
```

- Parámetros de body

- Uuid (string, obligatorio)
 - name (string, obligatorio)
 - canSend (booleano, opcional)
 - canReceive (booleano, opcional)
 - createdAt (datetime, obligatorio)
 - integrator (string, obligatorio)
 - administrations (string, opcional)

- Parámetros de path

- Identifier (string, obligatorio)
 - uuid (string, obligatorio)

- Petición (text/plain)

```
{
  "uuid": "67dae4108a2f3",
  "name": "pruebas_esti54",
  "canSend": true,
  "canReceive": true,
  "createdAt": "2025-03-19 16:34:40",
  "integrator": "99999999R",
  "administrations": [
  ]
}
```

- Response 200 (application/json)

- Body

```
{
  "uuid": "67dae4108a2f3",
  "name": "pruebas_esti54",
  "certificates": [
    {
      "alias": "pruebas",
      "issuer": "FNMT-RCM",
      "serial":
"75078687968838737208550885083691084024",
      "hash":
"56485e81bfd5771f1e8c9bae981bd8bbdda115c5"
    }
  ],
  "administrations": [
    {
      "code": "string",
      "name": "string"
    }
  ],
  "canSend": true,
  "canReceive": true,
  "createdAt": "2025-03-19 16:34:40",
  "updatedAt": "2025-03-20 09:43:51"
}
```

- Response 400 (application/json)

- Body

```
{
  "errors": [
    "string"
  ],
  "code": "400",
  "message": "Bad Request"
}
```

- Response 401 (application/json)

- Body

```
{
  "code": "401",
  "message": "Unauthorized"
}
```

- Response 500 (application/json)

- Body

```
{
  "code": 500,
  "message": "Internal Server Error"
}
```

3.1.2.5 Eliminar Sistema [DELETE]

Eliminar un sistema.

- API Integrators

```
/v1/integrators/{identifier}/systems/{uuid}
```

- Parámetros de path

- identifier (string, obligatorio)
- uuid (string, obligatorio)

- Petición (text/plain)

```
{}
```

- Response 204 (application/json)

- Body

```
{}
```

- Response 401 (application/json)

- Body

```
{
  "code": "401",
  "message": "Unauthorized"
}
```

- Response 403 (application/json)

- Body

```
{
  "code": "403",
  "message": "Forbidden"
}
```

- Response 500 (application/json)

- Body

```
{
  "code": 500,
  "message": "Internal Server Error"
}
```

3.1.3 Certificados (/certificate)

3.1.3.1 Asociar certificado al Integrador [POST]

Asociar directamente un certificado a un integrador.

- API Integrators

```
/v1/integrators/{identifier}/systems
```

- Parámetros de body

- alias (string, obligatorio)
- publicKey (string, opcional)

TLP: CLEAR

USO PÚBLICO

- integrator (string, obligatorio)
- file (string, obligatorio)
- Parámetros de path
 - Identifier (string, obligatorio)
- Petición (text/plain)

```
{
  "integrator": "99999999R",
  "publicKey": "-----BEGIN CERTIFICATE-----
\r[...] /ZA\r\ng6e8Y1Zw\r\n-----END
CERTIFICATE-----\r\n",
  "alias": "X0000000T",

  "file": "ESPAÑOL_NIEPRUEBAS_PRUEBAS___X0000000T
.pem"
}
```

- Response 201 (application/json)

- Body

```
{
  "alias": "X0000000T",
  "issuer": "FNMT-RCM",
  "serial":
"94839352366543998784220973016859820468",
  "hash":
"aa2061dc22d2c36552cc324944d3e55bbc7d4d3c",
  "classification": "0",
  "validFrom": "2022-11-16 15:14:14",
  "validTo": "2026-11-16 15:14:14",
  "createdAt": "2025-03-20 09:23:26",
  "updatedAt": "2025-03-20 09:23:26",
  "isExpired": false
}
```

- Response 400 (application/json)

- Body

```
{
  "errors": [
    "string"
  ],
  "code": "400",
  "message": "Bad Request"
}
```

- Response 401 (application/json)

- Body

```
{
  "code": "401",
  "message": "Unauthorized"
}
```

- Response 500 (application/json)

- Body

```
{
  "code": 500,
  "message": "Internal Server Error"
}
```

3.1.3.2 Desasociar Certificado al Integrador [DELETE]

Desasociar directamente un certificado a un integrador.

- API integrators

```
/v1/integrators/{identifier}/certificates/{hash}
```

- Parámetros de path

- identifier (string, obligatorio)
- hash (string, obligatorio)

- Petición (text/plain)

```
{}
```


- Response 204 (application/json)

- Body

```
{}
```

- Response 401 (application/json)

- Body

```
{  
  "code": "401",  
  "message": "Unauthorized"  
}
```

- Response 403 (application/json)

- Body

```
{  
  "code": "403",  
  "message": "Forbidden"  
}
```

- Response 500 (application/json)

- Body

```
{  
  "code": 500,  
  "message": "Internal Server Error"  
}
```

3.2 Grupo FAQ

Los recursos ubicados bajo la ruta `/faqs` están relacionados con la gestión de las preguntas frecuentes del portal integradores.

3.2.1 FAQ (/faqs)

3.2.1.1 Recuperar el listado de FAQ del sistema [GET]

- API Integrators

```
/integrators/v1/faqs
```

- Parámetros de la query

- `site` (string, obligatorio): *private* o *public*.

- Petición (text/plain)

```
{}
```

- Response 202 (application/json)

- Body

```
{
  "items": [
    {
      "id": 0,
      "block": "string",
      "title": "string",
      "description": "string",
      "dateFrom": "2025-03-20T10:19:12.230Z",
      "dateTo": "2025-03-20T10:19:12.230Z",
      "createdAt": "string",
      "active": true,
      "sites": [
        "string"
      ]
    }
  ]
}
```

- Response 401 (application/json)

- Body

```
{
  "code": "401",
  "message": "Unauthorized"
}
```

- Response 500 (application/json)

- Body

```
{
  "code": 500,
  "message": "Internal Server Error"
}
```

3.3 Grupo Noticias

Los recursos ubicados bajo la ruta `/news` están relacionados con la gestión de las noticias que aparecen en el portal de integradores.

3.3.1 News (/news)

3.3.1.1 Visualizar las noticias del portal [GET]

- API Integrators

```
/integrators/v1/news
```

- Parámetros de la query

- site (string, obligatorio): *private* o *public*.

- Petición (text/plain)

```
{}
```

- Response 200 (application/json)

- Body

```
{
  "items": [
    {
      "id": 0,
      "title": "string",
      "description": "string",
      "dateFrom": "2025-03-20T10:23:51.559Z",
      "dateTo": "2025-03-20T10:23:51.559Z",
      "createdAt": "string",
      "active": true,
      "fixed": true,
      "sites": [
        "string"
      ],
      "position": 0
    }
  ]
}
```

- Response 401 (application/json)

- Body

```
{
  "code": "401",
  "message": "Unauthorized"
}
```

- Response 500 (application/json)

- Body

```
{
  "code": 500,
  "message": "Internal Server Error"
}
```

3.4 Grupo Notificaciones

Los recursos ubicados bajo la ruta `/notifications` están relacionados con la gestión de las noticias que aparecen en el portal de integradores.

3.4.1 Notificaciones (/notifications)

3.4.1.1 Visualizar las notificaciones del portal [GET]

- API Integrators

```
/integrators/v1/notifications
```

- Parámetros de la query

- site (string, obligatorio): *private* o *public*.

- Petición (text/plain)

```
{}
```

- Response 200 (application/json)

- Body

```
{
  "items": [
    {
      "id": 0,
      "title": "string",
      "description": "string",
      "notificationsButton": {
        "text": "string",
        "link": "string"
      },
      "dateFrom": "2025-03-20T10:40:26.667Z",
      "dateTo": "2025-03-20T10:40:26.667Z",
      "createdAt": "string",
      "active": true,
      "fixed": true,
      "notificationLevel": "string",
      "sites": [
        "string"
      ],
      "position": 0
    }
  ]
}
```

TLP: CLEAR

USO PÚBLICO

- Response 401 (application/json)

- Body

```
{  
  "code": "401",  
  "message": "Unauthorized"  
}
```

- Response 500 (application/json)

- Body

```
{  
  "code": 500,  
  "message": "Internal Server Error"  
}
```

3.5 Grupo Presentación

Los recursos ubicados bajo la ruta `/slides` están relacionados con la administración de las diapositivas que aparecen en el portal de integradores.

3.5.1 Slides (/slides)

3.5.1.1 Recuperar las diapositivas del portal [GET]

- API Integrators

```
/integrators/v1/notifications
```

- Parámetros de la query

- site (string, obligatorio): *private* o *public*.

- Petición (text/plain)

```
{}
```

- Response 200 (application/json)

- Body

```
{
  "items": [
    {
      "id": 0,
      "image": "string",
      "title": "string",
      "description": "string",
      "notificationsButton": {
        "text": "string",
        "link": "string"
      },
      "dateFrom": "2025-03-20T10:43:02.319Z",
      "dateTo": "2025-03-20T10:43:02.319Z",
      "active": true,
      "fixed": true,
      "sites": [
        "string"
      ],
      "position": 0
    }
  ]
}
```

- Response 401 (application/json)

- Body

```
{
  "code": "401",
  "message": "Unauthorized"
}
```

- Response 500 (application/json)

- Body

```
{
  "code": 500,
  "message": "Internal Server Error"
}
```

ANEXO I

El siguiente anexo muestra un ejemplo en java de generación del token.

```
import java.io.File;
import java.io.FileInputStream;
import java.security.KeyStore;
import java.security.PrivateKey;
import java.security.Signature;
import java.util.Arrays;
import java.util.Base64;
import java.util.Enumeration;

def project = context.testCase.testSuite.project

this.getClass().classLoader.addURL(new
File(context.expand(project.resourceRoot) +
"/libs/google-http-client-jackson2-
1.28.0.jar").toURL());
this.getClass().classLoader.addURL(new
File(context.expand(project.resourceRoot) +
"/libs/google-http-client-1.28.0.jar").toURL());

File key_pem = new
File(context.expand(project.resourceRoot) + "Ruta a
la clave publica del certificado");

String pem =
org.apache.commons.io.IOUtils.toString(new
FileInputStream(key_pem));

Signature sig =
Signature.getInstance("SHA256WithRSA");

KeyStore ks =
java.security.KeyStore.getInstance("PKCS12");
File key = new
File(context.expand(project.resourceRoot) + "Ruta
a la clave privada del certificado");
ks.load(new
FileInputStream(key),"alias_certificado".toCharArray());

Enumeration aliases = ks.aliases();
String keyAlias = "";
while (aliases.hasMoreElements()) {
    keyAlias = (String) aliases.nextElement();
    System.out.println(keyAlias);
}
//sig.initSign((PrivateKey)ks.getKey(keyAlias,
"alias_certificado".toCharArray()));
```



```

PrivateKey privateKey ;
com.google.api.client.json.webtoken.JsonWebSignatur
e.Header header = new
com.google.api.client.json.webtoken.JsonWebSignatur
e.Header();
header.setAlgorithm("RS256");
//header.setX509Certificates(Arrays.asList(new
String[]
{Base64.getEncoder().encodeToString((ks.getCertific
ate(keyAlias).getEncoded()))});
def cert_enc =
Base64.getEncoder().encodeToString(ks.getCertificat
e(keyAlias).getEncoded());
def res = [];
res.add(cert_enc);
List certs = Arrays.asList(res);
header.setX509Certificates(certs);
com.google.api.client.json.webtoken.JsonWebToken.Pa
yload payload = new
com.google.api.client.json.webtoken.JsonWebToken.Pa
yload();
long now = System.currentTimeMillis() / 1000L;
long exp = now + 7200;
payload.set("exp", exp);
payload.set("iat", now);
com.google.api.client.json.JsonFactory JSON_FACTORY
=
com.google.api.client.json.jackson2.JacksonFactory.
getDefaultInstance();
def token =
com.google.api.client.json.webtoken.JsonWebSignatur
e.signUsingRsaSha256((PrivateKey)ks.getKey(keyAlias
, "alias_certificado".toCharArray()) ,
JSON_FACTORY, header, payload);
context.testCase.setPropertyValue('token', new
String(token));
log.info(token)

```

TLP: AMBER

