
FACeB2B

Plataforma de distribución de facturas electrónicas entre empresas

Manual de API de Invoice para integradores

Versión

V1.0.1

Fecha de revisión

15/03/2024

Realizado por

Equipo FACeB2B



HISTÓRICO DE VERSIONES

Versión	Fecha	Revisado por	Descripción
1.0.0	01/05/2019	Equipo QA	Versión conforme a lo requerido en el Pliego de Prescripciones Técnicas 3301.
1.0.1	15/03/2024	Equipo QA	Revisión completa del documento

ÍNDICE

1 Introducción	4
1.1 <i>Objetivos</i>	4
2 Consideraciones previas	4
2.1 <i>Alta como integrador</i>	4
2.2 <i>Catálogo de APIs</i>	5
2.3 <i>Conceptos de autenticación</i>	5
3 Descripción del API	7
3.1 <i>Grupo Invoice</i>	7
3.1.1. <i>Invoice (/invoice)</i>	7
3.1.1.1. Remitir una factura [POST]	7
3.1.1.2. Descargar el archivo de la factura [GET]	8
3.1.1.3. Actualizar el estado de una factura [PATCH]	9
3.1.1.4. Ver detalle de la factura [GET]	10
3.1.1.5. Obtener listado de facturas [GET]	10
3.1.1.6. Validar firma de factura [POST]	12
3.1.1.7. Obtener detalle de una factura registrada [GET]	13
3.1.2. <i>Cancel Request (/invoices)</i>	14
3.1.2.1. Obtener las solicitudes de anulación de facturas [GET]	14
3.1.2.2. Solicitar la cancelación de una factura [POST]	14
3.1.2.3. Aceptar/Rechazar una solicitud de anulación [PATCH]	15
4 Anexo I	16

1 Introducción

1.1 Objetivos

El presente documento tiene como objetivo servir de manual de uso de las APIs expuestas por la Plataforma de distribución de facturas electrónicas entre empresas, FACeB2B.

En el apartado 2 Consideraciones previas se tratarán aspectos relacionados con el alta como integrador dentro del sistema FACeB2B y de seguridad como es el modelo de autenticación que expone el API.

Esta versión del documento tiene por objetivo hacer constar la incorporación al análisis funcional los requisitos derivados de lo dispuesto en el Documento de Licitación para la Contratación de Servicios basado en el Acuerdo Marco 26/2015 con título Desarrollo del Hub de intercambio de facturas entre empresas FACeB2B; con número de expediente 3301.

2 Consideraciones previas

2.1 Alta como integrador

Bajo la denominación de integrador se hace referencia a los usuarios de la Plataforma de distribución de facturas electrónicas entre empresas que interactúan con dicho sistema por medio de las APIs del catálogo de FACeB2B y no a través de los portales web que ofrece la plataforma.

Los integradores, administradores de las plataformas creadas en FACeB2B, pueden hacer uso del API Privada utilizando para la realizar las llamadas al sistema únicamente lo/s certificado/s que se hayan dado de alta en el portal de FACeB2B asociado a una de las plataformas que puede tener asociada una ESF.

No serán válidas las llamadas a la API firmadas con los certificados identificados en el sistema como administradores de WS, éstos únicamente serán válidos para las APIs de Clients e Invoicing Services.

2.2 Catálogo de APIs

Puede encontrar las APIs en las siguientes rutas:

- Servicios estables: <https://se-faceb2b.redsara.es/api/doc>
- Producción: <https://faceb2b.gob.es/api/doc>

En su versión V1 FACeB2B expone la siguiente API:

- /v1/invoices/

2.3 Conceptos de autenticación

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

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.

El token se genera a partir de un objeto JSON codificado en base 64 con los siguientes elementos:

- Header

El primer elemento es la cabecera del token, que a su vez tiene otros elementos, el tipo, en este caso un JWT y la codificación utilizada. El contenido sería el siguiente:

```
{
  "typ": "JWT",
  "alg": "RS256",
  "x5c":
    MIIF/TCCBOWgAwIBAgIQUj5ofy7TYXhWsKBZVz6lxDANBgkqhki
    G9w0BAQsFADBHMQswCQYDVQQGEwJFUzERMA8GA1UECgwI
    Rk5NVC1SQ00xJTAjBgNVBAsMHEFDIENvbXBvbmVudGVzIEluZm
    9ybcOhdGlib3MwHhcNMTYwMjAyMTIyNjAxWhcNMTkwMjAy
    MTIyNTU5WjCB2DELMAkGA1UEBhMCRVVMxMzANBgNVBAMk
    1BRFJJRDE8MDoGA1UECgwzTUlOSVNURVJJTyBERSBIQUJRU5
    EQSBZIEFETUlOSVNUNUkFDSU9ORVMgUM0aQkxJQ0FTMUsW
    YDVQQQLDEJESVJFQ0Njw5NOIERFIFRFQ05PTE9Hw41BUyBERSB
    MQSBCw6uB8h0EQ2JlRpKfqtDkh+ipDmUfinageM11sMXEebS+Yx
    O0iiqK0WgHPG27dSzd0Tfo2SKQ/XHsguTtrIoV4kktGhkb7IEpO8+
    G8QzHd347HiQAY/MruzeLAJjaBhcYzkCmMFw5xWc7k6PB0S82h
    eFB6RN+/SyPFj1QDUiZoVOuo4dfgFUZLCVBfMUBY73WNazVAoj
    qZhG9d8tAgg2c64nusuMDY+25MLUKFzsbzFg==
}
```

- Payload

Compuesto por los llamados JWT Claims donde irán colocados los atributos que definen el token. Existen varios, los más comunes a utilizar son:

- iat: Identifica la fecha de creación del token, válido para si se quiere poner una fecha de caducidad. En formato de tiempo UNIX
- exp: Identifica a la fecha de expiración del token. Se puede calcular a partir del iat. También en formato de tiempo UNIX.

```
{
  "iat": "1424180484",
  "exp": "1425390142"
}
```

- Signature

La firma es el tercer elemento. Está formado por los anteriores componentes (Header y Payload) cifrados en Base64 con una clave secreta (certificado identificado como Plataforma de Facturación), sirviendo de Hash para comprobar que el token es correcto.

El token codificado con el certificado sería:

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCIsIng1YyI6WmYJNSUIGL1RDQ0JPV2dBdD0lCQWdJUVVqNW9meTdUWVhoV3NLQlpWejZseERBTkNa3Foa2IHOXcwQkFRc0ZBREJITVFzd0NRWURWUWFHRXdKRIV6RVJNQThHQTFRUNnd0ISazVOVkmXU1EwMHhKVEFqQmdOVkJBc01IRUZESUV0dmJYQnZibVZ1ZEdWekIFbHVabTl5YmNPaGRHbGpiM013SGhjTk1UWXdkNakF5TVRJeU5qQXhXaGNOTVRrd01qQXINVEI5TIRVNVdqQ0lyREVMTUFR0ExVUVCaE1DUlZNeER6QU5CZ05WQkFjTUJrMUJSRkpKUkRFOE1Eb0dBMVVFQ2d3eIRVbE9TVk5VUIZKSIR5QkVSU0JJUVVOSIJVNUVRU0JaSUUVGRVRVbE9TVk5VWtGRFNVOU9SVk1nVU1PYVFreEpRMEZUTVVzd1NRWURWUWFMREVKR VNWSkZRMESkdzVOT0lFukZIRlJGUTA1UFRFOUhh3NDZCVXICRVJlTQk1RU0JKVGtaUFRdlaVFBes9NcnV6ZUxBSmphQmhjWXprQ21NRnc1eFdjN2s2UElwUzgyaGVGQjZSTis0UkdZUDE0OVZlTkdTUXJTMFdxSVhyWEpDSExRYjVjN0hCZU5wbTBDQXdFQUFhT0NBbEV3Z2dKTk1Ba0dBMVVKRXdRQ01BQXdnWUUVHQ0NzR0FRVUZCd0VCQkhVd2N6QTdCZ2dyQmdFRkJRy3dBWVI2YUhsMGNEb3ZMMjlkYzNCamlyMXdMbU5sY25RdVptNXRkQzVsY3k5dlkzTndMMDlqReM4yWSg0IWIIA9MSk5faUf6nD1vsaofBrTAyj8pLTTocGOs7YRE3EdEL_s4s8H4BlJo95ex_dpHdIHrV2gTBAevQHMKpIM7aUfLgyQ4o5ED8X9zFPycz7ofGXW9DoD2VJQOIKSdmPJYABeZlJBOPL8o47GSNuxrdWHLNFwlq2sXI_lzoGhWLCiPFgf4ZG2Pr2Sg3lu7olZRyGiXrYa5E2XhnNGC5fUIWtaxu8JrWbdkJR4OyONjBeArttK8YCgaEuwlzwJiA
```

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

3 Descripción del API

3.1 Grupo Invoice

Los recursos ubicados bajo la ruta /invoice están relacionados con las acciones relacionadas con el envío y gestión de facturas.

3.1.1. Invoice (/invoice)

3.1.1.1. Remitir una factura [POST]

Devuelve los datos de registro de la factura remitida.

- API Invoice

```
/v1/invoices
```

- Petición (InvoiceRequestType)

```
{
  "invoiceFile": {
    "mime": "text/xml",
    "content":
"PD94bWwgdmVyc2lvbjo0iMS4w...liBlb+PC94YWRlczpRdWFSaWZ5aW5n
UHJvcGVydGllcz48L2RzOk9iamVjdD48L2RzOINpZ25hdHVyZT48L2ZlOkZh
Y3R1cmFIPg==",
    "name": "factura_con_firma.xml"
  }
}
```

- Content: factura a remitir en base64

- Response 200 (application/json)

- Body

```
{
  "receivingUnit": {
    "dire": {
      "name": "EMPRESA S.A.",
      "code": "ES99999999R0000"
    },
    "role": "receiver"
  },
  "additionalAdministrativeCentres": [],
  "invoiceNumber": "1111ee",
  "registryNumber": "201901027100",
  "invoiceSeriesCode": "1111111ee",
  "amount": "1384.58",
```



```
{
  "currency": "EUR",
  "receptionDate": "2019-04-09T14:26:02+02:00",
  "issueDate": "2019-01-23T00:00:00+01:00",
  "statusInfo": {
    "name": "Registrada",
    "description": "Registrada",
    "code": "1200",
    "modificationDate": "2019-04-09T14:26:03+02:00"
  }
}
```

3.1.1.2. Descargar el archivo de la factura [GET]

Descarga el fichero de la factura.

- API Pública

/v1/invoices/{registry}/files

- Parámetros de path
 - registry (string, obligatorio): número de registro de la factura a descargar.
- Parámetros de query string:
 - Validate-signature (string, opcional): se puede solicitar la validación de la firma de la factura (indicando: “true” o “false”).
- Response 200 (application/json)
 - Body

[illegible]

3.1.1.3. Actualizar el estado de una factura [PATCH]

Actualización del estado de una factura. Previo a poder realizar el primer cambio de estado es necesario tener descargada la factura.

- API Pública

```
/v1/invoices/{registry}
```

- Parámetros de path
 - registry (string, obligatorio): número de registro de la factura a tramitar.
- Petición (UpdateInvoiceType)

```
{
  "status": "2500",
  "reason": "pasamos a pagada",
  "comment": "pruebas"
}
```

- Status: Los posibles estados de la factura son: **registrada** (1100), **confirmada** (1300), **pagada** (2500), **rechazada** (2600) y **anulada** (3100).
 - Se ha de especificar un motivo (reason). El comentario es opcional.
- Response 200 (application/json)
 - Body

```
{
  "receivingUnit": {
    "dire": {
      "name": "Kung Fu Fashion",
      "code": "ES99999999R0000"
    },
    "role": "receiver"
  },
  "additionalAdministrativeCentres": [],
  "invoiceNumber": "1111ee",
  "registryNumber": "201901027101",
  "invoiceSeriesCode": "1111111ee",
  "amount": "1384.58",
  "currency": "EUR",
  "receptionDate": "2019-04-09T14:29:20+02:00",
  "issueDate": "2019-01-23T00:00:00+01:00",
  "statusInfo": {
    "name": "Confirmada",
    "description": "Confirmada",
    "code": "1300",
    "modificationDate": "2019-04-09T14:56:10+02:00"
  }
}
```

3.1.1.4. Ver detalle de la factura [GET]

Devuelve toda la información de la factura.

- API Pública

```
/v1/invoices/{registry}/
```

- Parámetros de path
 - registry (string, obligatorio): número de registro de la factura a consultar.
- Response 200 (application/json)
 - Body

```
{
  "receivingUnit": {
    "dire": {
      "name": "Empresa",
      "code": "ES99999999R0000"
    },
    "role": "receiver"
  },
  "additionalAdministrativeCentres": [],
  "invoiceNumber": "1111ee",
  "registryNumber": "201901027101",
  "invoiceSeriesCode": "1111111ee",
  "amount": "1384.58",
  "currency": "EUR",
  "receptionDate": "2019-04-09T14:29:20+02:00",
  "issueDate": "2019-01-23T00:00:00+01:00",
  "statusInfo": {
    "name": "Registrada",
    "description": "Registrada",
    "code": "1200",
    "modificationDate": "2019-04-09T14:29:21+02:00"
  }
}
```

3.1.1.5. Obtener listado de facturas [GET]

Devuelve un listado paginado con las facturas de la ESF.

El listado tiene un límite por defecto de 1000 facturas, sin embargo, se puede especificar otro valor mediante el parámetro de query “limit”; si las facturas existentes excedieran el límite se deberá realizar otra petición indicando mediante parámetro de query las páginas siguientes.

- API Invoice

```
/v1/invoices
```



- Parámetros de query string
 - code (string, opcional): código de la unidad DIRe.
 - status (string, opcional): estado de la factura.
 - page (numérico, opcional); número de páginas.
 - limit (numérico, opcional): límite de registros a obtener.
- Response 200 (application/json) para obtener los detalles:
 - Body

```
{
  "items": [
    {
      "receivingUnit": {
        "dire": {
          "name": "Empresa",
          "code": "ES99999999R0000"
        },
        "role": "receiver"
      },
      "invoiceNumber": "2",
      "invoiceSeriesCode": "SERIE",
      "amount": "1384.58",
      "currency": "EUR",
      "receptionDate": "2018-12-18T15:56:57+01:00",
      "statusInfo": {
        "name": "Registrada",
        "description": "Registrada",
        "code": "1200",
        "modificationDate": "2018-12-18T15:56:59+01:00"
      }
    },
    {
      "receivingUnit": {
        "dire": {
          "name": " Empresa ",
          "code": " ES99999999R0000"
        },
        "role": "receiver"
      },
      "invoiceNumber": "2",
      "invoiceSeriesCode": "SERIE",
      "amount": "1384.58",
      "currency": "EUR",
      "receptionDate": "2018-12-18T16:19:52+01:00",
      "statusInfo": {
        "name": "Registrada",
        "description": "Registrada",
        "code": "1200",
        "modificationDate": "2018-12-18T16:19:55+01:00"
      }
    }
  ]
}
```


$$\left. \begin{array}{l} \} \\ \} \\] \\ \} \end{array} \right\}$$

3.1.1.6. Validar firma de factura [POST]

Devuelve la validación de la firma electrónica contenida en la factura. Las facturas en FACeB2B no deben ir obligatoriamente firmadas, es una cuestión opcional, pero en caso de ir firmadas, se puede validar dicha firma.

- API Consumo

/v1/invoices/{registry}/signature-validation

- **Parámetros del path:**
 - **registry** (string, obligatorio): número de registro de la factura a consultar.
- **Petición (InvoiceFileType, obligatorio)**

```
{
  "mime": "text/xml",
  "content":
"PD94bWwgdmVyc2l2bWVudWZREYXRhT2JqZWNOU
HJvcGVydGllcz48L3hhZGVzOINpZ25lZFB5b3BlcnRpZXM+PC94YWRLczpRd
WFsaWZ5aW5nUHJvcGVydGllcz48L2RzOk9iamVjdD48L2RzOINpZ25hdH
VyZT48L2ZlOkZhY3R1cmFIPg==",
  "name": "nombre_factura.xml"
}
```

- Content: factura a validar en base64.
- Response 200 (application/json)
 - Body

[illegible]

```
}
}
```

3.1.1.7. Obtener detalle de una factura registrada [GET]

Consulta del detalle de la factura.

- API Pública

```
/v1/invoices/{registry}
```

- Parámetros de path
 - registry (string, obligatorio): número de registro de la factura a consultar.
- Response 200 (application/json)
 - Body

```
{
  "receivingUnit": {
    "dire": {
      "name": "Empresa",
      "code": "ES99999999R0000"
    },
    "role": "receiver"
  },
  "additionalAdministrativeCentres": [],
  "invoiceNumber": "2",
  "registryNumber": "201801202376",
  "invoiceSeriesCode": "SERIE",
  "amount": "1384.58",
  "currency": "EUR",
  "receptionDate": "2018-12-19T12:25:12+01:00",
  "issueDate": "2013-01-17T00:00:00+01:00",
  "statusInfo": {
    "name": "Registrada",
    "description": "Registrada",
    "code": "1200",
    "modificationDate": "2018-12-19T12:25:14+01:00"
  }
}
```

3.1.2. Cancel Request (/invoices)

3.1.2.1. Obtener las solicitudes de anulación de facturas [GET]

Devuelve un listado con las solicitudes de anulación de las facturas.

- API Invoice

```
/v1/invoices/cancellation-request
```

- Parámetros de path
 - code (string, opcional): código DIRe de la unidad receptora.
- Response 200 (application/json)
 - Body

```
[  
  {"registry": "201901027101"},  
  {"registry": "201901033460"}  
]
```

3.1.2.2. Solicitar la cancelación de una factura [POST]

Método para solicitar la anulación de una factura remitida. Dicha solicitud deberá ser gestionada por la Plataforma de Facturación que gestiona la factura. Si la solicitud se acepta la factura quedará anulada, si la solicitud se cancela, la factura seguirá su curso.

- API Invoice

```
/v1/invoices/{registry}/cancellation-request
```

- Parámetros de path
 - registry (string, obligatorio): número de registro de la factura que tiene una solicitud de cancelación.
- Petición (CancelRequestType)

```
{  
  "reason": "C002",  
  "comment": "prueba solicitud cancelacion",  
  "commentReject": "prueba"  
}
```

- Response 200 (application/json) para obtener los detalles:
 - Body

```
{
  "id": 41,
  "reason": "C002",
  "commentRequest": "prueba solicitud cancelacion",
  "invoice": {"registryNumber": "201901027101"},
  "processed": false
}
```

3.1.2.3. Aceptar/Rechazar una solicitud de anulación [PATCH]

Actualiza el estado de una solicitud de anulación de una factura.

- API Consumo

```
/v1/invoices/{registry}/cancellation-request/{cancelRequest}
```

- Parámetros del path:
 - registry (string, obligatorio): número de registro de la factura a consultar.
 - cancelRequest (string, obligatorio): Id de la solicitud de cancelación.
- Petición (CancelRequestType)

```
{
  "action": "accept",
  "comment": "comentario"
}
```

- Response 200 (application/json)
 - Body

```
{
  "id": 61,
  "reason": "C002",
  "commentRequest": "prueba solicitud cancelacion",
  "invoice": {"registryNumber": "201901033460"},
  "processed": true
}
```

4 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.JsonWebSignature.Header header
= new com.google.api.client.json.webtoken.JsonWebSignature.Header();
header.setAlgorithm("RS256");
//header.setX509Certificates(Arrays.asList(new String[]
{Base64.getEncoder().encodeToString((ks.getCertificate(keyAlias).getEnc
oded()))});
def cert_enc =
Base64.getEncoder().encodeToString(ks.getCertificate(keyAlias).getEnco
ded());
```




```
def res = [];  
res.add(cert_enc);  
List certs = Arrays.asList(res);  
header.setX509Certificates(certs);  
com.google.api.client.json.webtoken.JsonWebToken.Payload payload =  
new com.google.api.client.json.webtoken.JsonWebToken.Payload();  
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.JsonWebSignature.signUsingRsaSha  
a256((PrivateKey)ks.getKey(keyAlias, "alias_certificado".toCharArray()),  
JSON_FACTORY, header, payload);  
context.testCase.setPropertyValue('token', new String(token));  
log.info(token)
```