Manual de Uso PyAfipWs: Interfase y herramientas para servicios web AFIP (Factura Electrónica y otros)

- **Documentación Componentes y Herramientas Generales:**
  - PyAfipWs: Interfase simil OCX con otros lenguajes (VB, VFP, Cobol ...) Costos y Condiciones
  - Manual: Documentación, Información Importante, Certificados, Errores Frecuentes
  - Herramienta "universal": archivos de intercambio TXT Cobol, DBF dBase/FoxPro, JSON PHP/Java
  - PyFEPDF: Generador de Factura Electrónica en formato PDF
- **Factura Electrónica - Servicios Web AFIP:**
  - Mercado Interno: Factura Electrónica A/B/C/M WSFEv1+ (RG2485/3067/3668/3749/4004/4109/4367)
  - Matrix (codificación productos): Factura Electrónica A/B con detalle (RG2904/3536)
  - Bienes de Capital: Bonos Fiscales Electrónicos - Factura Electrónica A (RG2557)
  - Exportación: Factura Electrónica E Exportadores (RG2758 RG3689 RG4401)
  - Turismo: Comprobantes Factura Electrónica T WSCT CAE/CAEA (RG3971)
  - FE Cred (FCE): Factura Electrónica Crédito MiPyMEs (RG4367)
  - Código de Autorización Electrónico Anticipado CAEA
- **Agropecuario - Servicios web AFIP:**
  - Código Trazabilidad de Granos: Transporte de granos WSCTGv4 (RG2806 RG3113 RG3493)
  - Liquidación y Certificación de Granos: WSLPGv1.17 F. C1116 A / B / RT (RG3419 RG3690 RG3691)
  - Liquidación de Tabaco Verde: WSLTVv1.3
  - Liquidación Única Mensual Lechería: WSLUMv1.3
  - Liquidación Sector Pecuario: Hacienda, Compra directa, Carne WSLSPv1.3
  - Remito Electrónico Cárnico: WSRemCarne (RG4256/18 y RG4303/18)
  - Remito Electrónica Harinero: WSRemHarina - (RG 4519/19)
  - Remito Electrónico Azúcar y Derivados: WSRemAzucar - (RG 4519/19)
  - Carta de Porte: WSCPE - (RG 5017/21)
  - Carta de Porte Derivados Granarios: WSCPEv2 - (RG 5235/22) **Nuevo!**
- **Otros webservices y utilidades AFIP**
  - Constatación de Comprobantes CAI, CAE, CAEA (WSCDC)
  - Padron Contribuyentes AFIP: Constancia de Inscripción RG1817/2005 WS-SR-Padron
  - Consulta de Operaciones Cambiarias: Compra de Divisas (WSCOC)
  - SIRE_CertificadoRetencionElectronica: Certificado de retención electrónica del Impuesto al Valor Agregado (SIRE-WS) RG4523/19
- **Webservices provinciales: ARBA (Prov. Bs.As.), AGIP (C.A.B.A), API (Sta.Fe), DGR (Córdoba):**
  - Remito Electrónico: COT Código de Operaciones de Translado (ARBA, API, AGIP, DGR)
  - Ingresos Brutos: Consulta de alícuotas WS DFE IIBB ARBA
- **SNT: Sistema Nacional de Trazabilidad ANMAT, SEDRONAR, SENASA**
  - Trazabilidad de Medicamentos: ANMAT Disposición 3683/2011
  - Trazabilidad de Productos Médicos: ANMAT Disposición 2303/2014 y 2175/14
  - Trazabilidad de Precursores Químicos: RENPRE SEDRONAR Resolución 900/12
  - Trazabilidad de Productos Fitosanitarios: SENASA Resolución 369/13

    - ♦ Trazabilidad de Productos Fitosanitarios: SENASA Resolución 369/21 (WS_INFO_EMPRESAS / WS_DATOS_GENERALES)
- **Aplicativos Genéricos y Herramientas Avanzadas:**
    - ♦ PyRece: Aplicativo visual simil SIAP - RECE (CSV, PDF, Email)
    - ♦ FE.py: Herramienta universal, unificada e integrada
    - ♦ FacturaLibre: Aplicacion online (web2py)
    - ♦ PyFactura Aplicativo visual y simple (gui2py) para CAE y PDF factura electrónica
    - ♦ LibPyAfipWs: Biblioteca DLL para lenguajes C / C++ y similares
    - ♦ Factura Electrónica en Python: Información Técnica (SOAP, XML, PDF, DBF, etc.)

Emisión y almacenamiento electrónico de comprobantes originales AFIP - Argentina. Resolución General 1361/02, 2265/07, 2289/07, 2485/08, RG2570/09, 2557/09, 2758/10, 2904/10, 3067/11, 3419/12, 3536/13, 3571/13, RG3668/14, RG3689/14, RG3749/15, RG3779/15, RG4109-E/17, RG4367/18, RG4401/19

2008-2019 © Mariano Reingart ? Febrero de 2019

Índice

Importante: leer primero antes de comenzar

Historial de Cambios

Para el detalle completo de los ajustes, modificaciones y mejoras al código fuente ver Versiones Publicacadas y Conjuntos de Cambios (Sitio del Proyecto en GitHub)

Noviembre 2022:

- Por RG5259 y RG5264 se implementa un nuevo método para la consulta de actividades e incorporaron una nueva estructura para informar actividad, también se incluye en la consulta.

Julio 2021:

- Implementación de código QR a partir del 01/03/2021 según RG4892/2020 Procedimiento. Régimen especial de emisión y almacenamiento electrónico de comprobantes originales. R.G. N° 4.291. Implementación del código ?QR?

Julio 2020:

- AFIP anuncia la desafectación de protocolos TLS antiguos (v1.0 y 1.1) a partir del 27/7/2020 según cronograma : Se actualizan librerías a tal efecto.

Junio 2020:

- RG4540/19 Procedimiento, Facturación. Emisión de notas de crédito y/o débito: Se agregan métodos WSFEv1.AgregarPeriodoComprobantesAsociados, WSMTXCA.AgregarPeriodoComprobantesAsociados

Febrero 2019:

- RG4367/18 Factura de Crédito Electrónica MiPyMEs FEv2.13 FCE: Se agrega fecha en método WSFEv1.AgregarCmpAsoc y datos opcionales CBU
- RG4401/19 WSFEXv1.5.0: Se agrega método WSFEXv1.GetParamMonConCotizacion y ejemplo RECEX1 /monctz para consultar cotización moneda aduana por fecha

Agosto 2018:

- RG 4109-E Bienes Muebles Registrables: Se agrega método WSFEv1.AgregarComprador y formato intercambio texto / dbf

Febrero 2018:

- Se incluye la nueva Autoridad Certificante -CA- que utilizá AFIP

Agosto 2017:

- Se agrega RG 4109-E COMPG_v2_10: Venta de Bienes Muebles Registrables. Se agrega método AgregarComprador y formato Registro Compradores TXT, Compradores.DBF y pseudocódigo de ejemplo WSFEv1.

Marzo 2017:

- Se agrega documentación sobre Actualizaciones ("Service Pack 2"), WSFEv2.9 (Cuit en Cbte.Asoc. y RG 4004-E Alquileres / Impuesto a las Ganancias) y Release v0.5 (WSMTX)
- Nuevo campo CUIT en método `AgregarCmpAsoc` para WSFEv1 y WSMTX

Febrero 2017:

- Documentación sobre Instaladores Firmados Digitalmente (Editor Comprobado)

Septiembre 2016:

- Se agrega documentación sobre Actualizaciones ("Service Pack 1")

Agosto 2016 (hasta revisión interna 739):

- Se agregan comentarios sobre Eventos y COMPGv2.8, para más info ver Novedades WSFEv1
- Se documentan métodos para solicitud de múltiples CAE por envio
- Se agregan mensajes de errores frecuentes y posibles soluciones (Protocolo SSL, CERTIFICATE_VERIFY_FAILED, fallas soap, service unavailable, tag not found, IO error, sincronizar relojes, etc.)
- Se ajusta cambio de generación de claves con longitud de 2048 bits por cambio en AFIP
- Se ajustan links por desafectación de Google Code (histórico)
- Se agregan aclaraciones sobre obtención de CAE, reproceso (incluyendo ejemplo avanzado para VFP y VB.NET), cache, etc.
- Se agrega documentación sobre métodos para certificados: automatización de generación y análisis (por ej. vencimiento)
- Se agrega datos opcionales RG 3779 RG 3668

Junio 2015 (hasta revisión interna 638):

- Se agrega Aclaraciones Generales PDF (IVA, leyendas RG2845 Art 27, Comprobante Autorizado, etc.)
- Se agrega logotipo AFIP al PDF y otras modificaciones a la plantilla de facturas electrónicas
- Se agrega atributos auxiliares y ejemplos de campos personalizables a generación de PDF
- Se unifico formato intercambio PDF y se agrega descripción de cada campo.
- Se agrega video explicativo WSASS y actualiza instructivo certificados
- Se actualizan parámetros WSAA SignTRA (passphrase y contenido archivos)
- Se agregan ejemplos simples de reutilización de TA para VFP y VB
- Se ajustan links al código fuente hacia repositorio GitHub por baja de GoogleCode
- Se agregan ejemplos de RECE1 y PYFEPDF
- Se ajusta descripción obtención de CAE WSFEv1
- Se agregan errores internos de AFIP
- Se realizan correcciones menores de tipeo (motivos_obs, base_imp)
- Se actualiza infomación para Factura C Exentos (RG3749)

Abril 2015 (hasta revisión interna 504):

- Se agrega metodo WSFEv1.EstablecerCampoFactura
- Se ajusta formato de intercambio WSMTXCA
- Se ajustan instaladores
- Se agregan más errores frecuentes de AFIP

Octubre 2014 (hasta revisión interna 479):

- Se agrega información y ejemplo datos opcionales RG 3668 WSFEv1
- Se agrega campos opcionales RG3668 al formato de RECE1
- Se ajusta manejo de errores (ejemplo VFP y simplificacion)
- Se agregan errores internos de AFIP (shared pool ora, account locked...), punto de venta y temas menores

Diciembre 2013 (hasta revisión interna 389):

- Se ajusta tabla comparativa (RG3536 y RG3571)
- Se agregan errores internos de AFIP ORA-12535
- Se ajusta formato RECE1 y temas menores (estilo)

Abril 2013 (hasta revisión interna 343):

- Se agrega ejemplo WSAA sobre reuso del ticket de acceso
- Se agrega ejemplo WSAA sobre verificación de canal seguro
- Se actualiza info sobre autoridades certificante
- Se actualiza info sobre renovación de certificados
- Se ajusta links WSBFE
- Cambios menores

Noviembre/Diciembre 2011 (hasta revisión interna 319):

- Se ajusta sección "WSFEXv1 Versión 1"
- Se agrega secciones "Ajustes de WSFEX a WSFEXv1" y "RECEX vs RECEX1"

- Se ajusta ejemplo WSFEX (WSFEXv1)
- Se actualiza "Mensajes XML" con instrucciones para enviar campos vacios o nulos
- Cambios menores (`coe.alreadyAuthenticated`)

Julio 2011 (hasta revisión interna 281):

- Se agrega sección "Version 1 (WSFEXv1)" a WSFEX y se ajusta al nuevo webservice
- Se agrega seccion "PyI25: Generador de Códigos de Barras"
- Se agrega sección "PyEmail: Envio de correo electrónico"
- Se agrega sección "Nuevas Validaciones" a WSFEv1
- Se ajusta secciones Instalación (descargas, instalación manual/silenciosa)

Abril/Mayo 2011 (hasta revisión interna 221):

- Se agrega sección "Manejo de Excepciones"
- Se agrega sección "Verificación del Canal de Comunicación Seguro"
- Se agrega sección "Tabla Comparativa Webservices"
- Se agrega sección "Reutilización de Ticket de Acceso" a WSAA
- Se agrega sección "Obtención de atributos avanzados" a WSAA y WSFEv1
- Se agrega sección "Factura C Monotributo" a WSFEv1
- Se actualiza secciones WSAA (métodos, atributos, errores)
- Se actualiza secciones WSFEv1 (métodos, atributos, errores)
- Se actualiza secciones WSFEX (métodos, atributos, errores)
- Se ajusta secciones Instalación (descargas, instalación manual/silenciosa)

Febrero/Marzo 2011 (hasta revisión interna 155):

- Se agrega sección WSMTXCA (objeto, métodos, atributos, ejemplo, errores) y RECEM
- Se agrega sección "Ejemplo Interactivo de Autenticación (ej-WSAA)"
- Se actualiza sección WSFEv1 y RECE1
- Se agrega sección "PyFEPDF" (interfaz COM, herramient CLI, archivos de texto, tablas DBF, configuración y diseñador visual)
- Se ajusta sección "Interfase por archivos de texto símil SIAP - RECE" (configuración, tablas DBF)

Ver historial completo desde revisión 1 -2009- (solo usuarios autorizados)

Funcionamiento del campo ID

Importante: El ID no aplica a los nuevos webservices (WSFEv1 ni WSMTXCA), igualmente se aconseja leer y seguir las siguientes recomendaciones. Si bien los nuevos webservices tampoco tienen reprocesamiento, la interfaz lo realiza internamente, verificando los datos enviados y recibidos (ver información adicional en cada webservice).

El ID es el Identificador del requerimiento requerido por AFIP, un número interno de secuencia controlado por el emisor, que permite identificar de manera única cada operación de autorización (solicitud de CAE). Este dato es de vital importancia para poder recuperar un CAE frente a problemas de comunicación o fallas del hardware/software. Sin el, és imposible recuperar un CAE y se puede llegar a bloquear todo el circuito de facturación electrónica en los servidores de AFIP. Por ello es recomendable que sea un dato propio del sistema de facturación, almacenado en un soporte permanente (base de datos en el disco rígido o similares)

El ID debe ser único para cada comprobante autorizado, y debe cambiar si hubo rechazo previo. Los ID no deben reutilizarse salvo para realizar reprocesos (recuperar un CAE ya emitido). La AFIP almacena las transacciones según su ID, a mismo ID, mismo resultado y CAE (independientemente de los otros datos de la operación de transacción).

En el caso de fallas, los webservices poseen métodos para recuperar el último ID informado (por ej., ver UltNro o GetLastID)

Si bien los ejemplos (incluyendo los oficiales) utilizan generalmente el último ID + 1, esto es solo para ilustrar el método en general, no es recomendable su uso en producción, además de requerir comunicación extra (no es obligatorio llamar al método último ID), ya que puede generar inconvenientes en caso de múltiples equipos solicitando autorización simultáneamente, fallas de comunicación o problemas de almacenamiento. Sería recomendable, por ej, que el ID sea un campo autonumérico, autoincremental o serial secuencial que acompañe a la factura.

El ID es arbitrario y no tiene relación con el nº de factura. La única condición es que debe ser único, pero no es necesario que sea creciente o secuencial.

Validez de Comprobantes Electronicos emitidos

El uso incorrecto de los servicios web (no respetando el funcionamiento de los campos, en especial el ID, reproceso, motivos de observación y errores), puede ocasionar comportamientos erráticos (facturas no autorizadas o autorizadas con otros datos). Se recomienda revisar el funcionamiento general de su sistema en producción utilizando las validaciones que prevee la AFIP:

- Consulta interactiva de CAE: https://serviciosweb.afip.gob.ar/genericos/comprobantes/cae.aspx
- Consulta interactiva de CAE Anticipado: https://serviciosweb.afip.gob.ar/genericos/comprobantes/caea.aspx
- Por Clave Fiscal, servicio Verificación de validez de comprobantes emitidos (solo receptores)
- Por servicio web para Facturas E, A y B (Bono Fiscal): Método Recuperar comprobante (para facturas de exportación, solo emisores)

Para más información y automatización del proceso ver servicio web Constatación de Comprobantes WSCDC.

Mensajes XML

Los mensajes XML contienen la comunicación transmitida entre el sistema y la AFIP en el formato de intercambio de los servicios web:

- Atributo `XmlRequest` (requerimiento.xml): incluye credenciales de acceso y la solicitud remota (método a utilizar, parámetros, etc.)
- Atributo `XmlResponse` (respuesta.xml): incluye los datos devueltos por el proceso remoto, incluyendo mensajes de error (si corresponde)

Almacenar estos archivos es de vital importancia para el registro de las operaciones realizadas por webservice y su eventual verificación, depuración o solución de inconvenientes. La información contenida en estos archivos es solo relevante a los desarrolladores, no es necesario que sean enviados al usuario final.

Dependiendo de la interfaz, se pueden recuperar estos mensajes en forma de cadenas de texto para su posterior almacenamiento, o generar los archivos correspondientes.

Para más información ver atributos XmlRequest y XmlResponse o parámetros /xml --trace:

- Guardar `WSFEv1.XmlRequest` y `WSFEv1.XmlResponse` (ver Manejo de Excepciones para una rutina completa)
- `RECE1.EXE /xml --trace` generará los archivos con la extensión xml en el directorio de la herramienta (ver Parámetros para más información)

Nota de Seguridad: los mensajes pueden incluir datos "sensibles" (credenciales de acceso a AFIP: token y sign), que pueden ser válidos por varias horas aún después de efectuada la comunicación. En caso compartir estos archivos (por ej en el foro público de consultas), se deberían tomar las medidas necesarias para evitar inconvenientes (recortar el campo sign es suficiente para que no se puedan reutilizar las credenciales de acceso).

Importante: En ciertos casos, para determinados webservices, AFIP solicita tag vacios (por ej. `permiso_existente` en WSFEXv1), por lo que se ha ajustado en la biblioteca de comunicación SOAP (versión 1.27 y superiores) para manejar tres escenarios:

- Enviar un dato, por ej `permiso_existente = "S"`: `<permiso_existente>S</permiso_existente>`
- Enviar un tag vacio completando con string vacio, por ej. `permiso_existente = ""`: `<permiso_existente></permiso_existente>`
- No enviar un tag (valor NULL, NIL, None, Void, etc.), por ej. `permiso_existente = NULL` (no genera XML)

En caso de no seguir las especificaciones de AFIP, el webservice retornará un mensaje de error del tipo: `Server was unable to read request. ---> There is an > error in XML document (5, 1353). ---> Input string was not in a correct format` donde indica revisar el `XmlRequest`, línea 5, columna 1353 (en este caso puntual), para corregir el dato enviado.

Documentación Oficial AFIP

Recomendamos revisar y seguir los lineamientos y normativa de AFIP:

- http://www.afip.gov.ar/fe/
- http://www.afip.gov.ar/ws/

Recordamos que la interfaz y bibliotecas (PyAfipWs) no realizan ningún tipo de chequeo o validación adicional (ya que esta tarea la realizan los servidores de AFIP), y tampoco convierte entre distintos tipos de datos (salvo conversión trivial a cadenas -strings- para ser enviadas por XML), por lo que es responsabilidad de la aplicación principal cumplir con las especificaciones y legislación vigente referidas a los servicios web de AFIP, proporcionando los datos correctos que correspondan.

Modos Homologación y Producción

Por diseño, el instalador para Windows publicado de la interfaz COM (pyafipws.py) y herramientas por línea de comando (rece.py, recex.py, receb.py, rece1.py) funcionará inicialmente solo en modo homologación (testing), ya que su objetivo es simplemente facilitar las pruebas para que los usuarios que no concen Python

puedan evaluar el producto antes de interiorizarse en el lenguaje y/o contratar el soporte comercial.

Con los instaladores de evaluación, en modo de homologación (pruebas), se ignorará las URLs proporcionadas a CallWSAA, Conectar o especificadas en rece.ini. En ese caso, para activar modo producción (definitivo) es necesario ajustar el código fuente, ver instrucciones de Instalación desde Código Fuente (video explicativo). Ofrecemos soporte comercial, contemplando el acceso al instalador compilado & ajustado para producción, para los clientes que así lo requieran. Ver Costos y Condiciones

Configuración para Producción

Para entrar en el ambiente de producción de AFIP, además de utilizar el instalador correcto (no el de evaluación, ver arriba), se debe usar las URL/WSDL correspondientes para cada webservice (modo homologación deshabilitado):

- WSAA: https://wsaa.afip.gov.ar/ws/services/LoginCms?wsdl
- WSFE: https://servicios1.afip.gov.ar/wsfe/service.asmx *-obsoleto desde 2011 por AFIP-*
- WSFEv1: https://servicios1.afip.gov.ar/wsfev1/service.asmx?WSDL
- WSFEXv1: https://servicios1.afip.gov.ar/wsfexv1/service.asmx?WSDL
- WSMTXCA: https://serviciosjava.afip.gob.ar/wsmtxca/services/MTXCAService?wsdl
- WSBFEv1: https://servicios1.afip.gov.ar/wsbfev1/service.asmx?WSDL

Ejemplo en el `rece.ini`:

```
[WSAA]
CERT=reingart.crt
PRIVATEKEY=reingart.key
URL=https://wsaa.afip.gov.ar/ws/services/LoginCms
CACERT=conf/afip_ca_info.crt

[WSFEv1]
CUIT=20267565393
ENTRADA=entrada.txt
SALIDA=salida.txt
URL=https://servicios1.afip.gov.ar/wsfev1/service.asmx?WSDL
```

Ejemplo en Visual Basic o similar (segundo parámetro del método `Conectar`):

```
Certificado = "reingart.crt" ' certificado de producción
ClavePrivada = "reingart.key" ' clave privada de producción
cms = WSAA.SignTRA(WSAA.CreateTRA("wsfe"), Certificado, ClavePrivada)
...

cache = ""
wsaa_url = "https://wsaa.afip.gov.ar/ws/services/LoginCms?wsdl"
ok = WSAA.Conectar(cache, wsaa_url)
ta = WSAA.LoginCMS(cms)

wsfev1_url = "https://servicios1.afip.gov.ar/wsfev1/service.asmx?WSDL"
ok = WSFEv1.Conectar(cache, wsfev1_wsdl)

WSFEv1.Cuit = "20267565393"  ' CUIT del emisor (segun certificado)
...
```

Luego de configurar las opciones para producción, se puede verificar el correcto funcionamiento de las credenciales y URL utilizando los métodos auxiliares , por ej. consultando los puntos de venta habilitados:

- parametros RECE1 /ptosventa
- método ParamGetPtosVenta

Aclaración: No es obligatorio usar el instalador ni realizar compilación, se puede bajar, revisar, modificar y ejecutar directamente el código fuente Python publicado en modo homologación o producción.

Importante: Los CAE obtenidos en modo homologación no tienen validez fiscal.

Advertencia: Teóricamente no debería ser posible conectarse a producción con certificados de homologación (y viceversa), pero en la práctica han sucedido casos (obteniendo CAE de homologación sin validez fiscal), por lo que recomendamos extremar las medidas de seguridad y control (verificando certificados y configuración de URL de servidores) para generar las facturas en el ambiente correcto. Ver Verificar Validez CAE y Comprobación WSAA

Manejo de Excepciones

Las versiones recientes de WSAA (2.02), WSFEv1 (1.10), WSMTXCA (1.01) y WSFEX (1.26/1.23) incluyen mejoras y unificaciòn en el manejo de excepciones/errores (atributos `LanzarExcepciones`, `Excepcion` y `Traceback`), permitiendo un control más simple de los `"Unexpected Python Error"` (errores no esperados que informa Python, el lenguaje de programación de la interfaz, generalmente por problemas de conexión o tipos de datos inválidos) sobre todo para lenguajes que no soportar mensajes de Error COM (previniendo mensajes por errores no controlados como *Error fatal: código de excepcion=C0000005* y el cierre del programa).

Dependiendo de cada caso (en especial para lenguajes de programación legados), es conveniente deshabilitar los errores no manejados (LanzarExcepciones = False) y controlar manualmente las excepciones y el valor retornado luego de llamar a cada método.

Ejemplo para Visual Basic:

```
Dim ta as Variant

WSAA.LanzarExcepciones = False
ta = WSAA.LoginCMS(cms)
If WSAA.Excepcion <> "" Then
    ' muestro al usuario solo el mensaje de error, no la traza:
    MsgBox WSAA.Excepcion, vbCritical, "Excepción"
    Exit Sub
End If
```

Mismo ejemplo para Visual Fox Pro:

```
WSAA.LanzarExcepciones = .F.
ta = WSAA.LoginCMS(cms)
IF LEN(WSAA.Excepcion) > 0 THEN
    ? WSAA.Traceback
    MESSAGEBOX(WSAA.Excepcion, 5 + 48, "Excepcion")
    CANCEL
ENDIF
```

También se puede verificar que la respuesta del método (en este caso en la variable ta, notar que es del tipo `Variant`) no sea nula, en ese caso se ha producido una excepción y el mensaje debería estar en el atributo Excepcion, con su traza en Traceback.

Nota: dependiendo de la version, los atributos Excepcion y Traceback pueden estar disponibles solo luego de ocurrido el error (utilizar las ulitmas versiones para poder consultar estos atributos en cualquier momento).

Es recomendable que se muestre al usuario solo la Excepción (la traza es información técnica que no tiene utilizad para el operador), y para facilitar la depuración, en todos los casos es útil guardar los Mensajes XML y excepción, ya que a veces los lenguajes de programación cortan el mensaje y se dificulta saber el motivo del error. Además, es recomendable siempre verificar que los objetos hayan sido creados (para no generar nuevos errores) y que estos sean compatibles con la versión que manejamos (si tenemos varias instalaciones distintas, para no solicitar atributos inexistentes)

Por ej. para VB (cambiar WSAA por el webservice que corresponda):

```
' verifico que el objeto haya sido creado:
If Not WSAA Is Nothing Then
  ' verifico que la versión del objeto sea compatible:
  If WSAA.Version >= "1.02a" Then
    ' creo el archivo guardo allí todos los atributos para depuración
    fd = FreeFile
    Open "c:\error.txt" For Append As fd
    Print #fd, WSAA.Excepcion
    Print #fd, WSAA.Traceback
    Print #fd, WSAA.XmlRequest
    Print #fd, WSAA.XmlResponse
    Close fd
  End If
End If
```

Ejemplo para VFP (cambiar WSAA por el webservice que corresponda):

```
*-- Depuración (grabar a un archivo los datos de prueba)
gnErrFile = FCREATE('c:\error.txt')
=FWRITE(gnErrFile, WSAA.Token + CHR(13))
=FWRITE(gnErrFile, WSAA.Sign + CHR(13))
=FWRITE(gnErrFile, WSAA.XmlRequest + CHR(13))
=FWRITE(gnErrFile, WSAA.XmlResponse + CHR(13))
=FWRITE(gnErrFile, WSAA.Excepcion + CHR(13))
=FWRITE(gnErrFile, WSAA.Traceback + CHR(13))
=FCLOSE(gnErrFile)
```

Error Handler

Importante: este método no es estrictamente necesario si su programa ya tiene incorporado el manejo de errores según el lenguaje de programación que utilicie (especialmente si utiliza el control de excepciones mencionado en la sección anterior)

Todas estas consideraciones no quitan que se deba tener previsto el controlador de errores habitual del lenguaje para capturar los errores graves (por ej. al no poder crear el objeto, enviar un tipo de dato incorrecto o llamar un método/atributo inexistente), ya sea utilizando:

- Visual Basic: `ON ERROR GOTO ...`

- Visual Fox Pro: `ON ERROR DO ...;`
- o bloques `TRY`, `EXCEPT`, `CATCH` o similares.

Lo mismo aplica a WSFEv1, WSMTXCA y WSFEX. Para más información ver los ejemplos actualizados.

Para VB

```
or GoTo ManejoError

r objeto interface (WSAA, WSFEv1, etc.)
AA = CreateObject("WSAA")
ar a los métodos de WSAA

ub

Error:

ubo error (tradicional, no controlado):

rar mensajes de Depuración en ventana de inmediato
 WSAA Is Nothing Then
 WSAA.Version >= "1.02a" Then
  Debug.Print WSAA.Excepcion
  Debug.Print WSAA.Traceback
  Debug.Print WSAA.XmlRequest
  Debug.Print WSAA.XmlResponse
d If

 WSFEv1 Is Nothing Then
 WSFEv1.Version >= "1.10a" Then
  Debug.Print WSFEv1.Excepcion
  Debug.Print WSFEv1.Traceback
  Debug.Print WSFEv1.XmlRequest
  Debug.Print WSFEv1.XmlResponse
  Debug.Print WSFEv1.DebugLog()
d If


ubo error:
Print Err.Description            ' descripción error afip
Print Err.Number - vbObjectError ' codigo error afip
 Case MsgBox(Err.Description, vbCritical + vbRetryCancel, "Error:" & Err.Number - vbObjectError & " en " & Err.
se vbRetry
  Debug.Assert False
  Resume
se vbCancel
  Debug.Print Err.Description
lect
```

Para VFP (con procedimiento para captura de errores automática):

```
#!
ON ERROR DO errhand2;
...
*-- Crear objeto interface Web Service de Factura Electrónica
WSFE = CREATEOBJECT("WSFEv1")
*-- llamar a los métodos de WSFE
...

*-- Procedimiento para manejar errores WSFE
```

```
PROCEDURE errhand2
    *--PARAMETER merror, mess, mess1, mprog, mlineno

    *-- Depuración (mostrar los mensajes de error en la ventana principal)

    ? WSFEv1.XmlRequest
    ? WSFEv1.XmlResponse
    ? WSFEv1.Excepcion
    ? WSFEv1.Traceback

    ? 'Error number: ' + LTRIM(STR(ERROR()))
    ? 'Error message: ' + MESSAGE()
    ? 'Line of code with error: ' + MESSAGE(1)
    ? 'Line number of error: ' + LTRIM(STR(LINENO()))
    ? 'Program with error: ' + PROGRAM()

    *-- Preguntar: Aceptar o cancelar?
    ch = MESSAGEBOX(WSFE.Excepcion, 5 + 48, "Error")
    IF ch = 2 && Cancelar
        ON ERROR
        CLEAR EVENTS
        CLOSE ALL
        RELEASE ALL
        CLEAR ALL
        CANCEL
    ENDIF
ENDPROC
```

Es posible seguir utilizando para las versiones anteriores o por compatibilidad hacia atrás, LanzarExcepciones = Verdadero (predeterminado), en dicho caso se generará un Error COM que deberá ser capturado por la aplicación, y el mensaje estará en `Err.Description` (para VB) o `MESSAGE()` (para VFP)

Time-Out

En casos de falla de los servidores de AFIP, la respuesta se puede demorar por lo que se puede configurar el parametro timeout (tiempo de espera) en el método `Conectar(cache, wsdl, proxy, wrapper, cacert, timeout)` para controlar el período y minimizar los bloqueos.

De manera predeterminada, el tiempo de espera en las últimas versiones es de 30 segundos, pero se pueden pasar otros valores como 300 para que espere hasta 5 minutos (si los servidores de AFIP están muy congestionados); o reducir a 15 o 10 para esperar un tiempo más reducido y dar una respuesta rápida al usuario (ver en ese caso Reproceso Automático para reintentar/consultar en caso de fallas en AFIP).

Es recomendable establecer la propiedad `LanzarExcepciones` a Falso y comprobar el valor de `Excepcion` (ver arriba), que generalmente informará el error del sistema operativo por "timeout". Varios webservices de AFIP tienen un método `Dummy` que se pueden combinar con el timeout para detectar si el servidor de AFIP esta funcionando correctamente o no (justamente ese método también puede fallar desde AFIP y es necesario controlar el tiempo de espera para un chequeo rápido).

Ver Errores de Conectividad y Errores internos de AFIP para mayor información sobre los problemas frecuentes con la comunicación hacia los servidores de AFIP.

Importante: Si no se utiliza los parámetros cache, proxy, wrapper y cacert, deberían ser pasados como string vacio "" (no saltear). Para poder utilizar estos parámetros se debería usar una versión actualizada de los componentes (2015)

Verificación del Canal de Comunicación Seguro

Por nueva Especificación Técnica WSAA 1.2.2
("Requerimientos de canal seguro comunicación para los CEE, WSAA y WSN") se debe verificar la identidad del servidor de AFIP (transformando lo que antes era una sugerencia (no documentada) en una "verificación obligatoria" de comprobar el certificado y URL del servidor)

Para ello los métodos `Conectar` reciben dos parámetros adicionales (actualmente solo para últimas versiones de desarrollo de WSAA 2.04 y WSFEv1 1.12 o superiror):

- wrapper: librería http con soporte avanzado de ssl (por el momento `"pycurl"`)
- cacert: ruta al certificado de la autoridad certificante a verificar `"geotrust.crt"` o `"comodo.crt"`. Puede utilizar el archivo con ambos certificados: afip_ca_info.crt que lo mantenemos actualizado con todos los certificados requeridos.

IMPORTANTE: es recomendable empezar a utilizar el archivo genérico de entidades certificantes, pasando `cacert=True` o `cacert="default"`. De este modo, si AFIP cambia el certificado o se vence alguna CA, es probable que el nuevo certificado esté en el archivo generico, por lo que no habría que modificar el código y agregarlo manualmente (siempre que se este utilizando una versión actualizada del proyecto)

Como solución provisoria se pueden descargar cacert.pem desde el siguiente link:
https://curl.haxx.se/docs/caextract.html Esos son los certificados de Firefox (Mozilla), que contiene todos los certificados existentes. Se debería reemplazar por el archivo afip_ca_info.crt

Ejemplo en Visual Basic:

```
""
 = "https://wsaa.afip.gov.ar/ws/services/LoginCms?wsdl"
 = ""                                          ' "pycurl" para situaciones especiales o versiones anteriores (inst
""                                             ' formato usuario:clave@servidor:puerto
= WSAA.InstallDir + "conf/afip_ca_info.crt" ' Indicar ruta completa (directorio conf instalador > 2.7)
ectar(cache, wsaa_url, proxy, wrapper, cacert)
```

Ejemplo en RECE.INI (solo se agrega los parámetros de configuración CACERT y WRAPPER, el resto se usa normalmente):

```
[WSAA]
URL=https://wsaa.afip.gov.ar/ws/services/LoginCms?wsdl
CACERT=conf/afip_ca_info.crt
#WRAPPER=pycurl
```

Nota: De realizar la verificación, debe agregar los parámetros a todas las secciones que se conecten con los webservices [WSAA], [WSFEv1], [WSLPG], etc., y usar un instalador actualizado.

Para verificar la Autoridad Certificante (CACERT), se recomienda probar el último ejemplo interactivo (cambiando httplib2 por pycurl para validar el nuevo certificado de wsaa). Para más información ver Ejemplo Interactivo de Autenticación (requiere instalación)

De encontrarse alguna inconsistencia, el método Conectar devolverá falso y en Excepción estará el motivo de error:

- No verifica el certificado: `(60, 'SSL certificate problem, verify that the CA cert is OK. Details: error:14090086:SSL routines:SSL3_GET_SERVER_CERTIFICATE:certificate verify failed')`
- No verifica URL: `(51, "SSL: certificate subject name 'wsaahomo.afip.gov.ar' does not match target host name 'wsaahomo.afip.gob.ar'")`
- El archivo de autoridades certificantes (parámetro `cacert`) es incorrecto: `SSLError: [Errno 185090050] _ssl.c:344: error:0B084002:x509 certificate routines:X509_load_cert_crl_file:system lib,error: (77, ''), SSLHandshakeError : [ SSL: CERTIFICATE_VERIFY_FAILED ] certifícate verify failed (_ssl.c:581 )`, o `WindowsError: [Error 2] El sistema no puede encontrar el archivo especificado` / `IOError: [Errno 2] No such file or directory` (usar alternativamente `conf/afip_ca_info.crt`)

Se utiliza una librería de transporte distinta (pycurl) y posiblemente sea reemplazada en el futuro, por lo que recomendamos dejar parametrizado este tema para evitar inconvenientes (pudiendo usar cualquier librería http y/o deshabilitar la comprobación de autoridad certificante al no pasarle los nuevos parámetros a Conectar, para mantener compatibilidad con versiones anteriores y/o en caso de que AFIP instale un certificado incompatible en sus servidores).

Tabla comparativa Webservices

A continuación se presenta un resumen de webservices, herramientas, atributos y métodos soportados:

Característica	**WSFE v0**	**WSFE v1, v1.1, v2, v2.6**	**WSBFE v0 y v1**	**WSFEX v0 y**
Normativa AFIP	RG2485	RG2485, RG2926, RG2959, RG2975, RG3067, RG3571, RG3668, RG3749, RG3779	RG2557	RG2758, RG3 RG3689
Régimen	Mercado Interno original	Mercado interno (sin informar detalle)	Bono Fiscal - Bienes de Capital	Exportación
Clases de Facturas	A, B, *C*, *M*	A, B, C, M	A, B	E
Sujetos comprendidos	R.I. (**obsoleto desde 1/7/2011**)	Responsables Inscriptos, Exentos y Monotributo: **Régimen General**	Responsables Inscriptos inscriptos en la *Secretaría de Comercio*	Exportadores i en los *Registro Especiales Ad* (**V1 obligator 31/12/2011**)
Objeto COM (Windows)	WSFE	WSFEv1	WSBFE o WSBFEv1	WSFEX o WS
Biblioteca COM (Windows)	wsfe.dll	wsfev1.dll	pyafipws.dll o wsbfev1.dll	pyafipws.dll o wsfexv1.dll
Modulo Python (Linux)	wsfe.py	wsfev1.py	wsbfe.py o wsbfev1.py	wsfex.py o ws
Herramienta símil SIAP/RECE (Windows/Linux)	rece.exe / rece.py	rece1.exe / rece1.py	receb.exe / receb.py	recex.exe / rec
Información General	PyAfipWs	Proyecto Version 1	Bonos Fiscales Electrónicos	Factura Electr Exportación
	WSFE	WSFEv1	WSBFE	WSFEX

Documentación Técnica (Manual)				
Nombre de servicio para `WSAA.CreateTRA`	"wsfe"	"wsfe"	"wsbfe"	"wsfex"
Método Solicitud de Autorización CAE o información de CAEA	`Aut` (CAE)	`CAESolicitar` / `CAEARegInformativo`	`Authorize` (CAE)	`Authorize`
Métodos para CAEA (anticipado)		`CAEASolicitar`, `CAEAConsultar`		
Campo ID (identificador único)	en `Aut`		en `Authorize`	en `Authoriz`
Reproceso	sí (realizado por AFIP)	automático (emulado en la interfaz)	sí (realizado por AFIP)	sí (realizado p
Campo productos/servicios	presta_serv=S/N	concepto=1/2/3		tipo_expo=1/2
Campo moneda/cotización		moneda_id, moneda_ctz	imp_moneda_id, imp_moneda_ctz	moneda_id, m
Método `CrearFactura` o `Aut`	id, presta_serv, tipo_doc, nro_doc, tipo_cbte, punto_vta, cbt_desde, cbt_hasta, imp_total, imp_tot_conc, imp_neto, impto_liq, impto_liq_rni, imp_op_ex, fecha_cbte, fecha_venc_pago, fecha_serv_desde, fecha_serv_hasta	concepto, tipo_doc, nro_doc, tipo_cbte, punto_vta, cbt_desde, cbt_hasta, imp_total, imp_tot_conc, imp_neto, imp_iva, imp_trib, imp_op_ex, fecha_cbte, fecha_venc_pago, fecha_serv_desde, fecha_serv_hasta, moneda_id, moneda_ctz	tipo_doc, nro_doc, zona, tipo_cbte, punto_vta, cbte_nro, fecha_cbte, imp_total, imp_neto, impto_liq, imp_tot_conc, impto_liq_rni, imp_op_ex, imp_perc, imp_iibb, imp_perc_mun, imp_internos, imp_moneda_id, Imp_moneda_ctz	tipo_cbte, pun cbte_nro, fech imp_total, tipo permiso_existe dst_cmp, clien cuit_pais_clie domicilio_clie id_impositivo, moneda_id, m obs_comercial forma_pago, i idioma_cbte, incoterms_ds, fecha_pago)
Método `AgregarItem` (detalle de artículo)			ncm, sec, ds, qty, umed, precio, bonif, iva_id, imp_total	codigo, ds, qty precio, imp_to
Método `AgregarCbteAsoc` (comprobantes asociados)		tipo_cbte, pto_vta, nro_cbte, CUIT, fecha		tipo_cbte, pto_ nro_cbte, cuit_
Método `AgregarPermiso` (permisos de embarque)				permiso_id, pa
Método `AgregarIVA` (subtotales de alícuota		iva_id, base_imp, importe		

de iva)				
Método `AgregarTributo` (otros impuestos)		tributo_id, ds, base_imp, alic, importe		
Método para consultar puntos de venta habilitados		`ParamGetPtosVenta`		
Método para recuperar un comprobante		`CompConsultar`	`GetCMP`	`GetCMP`
Método para obtener útlimo nº comprobante	`RecuperaLastCMP`	`CompUltimoAutorizado`	`GetLastCMP`	`GetLastCMP`
Métodos para obtener tablas de parámetros		`ParamGetTiposCbte`, `ParamGetTiposConcepto`, `ParamGetTiposDoc`, `ParamGetTiposIva`, `ParamGetTiposMonedas`, `ParamGetTiposOpcional`, `ParamGetTiposTributos`, `ParamGetCotizacion`, `ParamGetPtosVenta`	`GetParamMon`, `GetParamTipoCbte`, `GetParamUMed`, `GetParamTipoIVA`, `GetParamNCM`	`GetParamMo` `GetParamTi` `GetParamTi` `GetParamIc` `GetParamUM` `GetParamIr` `GetParamDs` `GetParamDs` `GetParamCt`

S.E.U.O.

Notas Generales

- Convención de nombres: si bien por transparencia se trató de utilizar los nombres de los métodos definidos en las especificaciones técnicas, se han cambiado los nombres de parámetros y métodos auxiliares para simplificar y unificar las interfaces, manteniendo cierta coherencia entre los distintos webservices, por lo que algunas denominaciones internas pueden variar respecto a las especificaciones técnicas de AFIP.
- Esta tabla contiene un resumen de las consideraciones generales, revisar la normativa vigente, especificaciones y documentación técnica actualizada para contemplar los casos particulares. Además, se respetan las descripciones de servicios (WSDL), por ej. si bien AFIP publica la especificación técnica "RG 2485 - Manual para el desarrollador V.2.5", el webservice sigue siendo WSFEv1; ver RG 2485 - WSFEV1 - Manual para el desarrollador V.2.6, RG 2904 - WSMTXCA - Manual para el Desarrollador - V.0.2, RG 2758 - wsfex - Manual del desarrollador - V.0.11, WSBFEV1 ? Bonos Fiscales Electrónicos Manual para el desarrollador V.1.1
- Todas las facturas generadas en PDF o formatos similares pueden incluir el detalle de las operaciones (artículos, códigos, descripción, precio unitario, etc.), independientemente de lo que obligue a informar AFIP en cada webservice. No se podría optar por utilizar el WSMTXCA ("con detalle", RG2904) si no ha recibido notificación por juez administrativo de AFIP, en general corresponde el WSFEv1 ("sin detalle", RG2485 et.al.) para la mayoría de los contribuyentes. Para emitir facturas con CAEA (código de autorización electrónico anticipado), se deben cumplir ciertas condiciones (cantidad mínima de facturas por mes), en general solo aplicable a Auto-Impresores y Grandes Contribuyentes.
- La numeración de los comprobantes comienza de 1 por cada punto de venta y tipo de comprobante ("no se comparten talonarios" como con la facturación tradicional). Incluso, cada webservice generalmente requiere dar de alta un punto de venta exclusivo. No se puede continuar la numeración

previa ni de otro webservice.
- La fecha de los comprobantes no puede ser anterior al último emitido, pero generalmente se puede autorizar facturas con fecha entre 5 a 10 días en el pasado o en el futuro por contingencias (dependiendo del webservice y si es servicios o productos). Ver método de consultas y recupero de comprobantes. No se podrá emitir comprobantes antes de la fecha indicada en el formulario de Empadronamiento (por ej., si se eligió el inicio en Julio/2015, no se pueden emitir facturas con fecha de junio).
- En caso de inoperatividad del servidor de AFIP, se debería emitir facturas tradicionales y presentar un formulario multinota en la agencia, indicando las facturas realizadas manualmente y dejando constancia del motivo. Para más información ver RG2485 Art 33

Se debe confirmar los temas de normativas/legislación con un Contador Público Nacional debidamente habilitado y con experiencia en el tema.

Disclaimer - Aviso Legal

Toda información es proporcionada a Titulo Informativo. El programa es software libre liberado bajo licencia GPLv3 y se entrega como está, sin garantías explícitas ni implicitas de ningún tipo, incluyendo sin limitación, pérdida de ganancias, interrupción de negocios, pérdida de programas u otros datos en sistemas de computación o cualquier otro reclamo. Al usarlo acepta hacerlo bajo su propia responsabilidad, conociendo la normativa y reglamentaciones existentes.

Instalación

Descargas

Bajar y Ejecutar el instalador:

Importante: ver Actualizaciones para novedades importantes, instaladores y cambios recientes.

Recomendado: Instalador unificado (conteniendo la interfaz para todos los servicios web para homologación -solo evaluación/pruebas-):

Windows 32 / 64 bits:

- PyAfipWs-2.7.2912-32bit-homo.exe (GitHub): Instalador de desarrollo unificado de la interfaz (para todos los webservices de factura electrónica, incluye WSAA, WSFEv1, WSMTXCA, WSFEXv1,WSBFEv1, WSCT, WSFECred, WSCDC, WSCOC, PyFEPDF, PYI25 y PYEMAIL, WS_SR_Padron, COT/ IIBB Arba, Granos -WSLPG y WSCTG-, Pecuarios- WSLSP, WSLUM, WSLTV, Trazabilidad Medicamentos, Trazabilidad de Productos Médicos, Trazabilidad de Precursores Quínicos, Trazabilidad de Productos Agroquímicos Fitosanitarios, Trazabilidad de Productos Veterinarios). **Recomendado para desarrollo/evaluación**
- PyAfipWs-2.7.1949-32bit+wsaa_2.11c+wsfev1_1.19a-typelib.exe: Instalador de desarrollo para referenciar vía librería de tipos- (TypeLib). Sólo incluye WSAA y WSFEv1.

Consultar por otros webservices y sistemas operativos.

IMPORTANTE: Los instaladores publicados son solo para evaluación, por lo que pueden no estar actualizados con los últimos ajustes en los servidores y/o ejemplos. Ver instructivo Instalación Codigo Fuente

(video explicativo) o consultar por soporte comercial.

Los ejemplos completos del uso de la interfase en Visual Basic, Visual Foxpro y otros lenguajes se puede descargar desde:

- GitHub (repositorio actualizado) o Google Code (histórico)

Para guias iniciales, ver base de datos con ejemplos completos de WSFEv1 y WSFEX (implementación de referencia):

- factura_electronica.mdb MS Access 2000 o sup., programado en VBA
- pyafipws.mdb: MS Access 97, programado en VBA ***(nuevo con FCE)***
- pyafipws2k.mdb: MS Access 2000 o sup., programado en VBA (versión anterior)

IMPORTANTE : las últimas versiones incorporan nuevas funcionalidades y deben ser testeadas con su sistema antes de ser instaladas en producción. De encontrar problemas de instalación (por causas del sistema operativo o similar) que impidan crear el objeto o conectarse al webservice, se recomienda utilizar las versiones anteriores o registrar manualmente los componentes de forma autónoma con `--register`. Ver Instalación Manual para más detalles. Recordamos que este tipo de incidentes pueden ser consultados con el soporte comercial o soporte comunitario, ver Más Información

Incidencias Conocidas: Dado que existen varias versiones dependiendo de que sistema operativo se utilice, en general no debería haber problemas de instalación, pero puede haber cuestiones de permisos de acceso / políticas de seguridad en algunos entornos Windows (por ej. *Permission Denied*). En esos casos recomendamos instalarlo como Administrador o darle permisos de modificación (Propiedades, Solapa de Seguridad) al usuario del sistema operativo en la carpeta del componente (generalmente `C:\Archivos de programa\PyAfipWs` o `Program Files`). Ver Permisos en Windows para mayor información.

Se recomienda desinstalar cualquier versión anterior, y borrar la carpeta de instalación para evitar archivos temporales residentes o temas similares.

Instalación Manual Avanzada

Para la interfaz COM, cada webservice tiene su DLL o EXE que puede ser instalado manualmente (no recomendado en general ya que no verifica si el archivo esta en uso, no desinstala automáticamente la versión anterior, entre otros temas que pudieran surgir):

- Instalación manual: `pyafipws.exe --register` o utlizar `regsvr32 pyafipws.dll`
- Desinstalación manual: `pyafipws.exe --unregister` o `utlizar regsvr32 /u pyafipws.dll`

El proceso es similar para instalar la interfaz a otros webservices (cambiando `pyafipws` por `wsfev1`, `wsfexv1`, etc. según corresponda, por ej. `wsfexv1.exe --register`) También se puede registrar directamente con python, `python wsaa.py --register`

Ambos comandos de instalación son similares, pero instalan la interfaz de manera distinta:

- Embebida (InProcServer): registrando la biblioteca DLL con regsvr32, cuando se invoque a la interfaz, se carga en el mismo proceso del programa principal.
- Independiente (LocalServer): ejecutando el programa EXE con --register, cuando se invoque a la

interfaz, se crea un nuevo proceso externo al programa principal (se verá el ejecutable por ej. en el administrador de tareas)

NOTA: solo es necesario registrar el EXE o la DLL (no ambas). En general es conveniente registrar las DLL para que se cargue como una librería compartida embebida (InProcServer) y evitar temas de permisos/firewall, ya que el EXE registra un proceso independiente (LocalServer) con sus implicancias (pueden surgir temas de permisos/firewall en ciertos sistemas operativos, pero se ejecuta independientemente, eliminando inconvenientes de dependencias y errores de carga).

Instalación Silenciosa (Desatendida)

Si bien el instalador muestra un asistente para la instalación (selección del idioma, licencia, y directorio, progreso), es posible instalarlo sin intervención del usuario, en segundo plano sin mostrar pantalla alguna, con:

```
instalador-PyAfipWs-1.25c-homo.exe /S /D=C:\PyAfipWS
```

Las opciones son:

- `/S` indica instalación silenciosa
- `/D` indica el directorio de instalación

Guía de Instalación

La interfase se instala automáticamente en `C:\Archivos de Programa`, en el directorio elegido en el segundo paso de instalación, como se describe a continuación:

Firma Digital (Editor Comprobado)

Como paso previo de seguridad, al ejecutar el instalador, verificar que el mismo haya sido firmado por nosotros, debiendo aparecer:

- Editor Comprobado: Sistemas Agiles (Mariano Reingart)

Consultar en caso de no comprobarse (o mencionar "Editor Desconocido"), ya que no sería un instalador compilado originalmente por nosotros sin alteraciones.

Paso 1: Seleccionar idioma

Idiomas disponibles: I

- Spanish: español (predeterminado)
- English: inglés

Paso 2: Licencia

Para utilizar el software, debe leer y aceptar la licencia:

Paso 3: Direcorio de instalación

De manera predeterminada, la interfaz se instalará en una carpeta bajo el directorio estándard de windows (por ej. `C:\Archivos de Programa` o `C:\Program Files\`). Se recomienda instalarla directamente en el directorio raiz (por ej,`C:\PYAFIPWS`) para simplificar el acceso a las herramientas por linea de comandos:

Paso 3: Finalizar instalación

Por último, el instalador realizará la instalación:

- Desinstalar versiones anteriores
- Crear el directorio de destino
- Extraer y copiar los archivos
- Crear desinstalador
- Copiar configuración (RECE.INI, si corresponde)
- Registrar DLL

Puede ver el avance y estado final presionando el botón *Ver Detalles*:

Referenciar TypeLib

Ciertos instaladores (-typelib) incluyen una librería de tipos que es posible referenciar, por ej. en VB:

Importante: En general no es necesario referenciar el componente. Invocando `CreateObject` debería funcionar y devolvería el objeto dinámico en la mayoría de los lenguajes de programación (VB, VFP, etc.)

Más información

- Soporte Comercial (asesoramiento inicial sin cargo de cortesía -limitado y sin compromisos-):
  - Web: http://www.sistemasagiles.com.ar/trac/wiki/PyAfipWs
  - Por email: facturaelectronica@sistemasagiles.com.ar
  - Telefónicamente: (011) 15-3048-921
- Soporte comunitario (gratuito):
  - Sitio del proyecto: http://www.pyafipws.com.ar/
  - Grupo de noticias/discusiones: http://groups.google.com/group/pyafipws
  - Código Fuente: https://github.com/reingart/pyafipws (actual) y http://code.google.com/p/pyafipws/ (histórico)
  - Incidencias: https://github.com/reingart/pyafipws/issues

Servicio Web de Autenticación y Autorización (WSAA)

Este servicio es necesario para acceder al resto de los servicios de negocios (ej. Factura Electrónica) ofrecidos por la AFIP.

Se debe generar un Ticket de Requerimiento de Acceso (TRA), firmarlo criptográficamente (generando un mensaje CMS) que es envía al servicio web para obtener un Ticket de Acceso (TA).

Esta interfase crea el mensaje en XML, lo firma criptográficamente y lo codifica en base-64 como lo requiere la AFIP, por lo que en general no es necesario realizar ninguno de estos pasos.

Para más información oficial ver especificaciones técnicas del WSAA y Arquitectura WS en el sitio de la AFIP.

Importante: es requerido tramitar los certificados para poder operar este webservice. Para más información ver: Instructivos sobre Certificados

Objeto

- El objeto COM se crea invocando a CreateObject("WSAA")

Métodos

- **`CreateTRA(service, ttl)`**: recibe el nombre del servicio (wsfe, wsbfe, wsfex, wsctg, etc.) y el tiempo de vida (en segundos, por defecto 2400) y devuelve un ticket de requerimiento de acceso (string con formato xml). Se puede revisar o modificar el xml devuelto (en general no es necesario)
- **`SignTRA(tra, certificado, clave_ privada, passphrase)`**: recibe el ticket de requerimiento de acceso (string con formato xml, creado previamente) y las ubicaciones del certificado y clave privada (o el contenido de los mismos). Firma el ticket y devuelve el mensaje CMS (string con formato xml). **Importante**: El parámetro opcional `passphrase` permite protejer la clave privada por contraseña (*Disponible a partir de la versión 2.08*). También se puede pasar el contenido de los archivos (como string) en `certificado` y `clave_ privada` (pudiendo ser guardados en la base de datos u otro tipo de almacenamiento).
- **`CallWSAA(cms, url_webservice)`**: Recibe el mensaje CMS (string con formato xml) y la URL del webservice. Se conecta con el servidor remoto, ejecuta el método remoto LoginCMS y devuelve el Ticket de Acceso (TA, string con formato xml) generado por la AFIP. Si no se especifíca url, se utiliza servidores de homologación.
- **`Conectar(cache, url_webservice_wsdl, proxy, wrapper, cacert)`** ***Nuevo!*** : Establece la conexión con el servidor remoto, recibe el directorio de archivos temporales, la URL del WSDL (descripción del webservice) y proxy en formato 'usuario:clave@servidor:puerto'. Si no se especifíca url, se utiliza servidores de homologación. Parametros adicionales optativos: wrapper es la librería HTTP a utilizar y cacert la ruta al certificado de la autoridad de certificante del servidor (CA)
- **`LoginCMS(cms)`** ***Nuevo!*** : Recibe el mensaje CMS (string con formato xml) ejecuta el método remoto LoginCMS y devuelve el Ticket de Acceso (TA, string con formato xml) generado por la AFIP. Debe conectarse previamente con Conectar. Establece los atributos Token, Sign.
- **`AnalizarXml(xml)`** ***Nuevo!*** : analizar el mensaje xml para poder extraer atributos puntuales (xml puede ser "XmlRequest", "XmlResponse" o cualquier texto xml válido, por defecto el ticket de acceso)
- **`ObtenerTagXml(tag1, tag2, ...)`** ***Nuevo!*** : busca en el mensaje xml analizado la etiqueta tag1, luego tag2 y así sucesivamente, devolviendo el contenido (texto) del dato si fue encontrada, o nulo en caso contrario.
- **`Expirado(fecha)`** ***Nuevo!*** : recibe una fecha (por defecto la fecha de expiración del último ticket de acceso generado), y verifica contra el reloj del equipo si dicha fecha ha expirado (en cuyo caso devuelve verdadero)
- **`Autenticar(service, crt, key, [wsdl, proxy, wrapper, cacert, cache, debug])`** ***Nuevo!*** : Método unificado para obtener el ticket de acceso. Almacena temporalmente el ticket de acceso en un archivo en el directorio cache (debe tener permisos de escritura en ese directorio), y automáticamente lo reutiliza o solicita uno nuevo en caso de que no haya vencido.

Disponible a partir de la versión 2.07

Se recomienda comenzar a utilizar Conectar y LoginCMS en reemplazo de CallWSAA, ya que los nuevos métodos tienen mejor manejo de errores, pueden reutilizar la conexión, etc.

Métodos para Certificados

Los métodos adicionales para el manejo de certificado permiten automatizar el procesamiento y generar pedidos de certificado directamente con este componente, sin necesidad de utilizar comandos externos.
Nuevo! **Disponible a partir de la versión 2.10**

- **`AnalizarCertificado(crt)`**: "Carga un certificado digital y extrae los campos más importantes (establece propiedades `Identidad`, `Caducidad`, `Emisor`).
- **`CrearClavePrivada(self, filename="privada.key", key_length=2048, pub_exponent=0x10001, passphrase="")`**: Crea una clave privada (private key), necesaria para generar un certificado. Queda referenciada internamente para poder crear el CSR (ver siguiente método), por lo que las dos operaciones deben hacerse en la misma instancia. Puede utilizarse una contraseña (`passphrase`) para mayor seguridad (especialmente si la clave privada va a alojarse en una carpeta compartida).
- **`CrearPedidoCertificado(self, cuit="", empresa="", nombre="pyafipws",filename="empresa.csr")`**: Crear un certificate signing request (X509 CSR) - pedido de certificado a gestionar por Clave Fiscal de AFIP para obtener el crt. Se debe indicar el CUIT del representante, nombre de la empresa (tal cual aparece en la constancia de inscripción), nombre de "fantasia" / alias del certificado (identificando punto de venta, sistema de facturación, etc.), y nombre del archivo a generar (recomendable ruta absoluta a una carpeta con permisos de escritura).

```
wsaa.AnalizarCertificado("reingart.crt")
print wsaa.Identidad
print wsaa.Caducidad, "(Fecha de vencimiento)"
print wsaa.Emisor
print wsaa.CertX509
```

Por linea de comando puede ejecutarse el módulo `wsaa.py` con el método `--analizar`:

```
WSAA_cli.exe --analizar reingart.crt reingart.key
```

Atributos

- **`Token`**: el el código de autorización generado por la AFIP, necesario para utilizar el resto de los servicios web.
- **`Sign`**: el la firma de autorización generado por la AFIP, necesario para utilizar el resto de los servicios web.
- **`Version`**: versión de la interfase (ej. ?1.6?)
- **`InstallDir`** ***Nuevo!*** : directorio de instalación (ej. `C:\Archivos de Programa\WSAA`)
- **`Identidad`**, **`Caducidad`**, **`Emisor`**, **`CertX509`**, ***Nuevo!*** : campos del certificado analizado (*actualización 2.10 o posterior*)
- **`XmlRequest`**, **`XmlResponse`**: requerimiento y respuesta XML (para depuración)
- **`Excepcion`**, **`Traceback`**: mensaje de error y traza de rastreo (para depuración)
- **`LanzarExcepciones`**: establece si se deben emitir errores al lenguaje de programación (habilitado por defecto), o serán controlados por el programa (revisando el atributo Excepcion luego de cada

método)

Ejemplo

```
Dim WSAA As Object

' Crear objeto interface Web Service Autenticación y Autorización
Set WSAA = CreateObject("WSAA")

' Generar un Ticket de Requerimiento de Acceso (TRA)
service = "wsfe" ' servicio a acceder
ttl = 2400       ' tiempo de vida en segundos
tra = WSAA.CreateTRA(service, ttl)
' Especificar la ubicacion de los archivos certificado y claves
Path = CurDir() + "\"
Certificado = "reingart.crt" ' certificado es el firmado por la afip
ClavePrivada = "reingart.key" ' clave privada usada para crear el cert.

' Generar el mensaje firmado (CMS)
cms = WSAA.SignTRA(tra, Path + Certificado, Path + ClavePrivada)

WSAA.LanzarExcepciones = False  ' revisar Excepcion para controlar errores

' Llamar al web service para autenticar:
' anteriormente:
'    ta = WSAA.CallWSAA(cms, "https://wsaa.afip.gov.ar/ws/services/LoginCms") ' Hologación
' ahora:
cache = "" ' Directorio para archivos temporales (dejar en blanco para usar predeterminado)
wsdl = "https://wsaahomo.afip.gov.ar/ws/services/LoginCms?wsdl" ' homologación
proxy = "" ' usar "usuario:clave@servidor:puerto"
ok = WSAA.Conectar(cache, wsdl, proxy)
ta = WSAA.LoginCMS(cms) ' otener ticket de acceso

If WSAA.Excepcion <> "" Then
    MsgBox WSAA.Excepcion, vbCritical, "Excepción"
End If

'Obtener Token y Sign de autorización
Token = WSAA.Token
Sign  = WSAA.Sign
```

El Ticket de Acceso contiene el Token (código) y Sign (firma) son los elementos que se utilizan en los otros servicios web para lograr acceso a los mismos.

Estos datos se pueden usar por un período de tiempo (40 minutos, modificable, ver Reutilización de Ticket de Acceso), siendo necesario volver a invocar el WSAA solo al finalizar este período, para obtener nuevos Token y Sign.

Errores

El WSAA informa errores alfanuméricos (Err.Number ? vbObjectError = 0), debiendo verificar la descripción (Err.Description). Entre ellos los más comunes son:

- cms.cert.expired, cms.cert.invalid, cms.cert.untrusted: El certificado no fue firmado por la AFIP, expiró, es inválido, no se confía en su emisor, o no coincide la fuente. Se debe revisar o emitir correctamente los certificados. Ver trámite de Generación

- xml.generationTime.invalid, xml.expirationTime.expired, xml.expirationTime.invalid: El tiempo de generación/expiración es inválido. Debe sincronizar relojes con el servidor de la AFIP. En Windows: propiedades del reloj, habilitar Hora de Internet, ingresar time.afip.gov.ar
- wsn.unavailable, wsaa.unavailable: servicio no disponible momentáneamente
- coe.notAuthorized: Computador no autorizado a acceder los servicios de AFIP. Ver trámite de Asociación
- cms.cert.untrusted: Certificado no emitido por AC de confianza. Revisar URL para ambiente correcto (producción/homologación) y Comprobación de Autenticación
- wsn.notFound: Servicio informado inexistente: se debe crear el ticket para un servicio válido (wsfe, wsfex, etc.)
- coe.alreadyAuthenticated: El CEE ya posee un TA valido para el acceso al WSN solicitado (ver reutilización del ticket de acceso abajo)

Para más errores frecuentes ver Falla SOAP WSAA

Reutilización de Ticket de Acceso

Ante problemas de "timeout" y "java socket read failed" al intentar solicitar ticket de acceso (WSAA) o "ns1:coe.alreadyAuthenticated: El CEE ya posee un TA valido para el acceso al WSN solicitado" (en general por sobrecarga o limitación de AFIP), es posible mitigar el tema reusando un ticket de acceso (token y sign) por hasta 12 hs (40 minutos por defecto), estableciendo el tiempo de vida del ticket pasandolo en el segundo parámetro de `CreateTRA` (`ttl`, en segundos, en este manual esta documentado en secciones previas)

Por ej para un tiempo de vida de 10 hs (10*3600 segundos, valor recomendado para evitar problemas de sincronización de relojes):

```
tra = WSAA.CreateTRA("wsfe", 36000)
```

Luego de solicitar el ticket de acceso, el Token y Sign puede ser usado por el período especificado (dichos valores pueden almacenarse en memoria, en un archivo o base de datos, etc.).

Importante': se puede guardar `WSAA.Token` y `WSAA.Sign` en una variable global, y luego reutilizarlo cada vez que se genera una factura.

```
' luego de llamara WSAA.LoginCMS o WSAA.CallWSAA o WSAA.Autenticar

token = WSAA.Token
sign = WSAA.Sign

' guardar token y sign en la base de datos, archivo o una variable global

' antes de llamar a WSFEv1.CAESolicitar, WSFEXv1.Authorize o similar:

WSFEv1.Token = token
WSFEv1.Sign = sign
```

En general no es necesario verificar la fecha del ticket de acceso, ya que en general es válido por 12 horas.

Igualmente, desde las versiones actualizadas de WSAA (2.02 o superior), es posible llamar al método `WSAA.Expirado()` antes de utilizar el ticket de acceso, y este devolverá Verdadero si el ticket de acceso sigue siendo válido (dada la fecha del reloj del equipo), o Falso si ha vencido su plazo y debe solicitarse un nuevo ticket de acceso.

Por ejemplo, luego de obtener el Token y Sign (ya sea por CallWSAA o LoginCMS), obtener y guardar la fecha de expiración:

```
expiracion = WSAA.ObtenerTagXml("expirationTime")
```

Luego antes de usar el Token y Sign (por ej, antes de llamar a WSFEv1, WSFEX, etc.), preguntar si el TA es válido (no ha expirado):

```
expiro = WSAA.Expirado(expiracion)
```

Si expiro es True, se debe generar otro ticket de acceso porque ha vencido su validez (con este método se puede guardar y verificar independientemente la fecha de expiración por cada servicio a usar: wsfe, wsfex, wsbfe, wsmtxca, etc).

Si se comparte el certificado entre distintas máquinas o programas, debe compartirse el ticket de acceso (por ej., guardandolo en un archivo o base de datos) y realizar esta verificación en cada instancia.

Ejemplo completo en:

- reusar_ticket_acceso.bas (VB) y reusar_ticket_acceso_avanzado.prg (VFP) más avanzado guardando el TA en un archivo (y poder compartirlo por red o entre varios programas)
- reusar_ticket_acceso.prg (VFP) y reusar_ticket_acceso_simple.bas (VB) simplificado, usando el TA en memoria

Nota: No es necesario guardar el ticket de acceso completo en un archivo para reutilizarlo. Este ejemplo es útil si se debe compartir el ticket entre diferentes programas u equipos en una red. Si se usa en un solo programa y este permanece abierto, se puede mantener el valor del `Token` y `Sign` directamente en memoria (por ej. en una variable) y reusarlo por hasta 12 hs (generalmente, solicitandolo al iniciar el programa y manteniendolo por una jornada de trabajo completa). También se podría almacenar el valor de `Token` y `Sign` en una base de datos o similar.

Ejemplo simple reutilización de TA

Ejemplo completo de reuso (usando el ticket de acceso en memoria):

```
Function Autenticar(TA as String = "") as String
    ' Procedimiento para autenticar con AFIP y reutilizar el ticket de acceso
    ' Llamar antes de utilizar WSAA.Token y WSAA.Sign (WSAA debe estar definido a nivel de módulo)
    Dim ok, expiracion, solicitar, token, sign

    ' cargar ticket de acceso previo (si no se mantiene WSAA instanciado en memoria)
    If TA <> "" Then
        ok = WSAA.AnalizarXml(TA)
    End If

    ' revisar si el ticket es válido y no ha expirado:
    expiracion = WSAA.ObtenerTagXml("expirationTime")
    Debug.Print "Fecha Expiracion ticket: ", expiracion
    If IsNull(expiracion) Then
        solicitar = True                            ' solicitud inicial
    Else
        solicitar = WSAA.Expirado(expiracion)       ' chequear solicitud previa
    End If
```

```
    If solicitar Then
        ' Generar un Ticket de Requerimiento de Acceso (TRA)
        tra = WSAA.CreateTRA()

        ' uso la ruta a la carpeta de instalaciòn con los certificados de prueba
        ruta = WSAA.InstallDir + "\"
        Debug.Print "ruta", ruta

        ' Generar el mensaje firmado (CMS)
        cms = WSAA.SignTRA(tra, ruta + "reingart.crt", ruta + "reingart.key") ' Cert. Demo

        ok = WSAA.Conectar("", "https://wsaahomo.afip.gov.ar/ws/services/LoginCms") ' Homologacion

        ' Llamar al web service para autenticar
        TA = WSAA.LoginCMS(cms)
    Else
        Debug.Print "no expirado!", "Reutilizando!"
    End If
    Debug.Print WSAA.ObtenerTagXml("destination")

    ' Obtener las credenciales del ticket de acceso (desde el XML por si no se conserva el objeto WSAA)
    token = WSAA.ObtenerTagXml("token")
    sign = WSAA.ObtenerTagXml("sign")
    ' Al retornar se puede utilizar token y sign para WSFEv1 o similar
    ' Devuelvo el ticket de acceso (RETURN) para que el programa principal lo almacene si es necesario:
    Autenticar = TA
End Function
```

Importante: en este caso, WSAA debe ser un objeto global y persistente (al menos por lo que dure el programa en producción), no se debe instanciar por cada llamada a Autorizar; de lo contrario, se perderán el Token y Sign, no reutilizando el ticket de acceso. Si esto no es posible, se debe persistir el Ticket de Acceso en un archivo o base de datos compartida.

Ejemplo avanzado reutilización de TA

Ejemplo completo de reuso (almacenando el ticket de acceso en un archivo, también podría usarse una tabla en base de datos, sesión u otro medio durable):

```
' inicializo las variables:
Token = ""
Sign = ""

' busco un ticket de acceso previamente almacenado:
If Dir("ta.xml") <> "" Then
    ' leo el xml almacenado del archivo
    Open "ta.xml" For Input As #1
    Line Input #1, ta_xml
    Close #1
    ' analizo el ticket de acceso previo:
    ok = WSAA.AnalizarXml(ta_xml)
    If Not WSAA.Expirado() Then
        ' puedo reusar el ticket de acceso:
        Token = WSAA.ObtenerTagXml("token")
        Sign = WSAA.ObtenerTagXml("sign")
    End If
End If

' Si no reuso un ticket de acceso, solicito uno nuevo:
If Token = "" Or Sign = "" Then
```

```
    ' Generar un Ticket de Requerimiento de Acceso (TRA)
    tra = WSAA.CreateTRA("wsfe", 43200) ' 3600*12hs
    ' Especificar la ubicacion de los archivos certificado y clave privada
    cert = "reingart.crt" ' certificado de prueba
    clave = "reingart.key" ' clave privada de prueba
    ' Generar el mensaje firmado (CMS)
    cms = WSAA.SignTRA(tra, cert, clave)
    If cms <> "" Then
        ' Llamar al web service para autenticar (cambiar URL para produccion):
        wsdl = "https://wsaahomo.afip.gov.ar/ws/services/LoginCms?wsdl"
        ok = WSAA.Conectar("", wsdl)
        ta_xml = WSAA.LoginCMS(cms)
        If ta_xml <> "" Then
            ' guardo el ticket de acceso en el archivo
            Open "ta.xml" For Output As #1
            Print #1, ta_xml
            Close #1
        End If
        Token = WSAA.Token
        Sign = WSAA.Sign
    End If
    ' reviso que no haya errores:
    Debug.Print "Excepcion:", WSAA.Excepcion
    If WSAA.Excepcion <> "" Then
        Debug.Print WSAA.Traceback
    End If
End If

' Imprimir los datos del ticket de acceso: ToKen y Sign de autorización
Debug.Print "Token: " + Token
Debug.Print "Sign: " + Sign
```

Ejemplo del método Autenticar para reutilización de TA

Importante: A partir de la actualización 2.07, es posible utilizar el método unificado `Autenticar` (que automáticamente contempla el tema de reutilización al guardar el Ticket de Acceso en un archivo temporal en el directorio cache), en conjunción con `SetTicketAcceso`:

```
# Crear la instancia (usar CreateObject en VB y similares)
wsaa = WSAA()

# servidor de AFIP para homologación (cambiar para producción)
wsaa_url = "https://wsaahomo.afip.gov.ar/ws/services/LoginCms?wsdl"

# parametros opcionales:
proxy = ""                          # solo usar si hay servidor intermedio
wrapper = ""                        # httplib2 (default), pycurl (depende proxy)
cacert = "conf/afip_ca_info.crt"    # autoridades certificantes (servidores)
cache = ""                          # directorio archivos temporales (verificar permisos)
debug = False                       # depuración interna (en VB es palabra reservada, usar otro nombre)

# obtener el TA para pruebas
ta = wsaa.Autenticar("wsfe", "reingart.crt", "reingart.key", wsaa_url, proxy, wrapper, cacert, cache, debug)

# utilizar las credenciales:
print wsaa.Token
print wsaa.Sign

# establecer Ticket de Acceso en un solo paso (Nuevo método):
wsfev1.SetTicketAcceso(ta)
```

Nota: dependiendo del sistema operativo, deberá conceder permisos de acceso de escritura en la carpeta cache para poder almacenar el ticket de acceso. También es posible pasar el 8vo parámetro cache de Autenticar con otro directorio para los archivos temporales. Ver Permisos en Windows para más información.

Obtención de Atributos Avanzados WSAA

En versiones recientes (2.04a o superior), si no hubo excepción es posible revisar y obtener datos avanzados del ticket de acceso (útiles para depuración y solución de errores):

- Origen (Source): `WSAA.ObtenerTagXml("source")`
- Destino (Destination): `WSAA.ObtenerTagXml("destination")`
- ID Único: `WSAA.ObtenerTagXml("uniqueId")`
- Fecha de Generación: `WSAA.ObtenerTagXml("generationTime")`
- Fecha de Expiración: `WSAA.ObtenerTagXml("expirationTime")`

Si ha ocurrido error (llamar previamente a `WSAA.AnalizarXml("XmlResponse")` para analizar la respuesta):

- Codigo de Fallo: `WSAA.ObtenerTagXml("faultcode")`
- Mensaje de Fallo: `WSAA.ObtenerTagXml("faultstring")`
- Servidor: `WSAA.ObtenerTagXml("ns3:hostname")`

Estos datos son útiles para determinar problemas con el certificado, ver Comprobación de Autenticación WSAA

Servicio Web de Factura Electrónica (WSFE)

Este servicio es ofrecidos por la AFIP para la facturación electrónica.

Para poder acceder, se debe obtener un Ticket de Acceso (TA) previamente (para utlizar el Token y Sign y enviarlos a los métodos de facturación electrónica).

Esta interfase recibe los valores y crea los mensaje en XML, lo envía y analiza la respuesta XML, por lo que no es necesario modificar XML. Se recomienda pasar los valores en formato string, números separados por punto y fechas YYYYMMDD.

Para más información ver especificaciones técnicas del WSFE en el sitio de la AFIP:

http://wswhomo.afip.gov.ar/fiscaldocs/WSFE/WSFE-ManualParaElDesarrollador-090317.pdf

Objeto

- El objeto COM se crea invocando a CreateObject("WSFE")

Métodos

- **`Conectar(url)`**: realiza la conexión a los servidores de la AFIP (primer paso esencial). Si no se especifica ur, se utiliza servidores de homologación.
- **`Dummy()`**: servicio de prueba para obtener el estado de los servidores de la AFIP.

- **`UltNro()`**: recuperar el último número de transacción (id). Devuelve el último valor de secuencia utilizado al pedir autorización. Uso opcional.
- **`Aut(id, presta_serv, tipo_doc, nro_doc, tipo_cbte, punto_vta, cbt_desde, cbt_hasta, imp_total, imp_tot_conc, imp_neto, impto_liq, impto_liq_rni, imp_op_ex, fecha_cbte, fecha_venc_pago, [fecha_serv_desde], [fecha_serv_hasta])`**: autoriza la emisión de factura electrónica, recibe el id de secuencia (recuperado por UltNro), presta_serv en 1 si se esta facturando servicios (en este caso es obligatorio el uso de la fecha de servicio desde y hasta), los datos de la factura a emitir, y devuelve el Código de Autorización Electrónico (CAE). Ver ejemplo para el detalle de los parámetros.
- **`RecuperaLastCMP(punto_venta, tipo_comprobante)`**: recibe el punto de venta y tipo de comprobante (strings/enteros), devuelve el último número de comprobante autorizado. Uso opcional.
- **`RecuperaQty()`**: recupera cantidad máxima de registros de detalle (por el momento no es de mayor utilidad ya que por simplicidad se envía 1 registro de detalle por autorización). Uso opcional.

Atributos

- **`Token`**: es el código de autorización generado por la AFIP (WSAA)
- **`Sign`**: es la firma de autorización generado por la AFIP (WSAA)
- **`Cuit`**: es el número de CUIT del emisor de facturas, formato string sin guiones.
- **`AppServerStatus`**, **`DbServerStatus`**, **`AuthServerStatus`**: estados de los servidores de AFIP (string ?OK? en caso de estar funcionales)
- **`XmlRequest`**, **`XmlResponse`**: requerimiento y respuesta XML (para depuración)
- **`Respuesta`**, **`Motivo`**, **`Reproceso`**: valores complementarios que retorna Aut
- **`CbtDesde`**, **`CbtHasta`**, **`'''FechaCbte?`**: números y fecha de comprobante autorizado que retorna Aut
- **`ImpTotal`**, **`ImpNeto`**, **`'''ImptoLiq?`**: importes total, neto e IVA que retorna Aut
- **`CAE`**, **`Vencimiento`**: CAE y Fecha de vencimiento autorización retorna Aut
- **`Version`**: versión de la interfase (ej. ?1.6?)

Errores

La interfase procesa los errores WSFE y los informa con códigos numéricos (Err.Number ? vbObjectError = nº de error de AFIP) y su correspondiente descripción (Err.Description). Entre ellos los más comunes son:

- 1000: Usuario no autorizado a realizar esta operación. Revisar Token y Sign obtenido del WSAA
- 1001: CUIT no autorizado. Revisar en el sitio de la AFIP si se asoció el CUIT del emisor al servicio de factura electrónica, con el correspondiente certificado.
- 1012,1014,1016,1017,1018,1019: Inconsistencia en los datos enviados. Revisar formatos (fechas, números, etc., usar fechas YYYYMMDD y números separados por puntos, ambos pasarlos como string para evitar errores). Revisar tipo de datos (tipo_cbte, tipo_doc) que correspondan a las tablas usadas por la AFIP. Revisar que los datos sean válidos para la operación a realizar (ej. Factura A solo para tipo_doc = 80 CUIT, Facturas B por lote hasta $1000.-, etc.)

Ver especificaciones técnicas del WSFE para más información sobre el uso del servicio web, su formato, condiciones y errores.

Ejemplo

Autorizar una factura electrónica nacional a modo de ejemplo (ficticia):

```
' Crear objeto interface Web Service de Factura Electrónica

Set WSFE = CreateObject("WSFE")

' Setear tocken y sing de autorización (pasos previos)
WSFE.Token = WSAA.Token
WSFE.Sign = WSAA.Sign
' CUIT del emisor (debe estar registrado en la AFIP)
WSFE.Cuit = "20267565393"

' Conectar al Servicio Web de Facturación (servidor de homologación)
ok = WSFE.Conectar("https://wswhomo.afip.gov.ar/wsfe/service.asmx")

' Llamo a un servicio ?tonto?, para obtener el estado (opcional)
WSFE.Dummy
Debug.Print "appserver status", WSFE.AppServerStatus
Debug.Print "dbserver status", WSFE.DbServerStatus
Debug.Print "authserver status", WSFE.AuthServerStatus

' Recupera cantidad máxima de registros (opcional)
qty = WSFE.RecuperarQty()

' Recupera último número de secuencia ID
LastId = WSFE.UltNro()

' Recupero último número de comprobante (opcional)
tipo_cbte = 1: punto_vta = 1
LastCBTE = WSFE.RecuperaLastCMP(punto_vta, tipo_cbte)

' Establezco los valores de la factura o lote a autorizar: "100.00"
impto_liq = "21.00": impto_liq_rni = "0.00": imp_op_ex = "0.00"
fecha_cbte = Fecha: fecha_venc_pago = Fecha
' Fechas del período del servicio facturado (solo si presta_serv = 1)
fecha_serv_desde = Fecha: fecha_serv_hasta = Fecha

' Llamo al WebService de Autorización para obtener el CAE
cae = WSFE.Aut(id, presta_serv, _
  tipo_doc, nro_doc, tipo_cbte, punto_vta, _
  cbt_desde, cbt_hasta, imp_total, imp_tot_conc, imp_neto, _
  impto_liq, impto_liq_rni, imp_op_ex, fecha_cbte, fecha_venc_pago,
  fecha_serv_desde, fecha_serv_hasta) ' solo pasar si presta_serv=0
MsgBox "CAE: " & cae & " Vencimiento: " & WSFE.Vencimiento
Exit Sub

ManejoError:
' Si hubo error:
Debug.Print Err.Description            ' descripción error afip
Debug.Print Err.Number - vbObjectError ' codigo error afip
MsgBox Err.Description, vbCritical + vbRetryCancel, _
  "Error:" & (Err.Number - vbObjectError) & " en " & Err.Source
```

Nota: no es necesario (ni recomendado) recuperar el último ID y/o último número de comprobante. Dichos datos deberían estar almacenados de manera permanente en la aplicación que use la interfaz (ver aclaración).

Descripción del Método Aut (obtención de CAE)

Se debe llamar al método Aut con los siguiente parámetros:

- id: Número de identificación secuencial (debe almacenarse en el sistema local) No es obligatorio llamar a UltNro, puede utilizarse un dato local secuencial.
- presta_serv: 0 o 1 para indicar si es una factura de servicios
- tipo_doc, nro_doc: Tipo (80 CUIT, 96 DNI, etc.) y número de Documento
- tipo_cbte: Tipo de comprobante (1 Factura A , 6 Factura B, etc.)
- punto_vta: Nº de punto de venta (debe estar autorizado)
- cbt_desde, cbt_hasta: Nº de comprobante (si es un solo comprobante, repetir Nº)
- imp_total: Importe total de la factura
- imp_tot_conc: Importe total de conceptos no gravados por el IVA
- imp_neto: Importe neto (gravado por el IVA) de la factura
- impto_liq: Importe del IVA liquidado (incluyendo percepciones de IVA)
- impto_liq_rni: Importe IVA RNI (no se utiliza mas, dejar 0.00)
- imp_op_ex: Importe de operaciones exentas
- fecha_cbte: Fecha del comprobante (no puede ser mayor o menor a 5 días)
- fecha_venc_pago: Fecha límite de vencimiento para el pago de la factura
- fecha_serv_desde, fecha_serv_hasta: Fechas del período de servicios prestado

Ver todos los tipos de comprobante: http://wswhomo.afip.gov.ar/fiscaldocs/WSFE/CodCpbte.txt

Ver todos los tipos de documentos: http://wswhomo.afip.gov.ar/fiscaldocs/WSFE/CodDocID.txt

Si no hubiere inconvenientes, la llamada debe devolver el CAE y se establece el atributo WSFE.Resultado = "A" (Aceptado) y WSFE.Vencimiento. Sino, devuelve:

- cae = "": La interfase no pudo procesar la respuesta del WebService, o bien la conexión a internet esta caída, las direcciones de los servidores son incorrectas, o los servidores de la AFIP no funcionan o no devuelven la respuesta esperada.
- cae = "NULL": El servicio web rechazo los datos de la factura a generar. Se establece el atributo WSFE.Resultado = "R" (Rechazado) y WSFE.Motivo con los diversos motivos de rechazo proporcionados por el webservice:
    - WSFE.Motivo = "00" o "NULL": no hay error (solo como referencia)
    - WSFE.Motivo = "01": CUIT informada no es R.I.
    - WSFE.Motivo = "02": CUIT no autorizada a facturar electrónicamente
    - WSFE.Motivo = "03": CUIT registra inconvenientes con domicilio fiscal
    - WSFE.Motivo = "04": Punto de venta no se encuentra declarado
    - WSFE.Motivo = "05": Fecha del comprobante incorrecta
    - WSFE.Motivo = "06": CUIT no puede emitir comprobantes clase A
    - WSFE.Motivo = "07": para comprobantes clase A debe indicar CUIT
    - WSFE.Motivo = "08": CUIT informada es inválida
    - WSFE.Motivo = "09": CUIT informada no existe en el padrón
    - WSFE.Motivo = "10": CUIT informada no corresponde a un R.I.
    - WSFE.Motivo = "11": el nº de comprobante no es correlativo o fecha no corresponde
    - WSFE.Motivo = "12": el rango informado se encuentra autorizado
    - WSFE.Motivo = "13": *LA CUIT INDICADA SE ENCUENTRA COMPRENDIDA EN EL REGIMEN ESTABLECIDO POR LA RESOLUCION GENERAL N° 2177 Y/O EN EL TITULO I DE LA RESOLUCION GENERAL N° 1361 ART. 24 DE LA RG N° 2177-*

Para ver la lista completa de códigos de motivos F136:
http://wswhomo.afip.gov.ar/fiscaldocs/WSFE/F136_MOTIVOS.txt

Aún devolviendo el CAE y con el resultado Aceptado, puede haber un motivo (ej 01), que funcionan como advertencia para revisar la base de datos y corregir posibles irregularidades.

Advertencia: En el caso de que la conexión se interrumpa luego de que la AFIP registre la factura y antes de que la información del CAE llegue al equipo, la única alternativa es repetir la llamada a Aut con los mismos datos (incluido el ID original) para obtener el CAE generado anteriormente. En ese caso, se WSFE.Reproceso es ?S? para indicar que el CAE se ha recuperado. Por esto, es importante guardar el ID o utilizar un dato local.

Ver http://wswhomo.afip.gov.ar/fiscaldocs/WSFE/WSFE-GuiaAdicionalParaElProgramador.pdf

---

Servicio Web de Bono Fiscal Electrónico (WSBFE)

EL WSBFE (Bonos Fiscales Electrónicos) es un Servicio Web de la AFIP para Facturas Electrónicas de Bienes de Capital, correspondiente al Artículo 3 de la Resolución General 2557/2009,Este servicio es ofrecidos por la AFIP para la facturación electrónica.

Para poder acceder, se debe obtener un Ticket de Acceso (TA) previamente (para utlizar el Token y Sign y enviarlos a los métodos de facturación electrónica).

Esta interfase recibe los valores y crea los mensaje en XML, lo envía y analiza la respuesta XML, por lo que no es necesario modificar XML. Se recomienda pasar los valores en formato string, números separados por punto y fechas YYYYMMDD.

La operatoria es bastante similar al método de autorización del WSFE (con un ID secuencial, observaciones, reproceso, etc.), teniendo en cuenta esta mayor complejidad por tener que informar el detalle de cada item.

A su vez, el WSBFE devuelve mensajes de eventos (mantenimiento programado, advertencias, etc.), los que deben ser capturados e informados al usuario.

A diferencia del WSFE, este nuevo servicio funciona con tablas dinámicas de parámetros para los códigos de comprobante, moneda, alícuotas de iva, producto según NCM, zonas, unidades de medida. Estas tablas pueden sufrir modificaciones realizadas por la AFIP, con altas y bajas lógicas, por lo que tienen una fecha de vigencia (desde, hasta) y se proveen métodos para consultarlas por el mismo servicio web (a diferencia del WSFE, que las tablas eran documentadas estáticamente en el sitio web).

Para más información ver especificaciones técnicas del WSBFE en el sitio de la AFIP:

http://www.afip.gov.ar/ws/WSBFE/WSBFE-ManualParaElDesarrollador.pdf

http://www.afip.gov.ar/ws/WSBFE/WSBFE-GuiaAdicionalParaElProgramador.pdf

La programación de la interfase WSBFE es similar a WSFE, se recomienda revisarlo.

Objeto

- El objeto COM se crea invocando a CreateObject("WSBFE")

Métodos

- **`Conectar(url)`**: realiza la conexión a los servidores de la AFIP (primer paso esencial). Si no se especifica ur, se utiliza servidores de homologación.
- **`Dummy()`**: servicio de prueba para obtener el estado de los servidores de la AFIP.
- **`CrearFactura(tipo_doc, nro_doc, zona, tipo_cbte, punto_vta, cbte_nro, fecha_cbte, imp_total, imp_neto, impto_liq, imp_tot_conc, impto_liq_rni, imp_op_ex, imp_perc, imp_iibb, imp_perc_mun, imp_internos, imp_moneda_id, Imp_moneda_ctz, cancela_misma_moneda_ext, condicion_iva_receptor_id)`**: crea internamente una factura para luego poder autorizarla, recibe los datos de la factura a emitir. Ver ejemplo para el detalle de los parámetros.
- **`AgregarItem(ncm, sec, ds, qty, umed, precio, bonif, iva_id, imp_total)`**: agrega internamente un item (linea de factura) a una factura para luego poder autorizarla, recibe los datos del item a factura a emitir. Ver ejemplo para el detalle de los parámetros.
- **`Authorize(id)`**: autoriza la emisión de factura electrónica para bono fiscal electrónico, recibe el id de secuencia (recuperado por WSFE.UltNro), y devuelve el Código de Autorización Electrónico (CAE). Ver ejemplo para el detalle de los parámetros.
- **`GetCMP(tipo_cbte, punto_vta, cbte_nro)`**: recupera los datos de una factura autorizada, recibe tipo de comprobante, punto de venta y número de comprobante original, y devuelve el Código de Autorización Electrónico (CAE) obtenido en su momento. A su vez, establece los datos de la factura (Cae, FechaCbte, ImpTotal, ImpNeto, ImptoLiq). Ver ejemplo para el detalle de los parámetros y valores devueltos.
- **`GetLastCMP(tipo_cbte, punto_vta)`**: recupera el último número de factura autorizada, recibe tipo de comprobante y punto de venta. Ver WSFE.RecuperaLastCMP
- **`GetLastID()`**: recupera el mayor número de secuencia (id) utilizado. Ver WSFE.UltNro
- **`GetParamMon()`**, **`GetParamTipoCbte()`**, **`GetParamTipoIVA()`**, **`GetParamUMed()`**, **`GetParamNCM()`**: recupera valores referenciales de códigos de las tablas de parámetros, devuelve una lista de strings con el id/código, descripción del parámetro y vigencia -si corresponde- (ver ejemplos). Más información en Tablas de Parámetros

Atributos

- **`Token`**: es el código de autorización generado por la AFIP (WSAA)
- **`Sign`**: es la firma de autorización generado por la AFIP (WSAA)
- **`Cuit`**: es el número de CUIT del emisor de facturas, formato string sin guiones.
- **`AppServerStatus`**, **`DbServerStatus`**, **`AuthServerStatus`**: estados de los servidores de AFIP (string ?OK? en caso de estar funcionales)
- **`XmlRequest`**, **`XmlResponse`**: requerimiento y respuesta XML (para depuración)
- **`Respuesta`**, **`Obs`**, **`Reproceso`**: valores complementarios que retorna Aut
- **`CAE`**, **`Vencimiento`**: CAE y Fecha de vencimiento autorización retorna Aut
- **`Version`**: versión de la interfase (ej. ?1.11?)
- **`FechaCbte`**: fecha del comprobante (del comprobante recuperado devuelto por GetCmp)
- **`ImpTotal`**: importe total del comprobante (del comprobante recuperado devuelto por GetCmp)
- **`ImpNeto`**: importe neto del comprobante (del comprobante recuperado devuelto por GetCmp)
- **`ImptoLiq`**: impuesto liquidado (IVA) del comprobante (del comprobante recuperado devuelto por GetCmp)

Errores

La interfase procesa los errores WSFE y los informa con códigos numéricos (Err.Number ? vbObjectError = nº de error de AFIP) y su correspondiente descripción (Err.Description). Entre ellos los más comunes son:

- 1000: Usuario no autorizado a realizar esta operación. Revisar Token y Sign obtenido del WSAA
- 1001: CUIT no autorizado. Revisar en el sitio de la AFIP si se asoció el CUIT del emisor al servicio de factura electrónica, con el correspondiente certificado.
- 1012,1014,1016,1017,1018,1019: Inconsistencia en los datos enviados. Revisar formatos (fechas, números, etc., usar fechas YYYYMMDD y números separados por puntos, ambos pasarlos como string para evitar errores). Revisar tipo de datos (tipo_cbte, tipo_doc) que correspondan a las tablas usadas por la AFIP. Revisar que los datos sean válidos para la operación a

Ver especificaciones técnicas del WSBFE para más información sobre el uso del servicio web, su formato, condiciones y errores.

Ejemplo

Autorizar una factura de bienes de capital (bono fiscal electrónico) a modo de ejemplo (ficticia):

```
Dim WSAA As Object, WSBFE As Object

' Crear objeto interface Web Service Autenticación y Autorización
Set WSAA = !CreateObject("WSAA")

' Generar un Ticket de Requerimiento de Acceso (TRA) para WSBFE
tra = WSAA.CreateTRA("wsbfe")

' Especificar la ubicacion de los archivos certificado y clave privada
Path = !CurDir() + "\"
' Certificado: certificado es el firmado por la AFIP
' ClavePrivada: la clave privada usada para crear el certificado
Certificado = "reingart.crt" ' certificado de prueba
ClavePrivada = "reingart.key" ' clave privada de prueba

' Generar el mensaje firmado (CMS)
cms = WSAA.SignTRA(tra, Path + Certificado, Path + !ClavePrivada)
' Llamar al web service para autenticar:
ta = WSAA.CallWSAA(cms, "!https://wsaahomo.afip.gov.ar/ws/services/LoginCms") ' Homologación
' Una vez obtenido, se puede usar el mismo token y sign por 24 horas

' Crear objeto interface Web Service de Factura Electrónica
Set WSBFE = !CreateObject("WSBFE")

' Setear tocken y sing de autorización (pasos previos)
WSBFE.Token = WSAA.Token
WSBFE.Sign = WSAA.Sign
' CUIT del emisor (debe estar registrado en la AFIP)
WSBFE.Cuit = "20267565393"

' Conectar al Servicio Web de Facturación
ok = WSBFE.Conectar("!http://wswhomo.afip.gov.ar/wsbfe/service.asmx") ' homologación

' Establezco los valores de la factura a autorizar:
fecha = "20090530"
tipo_doc = 80: nro_doc = "23111111113"
zona = 0 ' Ver tabla de zonas
```

```
tipo_cbte = 1 ' Ver tabla de tipos de comprobante
punto_vta = 2: cbte_nro = 16
fecha_cbte = fecha
imp_total = "121.00": imp_tot_conc = "0.00": imp_neto = "100.00"
impto_liq = "21.00": impto_liq_rni = "0.00": imp_op_ex = "0.00"
imp_perc = "0.00": imp_iibb = "0.00": imp_perc_mun = "0.00"
imp_internos = "0.00"
imp_moneda_id = "10" ' Ver tabla de tipos de moneda
Imp_moneda_ctz = "1.0000" ' cotización de la moneda

' Creo una factura (internamente, no se llama al !WebService):
ok = WSBFE.!CrearFactura(tipo_doc, nro_doc, _
  zona, tipo_cbte, punto_vta, cbte_nro, fecha_cbte, _
  imp_total, imp_neto, impto_liq, _
  imp_tot_conc, impto_liq_rni, imp_op_ex, _
  imp_perc, imp_iibb, imp_perc_mun, imp_internos, _
  imp_moneda_id, Imp_moneda_ctz)

' Agrego un item:
ncm = "7308.10.00" ' Ver tabla de códigos habilitados del NCM
sec = "" ' Código de la Secretaría (no usado por el momento)
ds = "Prueba" ' Descripción completa del artículo (hasta 4000 car.)
umed = 1 ' kg, Ver tabla de unidades de medida
qty = "1.0" ' cantidad
precio = "105.0" ' precio neto (Fac.A), precio final (Fac..uras B)
bonif = "5.00" ' descuentos (en positivo)
iva_id = 5 ' 21%, ver tabla alícuota de iva
imp_total = "121.00" ' importe total final del artículo
' lo agrego a la factura (internamente, no se llama al !WebService):
ok = WSBFE.!AgregarItem(ncm, sec, ds, qty, umed, precio, bonif, iva_id, imp_total)

' Llamo al !WebService de Autorización para obtener el CAE
cae = WSBFE.Authorize(id)

If cae = "" Or WSBFE.Resultado <> "A" Then
  MsgBox "No se asignó CAE (Rechazado). Observación (motivos): " & WSBFE.Obs, _
    vbInformation + vbOKOnly
ElseIf WSBFE.Obs <> "" And WSBFE.Obs <> "00" Then
  MsgBox "Se asignó CAE pero con advertencias. Observación (motivos): " & WSBFE.Obs, _
    vbInformation + vbOKOnly
End If

MsgBox "!Resultado:" & WSBFE.Resultado & " CAE: " & cae & _
  " Reproceso: " & WSBFE.Reproceso & _
  " Obs: " & WSBFE.Obs, vbInformation + vbOKOnly

' Muestro los eventos (mantenimiento programados y otros mensajes de la AFIP)

For Each evento In WSBFE.Eventos
  If evento <> "0: " Then
    MsgBox "Evento: " & evento, vbInformation
  End If
Next
```

Nota: no es necesario (ni recomendado) recuperar el último ID y/o último número de comprobante. Dichos datos deberían estar almacenados de manera permanente en la aplicación que use la interfaz(ver aclaración).

Ejemplo para obtener códigos referenciales del Nomenclador Común Mercosur habilitados:

```
' recupero tabla del nomenclador común del mercosur
```

```
' ("codigo: descripción (vigencia desde - hasta)")
' Ej: "99.99.99.99: (item no incluído en el Beneficio Fiscal) (20070524 - 20091231)"
For Each x In WSBFE.GetParamNCM()
    Debug.Print x
Next
```

Descripción de la obtención de CAE

Como primer paso, se debe crear una factura (utilizada internamente por la interfase para contener los valores a autorizar), llamando al método CrearFactura con los siguiente parámetros:

- tipo_doc, nro_doc: Tipo (80 CUIT, 96 DNI, etc.) y número de Documento
- tipo_cbte: Tipo de comprobante (según tabla de parámetros)
- punto_vta: Nº de punto de venta (debe estar autorizado para WSBFE)
- cbt_nro: Nº de comprobante
- fecha_cbte: Fecha del comprobante (no puede ser mayor o menor a 10 días)
- zona_id: Zona (según tabla de parámetros) - Por el momento 0
- imp_total: Importe total de la factura
- imp_tot_conc: Importe total de conceptos no gravados por el IVA
- imp_neto: Importe neto (gravado por el IVA) de la factura
- impto_liq: Importe del IVA liquidado (incluyendo percepciones de IVA)
- impto_liq_rni: Importe IVA RNI (no se utiliza mas, dejar 0.00)
- imp_op_ex: Importe de operaciones exentas
- imp_moneda_id: Moneda de la factura (según tabla de parámetros)
- imp_moneda_ctz: Cotización de la moneda de la factura
- imp_perc: Importe de las precepciones
- imp_iibb: Importe de las percepciones de ingresos brutos
- imp_perc_mun: Importe de las percepciones municipales
- imp_internos: Importe de los impuestos internos

Luego, por cada artículo vendido (ítem), se debe llamar al método AgregarItem con los siguientes parámetros:

- ncm: código habilitado según el Nomenclador Común del Mercosur (NCM)
- sec: código de la secretaría (no utilizado por el momento, pasar ??)
- ds: Descripción completa
- precio: Precio Neto Unitario (facturas A) o precio final (facturas B)
- qty: Cantidad
- umed: Unidad de medida (según tabla de parámetros)
- bonif: Bonificación
- iva_id: Alícuota de IVA (según tabla de parámetros)
- imp_total: Importe total (incluyendo descuentos e IVA)

Como último paso, se debe llamar al método Authorize con los siguiente parámetros:

- id: Número de identificación secuencial (debe almacenarse en el sistema local) No es obligatorio llamar a WSFE.UltNro, puede utilizarse un dato local secuencial.

Si no hubiere inconvenientes, la llamada debe devolver el CAE y se establece el atributo WSBFE.Resultado = "A" (Aceptado) y WSBFE.Vencimiento. Sino, devuelve:

- cae = "": La interfase no pudo procesar la respuesta del WebService, o bien la conexión a internet esta caída, las direcciones de los servidores son incorrectas, o el servicio web rechazo los datos de la factura a generar. Se establece el atributo WSBFE.Resultado = "R" (Rechazado) y WSBFE.Obs con los diversos motivos de rechazo proporcionados por el webservice (ver WSFE.Motivo):

Aún devolviendo el CAE y con el resultado Aceptado, puede haber un motivo (ej 01), que funcionan como advertencia para revisar la base de datos y corregir posibles irregularidades.

Advertencia: Revisar las consideraciones de reproceso al igual que con WSFE.

Servicio Web de Factura Electrónica Exportación (WSFEX)

EL WSFEX es un Servicio Web de la AFIP para Facturas Electrónicas de Exportación, correspondiente a la Resolución General 2758/2010.

Para poder acceder, se debe obtener un Ticket de Acceso (TA) previamente (para utlizar el Token y Sign y enviarlos a los métodos de facturación electrónica).

Esta interfase recibe los valores y crea los mensaje en XML, lo envía y analiza la respuesta XML, por lo que no es necesario modificar XML. Se recomienda pasar los valores en formato string, números separados por punto y fechas YYYYMMDD.

La operatoria es bastante similar al método de autorización del WSBFE (con un ID secuencial, observaciones, reproceso, etc.), teniendo en cuenta esta mayor complejidad por tener que informar los datos de exportación, permisos, comprobantes asociados y el detalle de cada item.

A su vez, el WSFEX devuelve mensajes de eventos (mantenimiento programado, advertencias, etc.), los que deben ser capturados e informados al usuario.

Al igual que el WSBFE, este nuevo servicio funciona con tablas dinámicas de parámetros para los códigos de comprobante, moneda, idiomas, terminos de comercio exterior, paises y cuits, unidades de medida, etc.. Estas tablas pueden sufrir modificaciones realizadas por la AFIP, con altas y bajas lógicas, por lo que tienen una fecha de vigencia (desde, hasta) y se proveen métodos para consultarlas por el mismo servicio web (a diferencia del WSFE, que las tablas eran documentadas estáticamente en el sitio web).

Para más información ver especificaciones técnicas del WSFEX en el sitio de la AFIP:

- Manual para el desarrollador.pdf (version 0): Información importante sobre la operatoria
- Manual para el desarrollador.pdf (version 1): Información importante sobre la operatoria (**Nueva Versión**)
- FacturaElectronicaExportacion: Información general, cambios, ejemplos, descargas y parámetros vigentes (puede estar ligeramente desactualizado)

La programación de la interfase WSFEX es similar a WSBFE, se recomienda revisarlo.

WSFEX Versión 1 (WSFEXv1)

Según RG3066/11 AFIP publicó una nueva versión 1 (WSFEXv1) a entrar en vigencia **obligatoria** el **31-12-2011**, para más información ver Cambios WSFEXv1 respecto a WSFEXv0

Ambos webservices son muy similares, por lo que se documenta en esta sección solo las diferencias introducidas.

Importante: Si bien se mantiene retrocompatibilidad, para conectarse al nuevo webservice es necesario utilizar el objeto `WSFEXv1` y usar las nuevas URL:

- https://wswhomo.afip.gov.ar/wsfexv1/service.asmx?WSDL (homologación)
- https://servicios1.afip.gov.ar/wsfexv1/service.asmx?WSDL (producciòn)

Aclaración: WSFEXv1 es un **nuevo webservice** y AFIP ha agregado campos (bonificación), ha cambiado varios tipos de datos (en importes, cantidad de decimales), ha agregado códigos de tablas de parámetros (unidades de medida) y comprobantes asociados (remitos de tabaco) y además realiza nuevas validaciones, por lo que recomendamos probar exhaustivamente la interfaz con el desarrollo para exportación.

Ajustes desarrollos de WSFEX a WSFEXv1

Si no utiliza las características nuevas, es posible utilizar el código desarrollado para WSFEX con nuestra interfaz WSFEXv1:

Código Anterior (WSFEX):

```
' Crear objeto interface Web Service de Factura Electrónica de Exportación
Set WSFEX = CreateObject("WSFEX")

' Conectar al Servicio Web de Facturación
ok = WSFEX.Conectar("https://wswhomo.afip.gov.ar/wsfex/service.asmx") ' homologación
```

Código Nuevo (WSFEXv1):

```
' Crear objeto interface Web Service de Factura Electrónica de Exportación
Set WSFEX = CreateObject("WSFEXv1")

' Conectar al Servicio Web de Facturación
ok = WSFEX.Conectar("", "https://wswhomo.afip.gov.ar/wsfexv1/service.asmx?WSDL") ' homologación
```

RECEX vs RECEX1

Por linea de comando tenemos disponible la herramienta RECEX1.EXE que reemplaza a RECEX.EXE y utiliza prácticamente el mismo formato de archivo y campos. El nuevo formato esta documentado la sección RECEX1 - WSFEXv1

Objeto

- El objeto COM se crea invocando a CreateObject("WSFEX") para Versión 0 (hasta 30-12-2011)
- El objeto COM se crea invocando a CreateObject("WSFEXv1") para Versión 1 (desde 31-12-2011)

Métodos

- **`Conectar(url)`**: realiza la conexión a los servidores de la AFIP (primer paso esencial). Si no se especifica url, se utiliza servidores de homologación. Importante: para WSFEXv1 los parámetros son cache, url, proxy, httpwrapper y cacert, ver WSFEv1 ***Nuevo!***
- **`Dummy()`**: servicio de prueba para obtener el estado de los servidores de la AFIP.

- **`CrearFactura(tipo_cbte, punto_vta, cbte_nro, fecha_cbte, imp_total, tipo_expo, permiso_existente, dst_cmp, cliente, cuit_pais_cliente, domicilio_cliente, id_impositivo, moneda_id, moneda_ctz, obs_comerciales, obs, forma_pago, incoterms, idioma_cbte, incoterms_ds, fecha_pago,cancela_misma_moneda_ext, condicion_iva_receptor_id)`**: crea internamente una factura para luego poder autorizarla, recibe los datos de la factura a emitir. Ver ejemplo para el detalle de los parámetros. Importante: fecha_pago se agrega en WSFEXv1 ***Modificado!***
- **`AgregarItem(ncodigo, ds, qty, umed, precio, imp_total, bonif)`**: agrega internamente un item (linea de factura) a una factura para luego poder autorizarla, recibe los datos del item a factura a emitir. Ver ejemplo para el detalle de los parámetros. Importante: bonif se agrega en WSFEXv1 ***Modificado!***
- **`AgregarPermiso(id, dst)`**: agrega internamente un permiso de exportación a una factura para luego poder autorizarla, recibe identificación y país destino de la mercaderia. Ver ejemplo para el detalle de los parámetros.
- **`AgregarCmpAsoc(tipo_cbte_asoc, punto_vta_asoc, cbte_nro_asoc, cuit_cbte_asoc)`**: agrega internamente un comprobante asociado a una factura para luego poder autorizarla, recibe tipo de comprobante, punto de venta y número. Ver ejemplo para el detalle de los parámetros. Importante: cuit_cbte_asoc se agrega en WSFEXv1
- **`AgregarActividad(actividad_id)`**: agrega internamente las actividades asociadas a un comprobante. *Disponible desde actualización 01/12/22* ***Nuevo!***
- **`Authorize(id)`**: autoriza la emisión de factura electrónica para bono fiscal electrónico, recibe el id de secuencia (recuperado por WSFE.UltNro), y devuelve el Código de Autorización Electrónico (CAE). Ver ejemplo para el detalle de los parámetros.

Métodos Auxiliares del webservice

- **`GetCMP(tipo_cbte, punto_vta, cbte_nro)`**: recupera los datos de una factura autorizada, recibe tipo de comprobante, punto de venta y número de comprobante original, y devuelve el Código de Autorización Electrónico (CAE) obtenido en su momento. A su vez, establece los datos de la factura (Cae, FechaCbte, ImpTotal, ImpNeto, ImptoLiq). Ver ejemplo para el detalle de los parámetros y valores devueltos.
- **`GetLastCMP(tipo_cbte, punto_vta)`**: recupera el último número de factura autorizada, recibe tipo de comprobante y punto de venta. Ver WSFE.RecuperaLastCMP
- **`GetLastID()`**: recupera el mayor número de secuencia (id) utilizado. Ver WSFE.UltNro
- **`GetParamMon()`**, **`GetParamTipoCbte()`**, **`GetParamTipoExpo()`**, **`GetParamIdiomas()`**, **`GetParamUMed()`**, **`GetParamIncoterms()`**, **`GetParamDstPais()`**, **`GetParamDstCUIT()`**: recupera valores referenciales de códigos de las tablas de parámetros, devuelve una lista de strings con el id/código o CUIT y descripción del parámetro (ver ejemplos). Más información en Tablas de Parámetros
- **`GetParamCtz(moneda_id)`**: devuelve cotización y fecha de la moneda indicada como parámetro
- **`GetParamMonConCotizacion(fecha)`**: devuelve la cotización consultada de la base de datos aduanera de todas las monedas que posean cotización a la fecha indicada.
- **`ParamGetActividades()`**: devuelve los códigos de actividades, sus descripciones y el orden (si es actividad primaria, secundaria, etc) ***Nuevo!***

Atributos

- **`Token`**: es el código de autorización generado por la AFIP (WSAA)
- **`Sign`**: es la firma de autorización generado por la AFIP (WSAA)

- **Cuit**: es el número de CUIT del emisor de facturas, formato string sin guiones.
- **AppServerStatus**, **DbServerStatus**, **AuthServerStatus**: estados de los servidores de AFIP (string ?OK? en caso de estar funcionales)
- **XmlRequest**, **XmlResponse:** requerimiento y respuesta XML (para depuración)
- **Respuesta**, **Obs**, **Reproceso**: valores complementarios que retorna Aut
- **CAE**, **Vencimiento**: CAE y Fecha de vencimiento autorización retorna Aut
- **Vencimiento**: fecha devuelta junto con el CAE
- **CbteNro**: número de comprobante (del comprobante autorizado)
- **FechaCbte**: fecha del comprobante (del comprobante recuperado devuelto por GetCmp)
- **ImpTotal**: importe total del comprobante (del comprobante recuperado devuelto por GetCmp)
- **Version**: versión de la interfase (ej. ?1.11?)
- **Excepcion**, **Traceback** ***Nuevo!*** : mensaje de error y traza de rastreo (para depuración)
- **LanzarExcepciones** ***Nuevo!*** : establece si se deben emitir errores al lenguaje de programación (habilitado por defecto), o serán controlados por el programa (revisando el atributo Excepcion luego de cada método)

Errores

La interfase procesa los errores WSFEX y los informa con códigos numéricos (Err.Number ? vbObjectError = nº de error de AFIP) y su correspondiente descripción (Err.Description). Entre ellos los más comunes son:

- 500: Error interno DB - FEXGetLadt_CMP - get_last: problemas internos de los servidores de AFIP (se debe esperar a que lo solucionen).
- Operation FEXGetLast_CMP not found in WSDL: la URL que se está utilizando no es correcta (verificar URL para el ambiente que se está utilizando)
- 505: Error general (lockeo): problemas internos de los servidores de AFIP (se debe esperar a que lo solucionen).
- 1000: Usuario no autorizado a realizar esta operación. Revisar Token y Sign obtenido del WSAA
- 1001: CUIT no autorizado. Revisar en el sitio de la AFIP si se asoció el CUIT del emisor al servicio de factura electrónica, con el correspondiente certificado.
- 1012,1014,1016,1017,1018,1019: Inconsistencia en los datos enviados. Revisar formatos (fechas, números, etc., usar fechas YYYYMMDD y números separados por puntos, ambos pasarlos como string para evitar errores). Revisar tipo de datos (tipo_cbte, tipo_doc) que correspondan a las tablas usadas por la AFIP. Revisar que los datos sean válidos para la operación a

Ver especificaciones técnicas del WSFEX para más información sobre el uso del servicio web, su formato, condiciones y errores.

Ejemplo

Autorizar una factura electrónica de exportación a modo de ejemplo (ficticia):

```
Dim WSAA As Object, WSFEX As Object

' Crear objeto interface Web Service Autenticación y Autorización
Set WSAA = CreateObject("WSAA")

' Generar un Ticket de Requerimiento de Acceso (TRA) para WSFEX
tra = WSAA.CreateTRA("wsfex")

' Especificar la ubicacion de los archivos certificado y clave privada
Path = CurDir() + "\"
```

```
' Certificado: certificado es el firmado por la AFIP
' ClavePrivada: la clave privada usada para crear el certificado
Certificado = "..\reingart.crt" ' certificado de prueba
ClavePrivada = "..\reingart.key" ' clave privada de prueba

' Generar el mensaje firmado (CMS)
cms = WSAA.SignTRA(tra, Path + Certificado, Path + ClavePrivada)

' Llamar al web service para autenticar: (Homologación)
ta = WSAA.CallWSAA(cms, "https://wsaahomo.afip.gov.ar/ws/services/LoginCms")

' Una vez obtenido, se puede usar el mismo token y sign por 24 horas
' (este período se puede cambiar)

' Crear objeto interface Web Service de Factura Electrónica de Exportación
Set WSFEX = CreateObject("WSFEXv1")
' Setear tocken y sing de autorización (pasos previos)
WSFEX.Token = WSAA.Token
WSFEX.Sign = WSAA.Sign

' CUIT del emisor (debe estar registrado en la AFIP)
WSFEX.Cuit = "20267565393"

' Conectar al Servicio Web de Facturación (homologación)
ok = WSFEX.Conectar("", "https://wswhomo.afip.gov.ar/wsfexv1/service.asmx?WSDL")

' Llamo a un servicio nulo, para obtener el estado del servidor (opcional)
WSFEX.Dummy
Debug.Print "appserver status", WSFEX.AppServerStatus
Debug.Print "dbserver status", WSFEX.DbServerStatus
Debug.Print "authserver status", WSFEX.AuthServerStatus

' Establezco los valores de la factura a autorizar:
tipo_cbte = 19 ' FC Expo (ver tabla de parámetros)
punto_vta = 7
' Obtengo el último número de comprobante y le agrego 1
cbte_nro = WSFEX.GetLastCMP(tipo_cbte, punto_vta) + 1 '16
fecha_cbte = Format(Date, "yyyymmdd")
tipo_expo = 1 ' tipo de exportación (ver tabla de parámetros)
permiso_existente = "N"
dst_cmp = 203 ' país destino
cliente = "Joao Da Silva"
cuit_pais_cliente = "50000000016"
domicilio_cliente = "Rua 76 km 34.5 Alagoas"
id_impositivo = "PJ54482221-l"
moneda_id = "012" ' para reales, "DOL" o "PES" (ver tabla de parámetros)
moneda_ctz = 0.5
obs_comerciales = "Observaciones comerciales"
obs = "Sin observaciones"
forma_pago = "30 dias"
incoterms = "FOB" ' (ver tabla de parámetros)
idioma_cbte = 1 ' (ver tabla de parámetros)
imp_total = "250.00"

' Creo una factura (internamente, no se llama al WebService):
ok = WSFEX.CrearFactura(tipo_cbte, punto_vta, cbte_nro, fecha_cbte, _
        imp_total, tipo_expo, permiso_existente, dst_cmp, _
        cliente, cuit_pais_cliente, domicilio_cliente, _
        id_impositivo, moneda_id, moneda_ctz, _
        obs_comerciales, obs, forma_pago, incoterms, _
        idioma_cbte, incoterms_ds, fecha_pago )
```

```
' Agrego un item:
codigo = "PRO1"
ds = "Producto Tipo 1 Exportacion MERCOSUR ISO 9001"
qty = 2
precio = "125.00"
umed = 1 ' Ver tabla de parámetros (unidades de medida)
imp_total = "250.00" ' importe total final del artículo
' lo agrego a la factura (internamente, no se llama al WebService):
ok = WSFEX.AgregarItem(codigo, ds, qty, umed, precio, imp_total)

' Agrego un permiso (ver manual para el desarrollador)
id = "99999AAXX999999A"
dst = 225 ' país destino de la mercaderia
ok = WSFEX.AgregarPermiso(id, dst)

' Agrego un comprobante asociado (ver manual para el desarrollador)
tipo_cbte_asoc = 19
punto_vta_asoc = 2
cbte_nro_asoc = 1
ok = WSFEX.AgregarCmpAsoc(tipo_cbte_asoc, punto_vta_asoc, cbte_nro_asoc)

'id = "99000000000100" ' número propio de transacción
' obtengo el último ID y le adiciono 1
' (advertencia: evitar overflow y almacenar!)
id = CStr(CCur(WSFEX.GetLastID()) + 1)

' Llamo al WebService de Autorización para obtener el CAE
cae = WSFEX.Authorize(id)

' Verifico que no haya rechazo o advertencia al generar el CAE
If cae = "" Or WSFEX.Resultado <> "A" Then
    MsgBox "No se asignó CAE (Rechazado). Observación (motivos): " & _
              WSFEX.obs, vbInformation + vbOKOnly
ElseIf WSFEX.obs <> "" And WSFEX.obs <> "00" Then
    MsgBox "Se asignó CAE pero con advertencias. Observación (motivos): " & _
              WSFEX.obs, vbInformation + vbOKOnly
End If

' Imprimo pedido y respuesta XML para depuración (errores de formato)
' si es posible almacenar para referencias futuras
Debug.Print WSFEX.XmlRequest
Debug.Print WSFEX.XmlResponse

MsgBox "Resultado:" & WSFEX.Resultado & " CAE: " & cae & _
        " Reproceso: " & WSFEX.Reproceso & _
        " Obs: " & WSFEX.obs, vbInformation + vbOKOnly

' Muestro los eventos (mantenimiento programados y otros mensajes de la AFIP)
For Each evento In WSFEX.Eventos
    If evento <> "0: " Then
        MsgBox "Evento: " & evento, vbInformation
    End If
Next

' Recuperar la factura
cae2 = WSFEX.GetCMP(tipo_cbte, punto_vta, cbte_nro)

Debug.Print "Fecha Comprobante:", WSFEX.FechaCbte
Debug.Print "Importe Total:", WSFEX.ImpTotal

If cae <> cae2 Then
    MsgBox "El CAE de la factura no concuerdan con el recuperado en la AFIP!"
```

```
Else
    MsgBox "El CAE de la factura concuerdan con el recuperado de la AFIP"
End If
```

Nota: no es necesario (ni recomendado) recuperar el último ID y/o último número de comprobante. Dichos datos deberían estar almacenados de manera permanente en la aplicación que use la interfaz(ver aclaración).

Ejemplo para obtener códigos referenciales de países destino habilitados:

```
' recupero tabla de código de pais destino ("codigo: descripción")
' Ej: "203: BRASIL"
For Each x In WSFEX.GetParamDstPais()
    Debug.Print x
Next
```

Ejemplo para obtener la fecha y cotización de una moneda:

```
' busco la cotización del dolar (ver Parametro Mon)
' Ej: "20100708: 3.937"
moneda_id = "DOL"
ctz = WSFEX.GetParamCtz(moneda_id)
MsgBox "Cotización Dólar: " & ctz
```

Descripción de la obtención de CAE

Como primer paso, se debe crear una factura (utilizada internamente por la interfase para contener los valores a autorizar), llamando al método CrearFactura con los siguiente parámetros:

- tipo_cbte: código de comprobante (19: 'Facturas de Exportación', 20: 'Nota de Débito por Operaciones con el Exterior', 21: 'Nota de Crédito por Operaciones con el Exterior'}
- punto_vta: Nº de punto de venta (debe estar autorizado para WSFEX)
- cbte_nro: Nº de comprobante
- fecha_cbte: Fecha del comprobante (no puede ser mayor o menor a 10 días)
- imp_total: Importe total de la factura
- tipo_expo: Tipo de exportacion (1: 'Exportación definitiva de Bienes', 2: 'Servicios', 4: 'Otros')
- permiso_existente: Indica si se posee documento aduanero de exportación (permiso de embarque). Posibles Valores: 'S', 'N', 'NULL' (vacío)
- dst_cmp: País de destino del comprobante (200: 'ARGENTINA', 203: 'BRASIL', 212: 'ESTADOS UNIDOS', etc.)
- cliente: Apellido y Nombre ó Razón Social del comprador
- cuit_pais_cliente: CUIT del país destino/Contribuyente (Ej: 50000000059: 'BRASIL - Persona F\xedsica', 51600000059: 'BRASIL - Otro tipo de Entidad', etc.)
- domicilio_cliente: Domicilio comercial cliente.
- id_impositivo: Clave de identificación tributaria del comprador.
- moneda_id: Moneda de la factura ('DOL': 'Dólar Estadounidense', 'PES': 'Pesos Argentinos', '012': 'Real', etc.)
- moneda_ctz: Cotización de la moneda de la factura
- obs_comerciales: observaciones comerciales (texto arbitrario)
- obs: observaciones (texto arbitrario)
- forma_pago: texto arbitrario (ej '30 días')
- incoterms: clausula de venta, terminos de comercio exterior ('DAF', 'DDP', 'CIF', 'FCA', 'FAS', 'DES', 'CPT', 'EXW', 'CIP', 'DDU', 'FOB', 'DEQ', 'CFR')

- incoterms_ds: información adicional de terminos comerciales
- idioma_cbte: idioma del comprobante {1: 'Español', 2: 'Inglés', 3: 'Portugués'}
- fecha_pago: Fecha de pago (yyyymmdd)

Luego, por cada artículo vendido (ítem), se debe llamar al método AgregarItem con los siguientes parámetros:

- codigo: código del producto
- ds: Descripción completa
- precio: Precio Unitario
- qty: Cantidad
- umed: Unidad de medida (según tabla de parámetros)
- imp_total: Importe total
- bonif: bonificacion sobre el producto (importe)

Consideraciones especiales (sólo WSFEXv1, ***Nuevo!***):

- Para descuentos, utilizar umed 99 y los importes en negativo, sin cantidad ni precio
- Para señas o adelantos, utilizar umed 97 y los importes en negativo o positivo (según corresponda), sin cantidad ni precio
- Para gratificaciones, utilizar umed 0 sin importes, cantidad ni precio
- Para bonificaciones por item, el cálculo es precio * qty - bonif

Adicionalmente se puede llamar al método AgregarPermiso para detallar los permisos de embarque y destinaciones de la mercadería, con los siguientes parámetros:

- id: Código de despacho ? Permiso de Embarque, formato 99999AAXX999999A (donde XX podrán ser números o letras)
- dst: País de destino de la mercadería

También se puede llamar al método AgregarCmpAsoc para detallar los permisos de embarque y destinaciones de la mercadería, con los siguientes parámetros:

- cbte_tipo: Código de tipo de comprobante (remitos solo WSFEXv1, ***Nuevo!***)
- cbte_punto_venta: Punto de venta
- cbte_numero: Numero de comprobante
- cbte_cuit: CUIT emisor del comprobante (solo WSFEXv1, ***Nuevo!***)

Como último paso, se debe llamar al método Authorize con los siguiente parámetros:

- id: Número de identificación secuencial (debe almacenarse en el sistema local) No es obligatorio llamar a WSFEX.!GetLastID, puede utilizarse un dato local secuencial.

Si no hubiere inconvenientes, la llamada debe devolver el CAE y se establece el atributo WSFEX.Resultado = "A" (Aceptado) y WSFEX.Vencimiento. Sino, devuelve:

- cae = "": La interfase no pudo procesar la respuesta del WebService, o bien la conexión a internet esta caída, las direcciones de los servidores son incorrectas, o el servicio web rechazo los datos de la factura a generar. Se establece el atributo WSFXE.Resultado = "R" (Rechazado) y WSFEX.Obs con los diversos motivos de rechazo proporcionados por el webservice (ver WSFE.Motivo):

Aún devolviendo el CAE y con el resultado Aceptado, puede haber un motivo (ej 01), que funcionan como advertencia para revisar la base de datos y corregir posibles irregularidades.

Advertencia: Revisar las consideraciones de reproceso al igual que con WSFE.

IMPORTANTE: Revisar las especificaciones técnicas de la AFIP en el Manual para el desarrollador.pdf por lo relativo a tablas de parámetros, métodos adicionales y operatoria en general (incluyendo obligatoriedad, formatos de los campos y validaciones)

Servicio Web de Factura Electrónica Mercado Interno Versión 1 (WSFEv1)

EL WSFEv1 es un Servicio Web de la AFIP para Facturas Electrónicas de Mercado Interno, correspondiente al Artículo 4 Opción B (sin detalle de factura) de la Resolución General 2904/2010 - RG2485/08 obligatorio a partir de 1-7-2011.

Para poder acceder, se debe obtener un Ticket de Acceso (TA) previamente (para utlizar el Token y Sign y enviarlos a los métodos de facturación electrónica).

Esta interfase recibe los valores y crea los mensaje en XML, lo envía y analiza la respuesta XML, por lo que no es necesario modificar XML. Se recomienda pasar los valores en formato string, números separados por punto y fechas YYYYMMDD.

La operatoria es bastante similar al método de autorización del WSFE (pero **SIN** un ID secuencial ni reproceso), teniendo en cuenta esta mayor complejidad por tener que informar el detalle de los impuestos (IVA) y tributos.

A su vez, el WSFEv1 devuelve mensajes de eventos (mantenimiento programado, advertencias, etc.), los que deben ser capturados e informados al usuario.

A diferencia del WSFE, este nuevo servicio funciona con tablas dinámicas de parámetros para los códigos de comprobante, moneda, alícuotas de iva. Estas tablas pueden sufrir modificaciones realizadas por la AFIP, con altas y bajas lógicas, por lo que tienen una fecha de vigencia (desde, hasta) y se proveen métodos para consultarlas por el mismo servicio web (a diferencia del WSFE, que las tablas eran documentadas estáticamente en el sitio web).

Para más información ver:

http://www.sistemasagiles.com.ar/trac/wiki/ProyectoWSFEv1

La programación de la interfase WSFEv1 es similar a WSFE, se recomienda revisarlo.

Objeto

- El objeto COM se crea invocando a CreateObject("WSFEv1")

Métodos

Métodos básicos de WSFEv1

- **`Conectar(cache, wsdl, proxy, wrapper, cacert)`**: realiza la conexión a los servidores de la AFIP (primer paso esencial). Si no se especifica url del wsdl, se utiliza servidores de homologación. El parámetro cache es un directorio donde se almacenan internamente la descripción del servicio (archivo WSDL) para mayor optimización. Parametros adicionales optativos: wrapper es la librería HTTP a utilizar y cacert la ruta al certificado de la autoridad de certificante del servidor (CA)

- **`CrearFactura(concepto, tipo_doc, nro_doc, tipo_cbte, punto_vta, cbt_desde, cbt_hasta, imp_total, imp_tot_conc, imp_neto, imp_iva, imp_trib, imp_op_ex, fecha_cbte, fecha_venc_pago, fecha_serv_desde, fecha_serv_hasta, moneda_id, moneda_ctz, caea, fecha_hs_gen, cancela_misma_moneda_ext, condicion_iva_receptor_id)`**: crea internamente una factura para luego poder autorizarla, recibe los datos de la factura a emitir. Ver ejemplo para el detalle de los parámetros.
- **`AgregarIva(iva_id, base_imp, importe)`**: agrega internamente un subtotal de IVA a una factura para luego poder autorizarla, recibe los datos del tipo de alícuota, base imponible e importe. Ver ejemplo para el detalle de los parámetros.
- **`AgregarTributo(tributo_id, Desc, base_imp, alic, importe)`**: agrega internamente un subtotal de tributo a una factura para luego poder autorizarla, recibe los datos del impuesto nacional, provincial o municipal (descripción), base imponible e importe. Ver ejemplo para el detalle de los parámetros.
- **`AgregarCmpAsoc(tipo_cbte_asoc, punto_vta_asoc, cbte_nro_asoc, cuit, fecha)`**: agrega internamente un comprobante asociado a una factura para luego poder autorizarla, recibe tipo de comprobante, punto de venta y número. Ver ejemplo para el detalle de los parámetros. El campo `cuit` fué agregado según FEv2.9 13-03-2017, *disponible desde actualización 1.19a* y refiere al CUIT emisor del comprobante asociado. El campo `fecha` fué agregado según FEv2.13 20-02-2019, *disponible desde actualización 1.22a* y refiere a la fecha del comprobante asociado,
- **`AgregarOpcional(opcional_id, valor)`**: agrega internamente un dato opcional a una factura para luego poder autorizarla, recibe el id del tipo de dato opcional y valor (dependiendo de si se trata de proyectos promovidos, bienes usados RG 3411, DD.JJ. F8001 "presunción de no vinculación con la actividad gravada" RG3668/14). Ver tabla de parámetros. *Disponible desde actualización 1.15a*
- **`AgregarComprador(doc_tipo=80, doc_nro=0, porcentaje=100.00)`**: agrega internamente un dato de comprador a una factura para luego poder autorizarla, recibe el tipo de documento (80 CUIT), número de documento y porcentaje (Bienes Usados Registrables RG 4109 E). Ver COMPG 2.10. *Disponible desde actualización 1.20a*
- **`EstablecerCampoFactura(campo, valor)`**: establece individualmente el valor de un campo del encabezado de la factura (ver parámetros del método `CrearFactura`), devuelve True si el campo pertenece al encabezado y se ha actualizado correctamente (ver ejemplo). *Disponible desde actualización 1.16a*
- **`ObtenerCampoFactura(campo, [[campo2, [campo3]])`**: devuelve individualmente el valor de un campo del encabezado u otras sub-estructuras de la factura (ver ejemplo). Útil luego de llamara a `CompConsultar`. *Disponible desde actualización 1.17a*
- **`AgregarPeriodoComprobantesAsociados(fecha_desde, fecha_hasta)`**: agrega internamente un periodo asociado a un comprobante N/C o N/D para luego poder autorizarla, recibe fecha desde y hasta. *Disponible desde actualización 1.25b 20/06/20*

- **`AgregarActividad(actividad_id)`**: agrega internamente las actividades asociadas a un comprobante. *Disponible desde actualización 10/11/22* ***Nuevo!***

Aclaración para Version 4 ~ RG 5616

Para mantener compatibilidad con version anterior también se pueden establecer los nuevos campos por separado:

- ok = wsfev1.EstablecerCampoFactura?("cancela_misma_moneda_ext", "N")
- ok = wsfev1.EstablecerCampoFactura?("condicion_iva_receptor_id", "1")

Importante: en `condicion_iva_receptor_id` es obligatorio a partir de la Version 4, pero no debe ser enviado hasta que este en produccion. En homologacion ya se encuentra habilitado.

Métodos principales de WSFEv1

- **`CAESolicitar()`**: autoriza la emisión de factura electrónica, devuelve el Código de Autorización Electrónico (CAE). Ver ejemplo para el detalle de los parámetros.
- **`CompConsultar(tipo_cbte, punto_vta, cbte_nro)`**: recupera los datos de una factura autorizada, recibe tipo de comprobante, punto de venta y número de comprobante original, y devuelve el Código de Autorización Electrónico (CAE) obtenido en su momento. A su vez, establece los datos de la factura (Cae, FechaCbte, ImpTotal, ImpNeto). Ver ejemplo para el detalle de los parámetros y valores devueltos.
- **`CompUltimoAutorizado(tipo_cbte, punto_vta)`**: recupera el último número de factura autorizada, recibe tipo de comprobante y punto de venta. Ver `WSFE.RecuperaLastCMP`
- **`CAEASolicitar(periodo, orden)`**: solicita un CAE Anticipado para la quincena correspondiente. Devuelve cadena vacia "" si no se ha solicitado. A partir de "COMPGv2.6" establece el campo `Observaciones`
- **`CAEAConsultar(periodo, orden)`**: recupera un CAE Anticipado emitido con anterioridad. Devuelve cadena vacia "" si no se ha solicitado.
- **`CAEARegInformativo`**: informa un comprobante emitido con CAE Anticipado. Se debe crear el comprobante previamente con `CrearFactura` consignando el CAEA. Actualiza los atributos (Resultado, Obs, etc.) de manera similar a `CAESolicitar`. Ver ejemplo para el detalle de los parámetros y valores devueltos.

Métodos alternativos para solicitud de múltiples CAE

- **`IniciarFacturasX()`**: inicializa lista interna de facturas (comprobantes) para Solicitar múltiples CAE, (llamar una única vez antes de `CrearFactura`/`AgregarFacturaX`)
- **`AgregarFacturaX()`**: agrega el último comprobante cargado internamente con `CrearFactura` a la lista para Solicitar múltiples CAE
- **`CAESolicitarX()`**: autoriza la emisión de múltiples facturas electrónicas (varios CAE por solicitud), devuelve la cantidad de registros enviados/recibidos a AFIP.
- **`LeerFacturaX(indice)`**: activa internamente una factura (dada su posición en la lista) para revisar los datoss devuelto por AFIP usando `ObtenerCampoFactura`, luego de haber llamado a `CAESolicitarX`

Métodos Auxiliares del webservice

- **`Dummy()`**: servicio de prueba para obtener el estado de los servidores de la AFIP.

- **ParamGetTiposMonedas()**, **ParamGetTiposCbte()**, **ParamGetTiposDoc()**, **ParamGetTiposIva()**, **ParamGetTiposOpcional()**, **ParamGetTiposTributos()**, **ParamGetTiposPaises()**: recupera valores referenciales de códigos de las tablas de parámetros, devuelve una lista de strings con el id/código, descripción del parámetro y vigencia -si corresponde- (ver ejemplos). Más información en Tablas de Parámetros. **ParamGetTiposPaises** agregado para "COMPGv2.6"
- **ParamGetCotizacion(moneda_id)**: devuelve cotización y fecha de la moneda indicada como parámetro
- **ParamGetPtosVenta()**: devuelve los puntos de venta autorizados para factura electrónica
- **ParamGetActividades()**: devuelve los códigos de actividades, sus descripciones y el orden (si es actividad primaria, secundaria, etc).
- **ParamGetCotizacion(moneda_id, fecha)**: se agrega fecha de cotizacion. ***Nuevo!***
- **ParamGetCondicionIvaReceptor(clase_cmp)**: consultar condiciones de iva por clase de comprobante. ***Nuevo!***

Métodos Utilitarios provistos por este componente

- **AnalizarXml(xml)**: analizar el mensaje xml para poder extraer atributos puntuales (xml puede ser "XmlRequest", "XmlResponse" o cualquier texto xml válido)
- **ObtenerTagXml(tag1, tag2, ...)**: busca en el mensaje xml analizado la etiqueta tag1, luego tag2 y así sucesivamente, devolviendo el contenido (texto) del dato si fue encontrada, o nulo en caso contrario.

Atributos

El componente tiene las siguientes propiedades que pueden ser consultadas o modificadas:

- **Token**: es el código de autorización generado por la AFIP (WSAA)
- **Sign**: es la firma de autorización generado por la AFIP (WSAA)
- **Cuit**: es el número de CUIT del emisor de facturas, formato string sin guiones.
- **AppServerStatus**, **DbServerStatus**, **AuthServerStatus**: estados de los servidores de AFIP (string ?OK? en caso de estar funcionales)
- **XmlRequest**, **XmlResponse**: requerimiento y respuesta XML (para depuración)
- **InstallDir** ***Nuevo!*** : directorio de instalación (ej. `C:\Archivos de Programa\WSAA`)
- **Excepcion**, **Traceback** ***Nuevo!*** : mensaje de error y traza de rastreo (para depuración)
- **LanzarExcepciones** ***Nuevo!*** : establece si se deben emitir errores al lenguaje de programación (habilitado por defecto), o serán controlados por el programa (revisando el atributo Excepcion luego de cada método)
- **Respuesta**, **Obs**, **Reproceso**: valores complementarios que retornan los métodos
- **CAE**, **Vencimiento**: CAE y Fecha de vencimiento autorización
- **CAEA**: código de autorización anticipado (si corresponde)
- **Version**: versión de la interfase (ej. ?1.11?)
- **FechaCbte**: fecha del comprobante (del comprobante recuperado devuelto por GetCmp)
- **ImpTotal, ImpNeto, !ImptoLiq, ImpOpEx, ImpIVA, ImpOpEx, ImpTrib**: importes del comprobante
- **ErrCode**: código de error (si corresponde)
- **ErrMsg**: mensaje de error (si corresponde)
- **Errores**: lista de errores (si corresponde)
- **Eventos**: lista de eventos (si corresponde)
- **Reprocesar**: habilita reproceso automático (habilitado por defecto, ver abajo)

- **`EmisitonTipo`**: tipo de código de autorización (CAE o CAEA)

Errores

La interfase procesa los errores WSFEv1 y los almacena en las propiedades Errores y ErrCode/ErrMsg (por ej, motivo de rechazo al intentar solicitar CAE). AFIP también puede enviar errrores o advertencias por la como Observaciones, propiedad Obs. Se recomienda revisarlos luego de llamar a los Webservices.

Ver ejemplos y especificaciones técnicas del WSFEv1 para más información sobre el uso del servicio web, su formato, condiciones y errores.

Ver Manejo de Excepciones para conocer las tecnicas para controlar las distintas excepciones que pueden ocurrir (ver atributos Excepcion y Traceback, en general por problemas de conectividad) y como proceder y almacenar los mensajes XML para depuración y futuras consultas (atributos XmlRequest y XmlResponse)

Validaciones WSFEv1

AFIP realiza varias validaciones a los datos enviados, muchas agregadas recientemente, por lo que para evitar inconvenientes se recomienda revisar las siguientes validaciones y consideraciones generales:

- *10063: Factura individual, DocTipo: 80, DocNro XXXXXXXXX no se encuentra inscripto en condicion ACTIVA en el impuesto.*: Las facturas A solo pueden ser emitidas a Responsables Inscriptos, consignando el CUIT válido y registrado (tipo_doc=80).
- *10014: Para facturas B (CbteDesde distinto a CbteHasta) el resultado de la operación ImpTotal / (CbteHasta - CbteDesde + 1) debe ser menor a $1000.* y *10015: Para facturas B (CbteDesde distinto a CbteHasta) menor a $1000 el campo DocNro deberá ser cero (0) y el campo DocTipo 99.*: Los lotes de facturas B menores a $1000.- pueden ser emitidos a sólo a Consumidor Final (tipo_doc=99 y nro_doc=0), indicando el importe total del reango de facturas.
- *10043: El campo ImpTotConc (Importe Total del Concepto) para comprobantes tipo C debe ser igual a cero (0).* y *10047: El campo ImpIVA (Importe de IVA) para comprobantes tipo C debe ser igual a cero (0).*, *10071: Para comprobantes tipo C el objeto IVA no debe informarse.*: Para emitir Facturas C, no se informa iva ni imp_tot_conc, ver caso especial Monotributo
- *10048: El campo 'Importe Total' ImpTotal, debe ser igual a la suma de ImpTotConc + ImpNeto + ImpOpEx + ImpTrib + ImpIVA.*: La sumatoria de los campos imp_neto, impto_iva, imp_op_ex, imp_tot_conc imp_trib contra imp_total no debe diferir en más de 1 centavo (o 0.01%).
- *10051: Los importes informados en AlicIVA no se corresponden con los porcentajes.*: La base imponible por la alícuota de IVA no debe diferir en más de 1 centavo con el impuesto liquidado de IVA (para cada tasa, ver AgregarIVA).
- *10020: El campo BaseImp en AlicIVA es obligatorio y debe ser mayor a 0 cero.*: Si el imp_neto es 0, no se debe enviar subtotal de IVA, ni para la tasa 0% (no usar AgregarIVA).
- *10018Si ImpIva? es igual a 0 el objeto Iva y AlicIva son obligatorios. Id iva = 3 (iva 0)*: Si el imp_neto es <>0 y el iva liquidado es 0, se debe enviar código de alicuota 3 (0%). -por ej, caso especial de hacer factura de 1 centavo-. Si no corresponde IVA (exento o no gravado), se debe enviar imp_iva NULO (vbNull, null o similar) y no enviar alícuotas.

Consideraciones especiales:

- Para emitir Facturas B a Responsables No Inscriptos, debe informarse en Tributos el importe correspondiente (ver AgregarTributo)
- Para factura E no corresponde este webservice, por más que sea a Tierra del Fuego. Ver WSFEX.

- imp_tot_conc debe utilizarse para los importes no gravados (por ej., cheques rechazados)
- IVA 0% es distinto a No Gravado (sería imp_tot_conc) o Exento (imp_op_ex). En esos casos no va alicuota de iva, e imp_iva = NULL y imp_neto = 0 (ver validaciones arriba)
- Si se usa IVA 0%, en el subtotal de la alícuota iva_id=3, se debe pasar base_imp = 100 e importe = 0. Además, imp_neto = 100 (gravado, aunque no liquide IVA)

Recordamos que si la AFIP detecta una validación no obligatoria (por ej. no corresponde emitir factura A), devolvera un mensaje (en el atributo WSFEv1.Obs) y el IVA liquidado no puede computarse como crédito fiscal (la factura debe incorporar dichos mensajes).

Dependiendo del caso, los mensajes de validación pueden ser enviados por AFIP tanto en WSFEv1.Obs o en WSFEv1.ErrMsg, se deben revisar ambos campos.

Nota: estas consideraciones pueden variar en cada caso, dependiendo de cada situación particular frente a AFIP. Recomendamos consultar con el contador los aspectos legales y la normativa vigente.

Ejemplo

Ejemplo Solicitud CAE WSFEv1

Autorizar una factura electrónica de mercado interno (versión 1) a modo de ejemplo (ficticia) CAE normal:

```
oject, WSFEv1 As Object

o interface Web Service Autenticación y Autorización
eateObject("WSAA")

Ticket de Requerimiento de Acceso (TRA) para WSFEv1
eateTRA("wsfe")

la ubicacion de los archivos certificado y clave privada
() + "\"
 certificado es el firmado por la AFIP
a: la clave privada usada para crear el certificado
"reingart.crt" ' certificado de prueba
= "reingart.key" ' clave privada de prueba

nensaje firmado (CMS)
gnTRA(tra, Path + Certificado, Path + ClavePrivada)

eb service WSAA para autenticar:
directorio temporal (usar predeterminado)
ttps://wsaahomo.afip.gov.ar/ws/services/LoginCms"  ' usar wsaa.afip.gov.ar en producción
información de servidor intermedio (si corresponde)
ectar(cache, url_wsdl, proxy, wrapper)
nCMS(cms)

enido, se puede usar el mismo token y sign por 24 horas
do se puede cambiar)

o interface Web Service de Factura Electrónica de Mercado Interno
CreateObject("WSFEv1")
SFEv1.version

en y sing de autorización (pasos previos)
= WSAA.Token
 WSAA.Sign
```

```
isor (debe estar registrado en la AFIP)
"20267565393"

 Servicio Web de Facturación
directorio temporal (usar predeterminado)
ttps://wswhomo.afip.gov.ar/wsfev1/service.asmx?WSDL" ' usar servicios1 para producción
información de servidor intermedio (si corresponde)
onectar(cache, url_wsdl, proxy) ' homologación

servicio nulo, para obtener el estado del servidor (opcional)

appserver status", WSFEv1.AppServerStatus
dbserver status", WSFEv1.DbServerStatus
authserver status", WSFEv1.AuthServerStatus

los valores de la factura a autorizar:

001
FEv1.CompUltimoAutorizado(tipo_cbte, punto_vta) + 1
(Date, "yyyymmdd")

  nro_doc = "20267565393"
bte_nro: cbt_hasta = cbte_nro
122.00": imp_tot_conc = "0.00": imp_neto = "100.00"
 00": imp_trib = "1.00": imp_op_ex = "0.00"
fecha: fecha_venc_pago = fecha
período del servicio facturado (solo si concepto = 1?)
sde = fecha: fecha_serv_hasta = fecha
PES": moneda_ctz = "1.000"

rearFactura(concepto, tipo_doc, nro_doc, tipo_cbte, punto_vta, _
  cbt_hasta, imp_total, imp_tot_conc, imp_neto, _
mp_trib, imp_op_ex, fecha_cbte, fecha_venc_pago, _
v_desde, fecha_serv_hasta, _
  moneda_ctz)

comprobantes asociados:
' solo nc/nd

 2
1
v1.AgregarCmpAsoc(tipo, pto_vta, nro, cuit, fecha)


estos varios

sto Municipal Matanza'"
00.00"

00"
gregarTributo(id, Desc, base_imp, alic, importe)

s de IVA

00.00"
 00"
gregarIva(id, base_imp, importe)

mpradores RG 4109-E bienes muebles registrables (%)
' solo productos y si hay más de un comprador:
v1.AgregarComprador(80, "30500010912", 99.99)
```

```
v1.AgregarComprador(80, "30999032083", 0.01)


E:
CAESolicitar()

Resultado", WSFEv1.Resultado
CAE", WSFEv1.cae

Numero de comprobante:", WSFEv1.CbteNro

ido y respuesta XML para depuración (errores de formato)
SFEv1.XmlRequest
SFEv1.XmlResponse

tado:" & WSFEv1.Resultado & " CAE: " & cae & " Venc: " & WSFEv1.Vencimiento & " Obs: " & WSFEv1.obs, vbInformat

eventos (mantenimiento programados y otros mensajes de la AFIP)
o In WSFEv1.eventos:
ento, vbInformation, "Evento"
```

Ejemplo Consultar CAE WSFEv1

Recuperar una factura electrónica de mercado interno (versión 1) a modo de ejemplo (ficticia, continua el ejemplo anterior) CAE normal:

```
' Buscar la factura
cae2 = WSFEv1.CompConsultar(tipo_cbte, punto_vta, cbte_nro)

Debug.Print "Fecha Comprobante:", WSFEv1.FechaCbte
Debug.Print "Fecha Vencimiento CAE", WSFEv1.Vencimiento
Debug.Print "Importe Total:", WSFEv1.ImpTotal

' comparar con los datos del ejemplo anterior:
If cae <> cae2 Then
    MsgBox "El CAE de la factura no concuerdan con el recuperado en la AFIP!: " & cae & " vs " & cae2
Else
    MsgBox "El CAE de la factura concuerdan con el recuperado de la AFIP"
End If

' obtener datos del encabezado (a partir de actualización 1.17a)
cae = wsfev1.ObtenerCampoFactura("cae")
tipo_doc = wsfev1.ObtenerCampoFactura("tipo_doc")
nro_doc = wsfev1.ObtenerCampoFactura("nro_doc")
imp_total = wsfev1.ObtenerCampoFactura("imp_total")
' obtener primer alicuota de IVA
imp_iva1 = wsfev1.ObtenerCampoFactura("iva", 0, "importe")
' obtener primer tributo
imp_trib1 = wsfev1.ObtenerCampoFactura("tributos", 0, "importe")
' obtener primer opcional
valor_opcional1 = wsfev1.ObtenerCampoFactura("opcionales", 0, "valor")
' obtener primer código de observacion de AFIP
obs_code1 = wsfev1.ObtenerCampoFactura("obs", 0, "code")
' pruebo obtener el segundo mensaje de observacion inexistente
obs_code2 = wsfev1.ObtenerCampoFactura("obs", 1, "msg")
debug.Print wsfev1.Excepcion  ' "El campo 1 solicitado no existe"
```

Ejemplo Consulta Puntos de Venta Autorizados WSFEv1

Ejemplo para obtener las tablas de parametros, en este caso puntos de venta habilitados (útil en producción para verificar el acceso al webservice):

```
' recupero tabla de parámetros de punto de venta habilitado ("3|EmisionTipo:CAE|Bloqueado:N|FchBaja:NULL"
For Each x In WSFEv1.ParamGetPtosVenta()
    Debug.Print x
Next
```

Ejemplo Solicitud CAEA y registro informativo WSFEv1

Autorizar una factura de mercado interno (version 1) a modo de ejemplo (ficticia) CAE Anticipado:

```
' PASO 1: Solicito CAE Anticipado para el período
' NOTA: solicitar por única vez para un determinado período
' consultar si se ha solicitado previamente

periodo = "201102"  ' Año y mes
orden = "2"         ' Segunda Quincena

' consulto CAEA ya solicitado
CAEA = WSFEv1.CAEAConsultar(periodo, orden)
If CAEA = "" Then
    ' solicito nuevo CAEA
    CAEA = WSFEv1.CAEASolicitar(periodo, orden)
End If

MsgBox "Periodo: " & periodo & " Orden " & orden & vbCrLf & "CAEA: " & CAEA & vbCrLf & _
        "Obs:" & WSFEv1.Obs & vbCrLf & _
        "Errores:" & WSFEv1.ErrMsg

    ' Si no tengo CAEA, termino
    If CAEA = "" Then End

' PASO 2: Establezco los valores de la factura a informar:
tipo_cbte = 6
punto_vta = 4005
cbte_nro = WSFEv1.CompUltimoAutorizado(tipo_cbte, punto_vta)
If cbte_nro = "" Then
    cbte_nro = 0                ' no hay comprobantes emitidos
Else
    cbte_nro = CLng(cbte_nro)   ' convertir a entero largo
End If
fecha = Format(Date, "yyyymmdd")
concepto = 1
tipo_doc = 80: nro_doc = "33693450239"
cbte_nro = cbte_nro + 1
cbt_desde = cbte_nro: cbt_hasta = cbte_nro
imp_total = "122.00": imp_tot_conc = "0.00": imp_neto = "100.00"
imp_iva = "21.00": imp_trib = "1.00": imp_op_ex = "0.00"
fecha_cbte = fecha: fecha_venc_pago = ""
' Fechas del período del servicio facturado (solo si concepto = 1?)
fecha_serv_desde = "": fecha_serv_hasta = ""
moneda_id = "PES": moneda_ctz = "1.000"

' creo una factura (con CAEA)
ok = WSFEv1.CrearFactura(concepto, tipo_doc, nro_doc, tipo_cbte, punto_vta, _
    cbt_desde, cbt_hasta, imp_total, imp_tot_conc, imp_neto, _
    imp_iva, imp_trib, imp_op_ex, fecha_cbte, fecha_venc_pago, _
```

```
    fecha_serv_desde, fecha_serv_hasta, _
    moneda_id, moneda_ctz, CAEA, fecha_hs_gen)

' Agrego los comprobantes asociados:
If False Then ' solo nc/nd
    tipo = 19
    pto_vta = 2
    nro = 1234
    ok = WSFEv1.AgregarCmpAsoc(tipo, pto_vta, nro, cuit, fecha)
End If

' Agrego impuestos varios
id = 99
Desc = "Impuesto Municipal Matanza'"
base_imp = "100.00"
alic = "1.00"
importe = "1.00"
ok = WSFEv1.AgregarTributo(id, Desc, base_imp, alic, importe)

' Agrego tasas de IVA
id = 5 ' 21%
base_imp = "100.00"
importe = "21.00"
ok = WSFEv1.AgregarIva(id, base_imp, importe)

' Informo comprobante emitido con CAE anticipado:
cae = WSFEv1.CAEARegInformativo()

Debug.Print "Resultado", WSFEv1.Resultado
```

Descripción de la obtención de CAE / CAEA

Como primer paso, se debe crear una factura (utilizada internamente por la interfase para contener los valores a autorizar), llamando al método CrearFactura o completar el registro de encabezado (ver archivo de intercambio), con los siguiente parámetros:

- tipo_doc, nro_doc: Tipo (80 CUIT, 96 DNI, etc. según tabla de parámetros de AFIP) y número de Documento del cliente (receptor de la factura). Usar tipo_doc=99 y nro_doc=0 para consumidores finales (Factura B < $1000)
- tipo_cbte: Tipo de comprobante (según tabla de parámetros AFIP)
- punto_vta: Nº de punto de venta (debe estar autorizado para WSFE)
- cbt_desde, cbt_hasta: Nº de comprobante (desde, hasta, generalmente el mismo número si es una factura individual, o un rango si son Factura B < $ 1000 a Consumidor Final -en este último caso se obtiene un único CAE para el "lote"-)
- fecha_cbte: Fecha del comprobante (no puede ser mayor o menor a 10 días)
- concepto: tipo de factura (1: productos, 2: servicios, etc.)
- imp_total: Importe total de la factura (debe ser igual a la suma de imp_tot_conc + imp_op_ex + imp_neto + imp_iva + imp_trib)
- imp_tot_conc: Importe total de conceptos no gravados por el IVA
- imp_neto: Importe neto (gravado por el IVA) de la factura (igual a la suma de base_imp para todas las alicuotas)
- imp_iva: Importe del IVA liquidado (igual a la suma de importe_iva para todas las alícutoas). Enviar null o similar si no corresponde y no se envían subtotales por alícuota.
- imp_trib: Importe de otros tributos (incluyendo percepciones de IVA, retenciones, IVA no inscripto, etc.)

- imp_op_ex: Importe de operaciones exentas
- moneda_id: Moneda de la factura (según tabla de parámetros, actualmente solo PES)
- moneda_ctz: Cotización de la moneda de la factura (actualmente solo 1.00)
- fecha_vto_pago: Fecha de vencimiento de pago (si es de servicios)
- fecha_serv_desde, fecha_serv_hasta: fecha del servicios
- caea: codigo de autorización anticipado (si corresponde)

Luego, por cada alicuota de IVA (excepto para no gravado y exento), se debe llamar al método AgregarIva o completar el registro respectivo, con los siguientes parámetros:

- iva_id: código Alícuota de IVA (según tabla de parámetros AFIP)
- base_imp: base imponible (importe)
- importe_iva: importe liquidado (base_imp por alicuota)

De existir otros tributos (percepciones/retenciones o similares), se debe llamar al método AgregarTributo o completar el registro respectivo, con los siguientes parámetros:

- tributo_id: código tipo de impuesto (según tabla de parámetros AFIP)
- desc: descripción del tributo (por ej. "Impuesto Municipal Matanza")
- base_imp: base imponible (importe)
- alic: alicuota (porcentaje)
- importe: importe liquidado

También se puede llamar al método AgregarCmpAsoc o completar el registro respectivo, para detallar los comprobantes asociados a una nota de crédito, con los siguientes parámetros:

- tipo: Código de tipo de comprobante (según tabla de parámetros AFIP)
- pto_vta: Punto de venta
- nro: Numero de comprobante

Como último paso, se debe llamar al método CAESolicitar o CAEARegInformativo dependiendo de la modalidad (o ejecutar el programa RECE1 si se esta utilizando la herramienta por linea de comando).

Si no hubiere inconvenientes, la llamada debe devolver el CAE y se establece el atributo WSFEv1.Resultado = "A" (Aceptado) y WSFEv1.Vencimiento del CAE (fch_venc_cae). Sino, devuelve cae = "": La interfase no pudo procesar la respuesta del WebService, o bien la conexión a internet esta caída, las direcciones de los servidores son incorrectas, o el servicio web rechazo los datos de la factura a generar. Se establece el atributo WSFEv1.Resultado = "R" (Rechazado) y WSFEv1.Obs (o WSFEv1.ErrMsg) con los diversos motivos de rechazo proporcionados por el webservice (ver WSFE.Motivo).

Aún devolviendo el CAE y con el resultado Aceptado, puede haber un motivo (ej "10017: Factura individual, DocTipo: 80, DocNro 33693450239 no se encuentra en condicion activa en los padrones de AFIP."), que funcionan como advertencia para revisar la base de datos y corregir posibles irregularidades.

Como requisito previo para la modalidad CAE Anticipado, se debe llamar a CAEASolicitar o CAEAConsultar con los datos del periodo y orden (antes de comenzar la quincena), para obtener el único CAEA para los comprobantes del período.

Advertencia: Este servicio web no tiene reproceso (a diferencia de WSFE). En caso de falla o pérdida de comunicación, usar el método para recuperar un comprobante emitido (CompConsultar) o usar la rutina de Reproceso Automático incluida en este componente.

Para más información ver especificaciones técnicas del WSFEV1 en el sitio de la AFIP:

- Minisitio Factura Electrónica AFIP: información general
- Manual para el desarrollador.pdf: Información importante sobre la operatoria

Reproceso Automático

Importante: En el caso de que la conexión se interrumpa al llamar a `CAESolicitar`, luego de que la AFIP registre la factura y antes de que la información del CAE llegue al equipo, se debe llamada a `CompConsultar` con los parámetros de la factura (tipo_cbte, punto_vta, cbte_nro) para recuperar el CAE generado anteriormente (ver ejemplo pseudocódigo y por RECE1 /get).

Este procedimiento puede ser realizado automáticamente por la interfaz (de manera similar a como realizaba AFIP originalmente en WSFEv0), habilitando el atributo `Reprocesar` = `True` (valor predeterminado), con lo que la interfaz automaticamente:

- Solicita un nuevo CAE en caso de que no exista en los registros de la AFIP
- Consulta y obtiene el CAE anterior en caso de que ya exista en los registros de AFIP

En ese caso, si se da un reprocesamiento automático, como se mencionó en el primer parrafo, WSFE.Reproceso se establece a ?S? para indicar que el CAE se ha recuperado.

Si bien no se utiliza un ID por lo que se minimizan los errores al reprocesar los comprobantes, igualmente se recomienda revisar el atributo de `Reproceso`, ya que un incorrecto uso puede generar inconvenientes al registrar los comprobantes en AFIP. Para ello la interfaz cuenta con los atributos `CbteNro`, `FechaCbte`, `ImpTotal`, `ImpNeto`, `ImptoLiq`, `ImpOpEx`, `ImpIVA`, `ImpOpEx`, `ImpTrib` o `XmlRequest` y `XmlResponse`, que deben ser contrastados contra la información enviada, en el caso de interrupciones o inconvenientes en la comunicación (a partir de la versión 1.07d, si Reproceso = ?S? la interfaz verificó automáticamente todos los datos enviados con los registrados en la AFIP y estos concuerdan)

Se recomienda almacenar los datos en un medio durable para poder enviar la misma factura que fuera interrumpida, salvo que se tenga certeza de que la misma no ha sido procesada por AFIP. Especialmente se debería guardar el número de factura antes de solicitar CAE, y no utilizar `/ult` o el método `CompUltimoAutorizado` para consultar el último número de comprobante emitido, ya que ante cualquier interrupción se no se podría recuperar los datos registrados en AFIP (o se podrían generar duplicados).

En síntesis, al re-enviar los datos de un comprobante que por algún motivo no ha devuelto CAE (cuando la comunicación se ha interrumpido y no ha sido aceptado o rechazado formalmente con `Resultado=A` o `R`), se recomienda revisar el reproceso y los datos registrados (devueltos por la interfaz), o en su defecto, deshabilitar `Reprocesar` con `False` e implementar un procedimiento propio de recuperación ante fallas.

Si el reproceso está habilitado, no es necesario modificar la aplicación en caso de fallas, simplemente reintentar la solicitud de CAE o información de CAEA, y la interfaz realizará los métodos pertinentes según se describió anteriormente.

En caso de que no haya sido necesario reprocesar el requerimiento, el atributo `Reproceso` quedará vacio. En caso de que haya sido necesario reprocesar pero no ha sido exitoso (ocurrió un error o validación que no permitió recuperar el CAE o los datos enviados no concuerdan con los registrados en AFIP), `Reproceso` será `N` y se informará el error original.

Lo expuesto anteriormente aplica también a CAEA -anticipado- al informar los comprobantes emitidos (`CAEARegInformativo`).

En ambas ocasiones se puede consultar el atributo `EmisionTipo` que contiene `CAE` o `CAEA` dependiendo del caso.

Nota: El Reproceso automático está incluido internamente en `CAESolicitar` y `CAEARegInformativo`, por lo que está disponible tanto en la biblioteca Python, Interfaz COM, la herramienta por línea de comando RECE1.EXE y al aplicativo PyRece.

Advertencia: El Reproceso automático es una característica adicional propia de la interface, por lo tanto puede sufrir variaciones o mejoras, se recuerda que la información expuesta en esta documentación aplica a la última versión de los instaladores.

Factura C Monotributo / Exento

Según RG 3067/2011, se adiciona la posibilidad de autorizar comprobantes clase C (monotributistas) a WSFEv1 (RG2485).

Según RG 3749/2015, se incluye a los sujetos exentos en el impuesto al valor agregado. Para poder emitir comprobantes clase C, además de seguir las pautas señaladas en esta sección, el punto de venta debe estar dado de alta como "Factura Electrónica - Exento en IVA - Web Services".

Los siguientes códigos de comprobantes (`tipo_cbte`) pueden ser utilizados en este caso:

- 11: Factura C
- 12: Nota de Débito C
- 13: Nota de Crédito C
- 15: Recibo C

Los requisitos fundamentales para poder validar un comprobante informado son las los siguientes:

1. Ingresar siempre `imp_tot_conc` = 0
2. Ingresar siempre `imp_op_ex` = 0
3. Ingresar siempre `imp_iva` = 0
4. Informar en `imp_neto` el importe subtotal de la factura
5. No informar subtotales de IVA (`AgregarIVA`)

Código de Ejemplo:

```
' Establezco los valores de la factura a autorizar:
tipo_cbte = 11
punto_vta = 4002
cbte_nro = 1
fecha = Format(Date, "yyyymmdd")
concepto = 1
tipo_doc = 80: nro_doc = "33693450239"
cbte_nro = cbte_nro + 1
cbt_desde = cbte_nro: cbt_hasta = cbte_nro
imp_total = "100.00": imp_tot_conc = "0.00": imp_neto = "100.00"
imp_iva = "0.00": imp_trib = "0.00": imp_op_ex = "0.00"
fecha_cbte = fecha: fecha_venc_pago = ""
' Fechas del período del servicio facturado (solo si concepto = 2 o 3)
```

```
fecha_serv_desde = "": fecha_serv_hasta = ""
moneda_id = "PES": moneda_ctz = "1.000"

ok = WSFEv1.CrearFactura(concepto, tipo_doc, nro_doc, tipo_cbte, punto_vta, _
      cbt_desde, cbt_hasta, imp_total, imp_tot_conc, imp_neto, _
      imp_iva, imp_trib, imp_op_ex, fecha_cbte, fecha_venc_pago, _
      fecha_serv_desde, fecha_serv_hasta, _
      moneda_id, moneda_ctz)

' Solicito CAE:
CAE = WSFEv1.CAESolicitar()
```

Documentación Oficial: Facturación Electrónica RG 2485 ? Proyecto FE v2.5 - Manual para el desarrollador (Revisión correspondiente al 01 de Abril de 2015)

Datos Opcionales AFIP WSFEv1

El componente permite enviar los datos opcionales requeridos por las diversas resoluciones de AFIP según la tabla de parámetros. *Disponible desde actualización 1.15b*

Datos opcionales para proyectos promovidos:

```
WSFEv1.AgregarOpcional(2, "1234")           ' identificador del proyecto
```

Datos opcionales para RG Bienes Usados 3411 (del vendedor):

```
WSFEv1.AgregarOpcional(91, "Juan Perez")    ' Nombre y Apellido
WSFEv1.AgregarOpcional(92, "200")           ' Nacionalidad (código de país)
WSFEv1.AgregarOpcional(93, "Balcarce 50")   ' Domicilio
```

Datos opcionales para RG 3668 Impuesto al Valor Agregado - Art.12 (*"presunción de no vinculación con la actividad gravada"*, FORM 8001):

```
            ' IVA Excepciones (01: Locador/Prestador, 02: Conferencias, 03: RG 74, 04: Bienes de cambio, 05: M
)           ' Firmante Doc Tipo (80: CUIT, 96: DNI, etc.)
67565393")  ' Firmante Doc Nro
            ' Carácter del Firmante (01: Titular, 02: Director/Presidente, 03: Apoderado, 04: Empleado)
```

Datos opcionales para RG 3749 (*"RG 3.368 Establecimientos de educación pública de gestión privada"*):

```
WSFEv1.AgregarOpcional(10, "1")              ' Actividad Comprendida
WSFEv1.AgregarOpcional(1011, "80")           ' Tipo de Documento (titular del pago)
WSFEv1.AgregarOpcional(1012, "20267565393")  ' Número de Documento (titular del pago)
```

Datos opcionales para RG4520:

```
WSFEv1.AgregarOpcional(5, "01")               ' Actividad
```

Datos opcionales para FEv2.13 (*"RG 4.367 Factura de Crédito Electrónica MiPyMEs (FCE)"*):

```
WSFEv1.AgregarOpcional(2101, "2850590940090418135201")  ' CBU
WSFEv1.AgregarOpcional(2102, "pyafipws")                ' alias
WSFEv1.AgregarOpcional(27, "SCA")                       ' tipo de transmisión (desde el 01/04/2021)
if tipo_cbte in (203, 208, 213):
```

```
    WSFEv1.AgregarOpcional(22, "S")                     ' Anulación
```

Obtención de Atributos Avanzados WSFEv1

En versiones recientes (1.12a o superior), si no hubo excepción es posible revisar y obtener datos avanzados de la respuesta (útiles para depuración y solución de errores) con los métodos `AnalizarXml` y `ObtenerTagXml`.

Ejemplo para recuperar datos puntuales de un comprobante emitido (moneda, cotización, CUIT, etc.):

```
cae2 = WSFEv1.CompConsultar(tipo_cbte, punto_vta, cbte_nro)
ok = WSFEv1.AnalizarXml("XmlResponse")
If ok Then
    ' datos del encabezado:
    Debug.Print "CAE:", WSFEv1.ObtenerTagXml("CodAutorizacion"), WSFEv1.CAE
    Debug.Print "CbteFch:", WSFEv1.ObtenerTagXml("CbteFch"), WSFEv1.FechaCbte
    Debug.Print "Moneda:", WSFEv1.ObtenerTagXml("MonId")
    Debug.Print "Cotizacion:", WSFEv1.ObtenerTagXml("MonCotiz")
    Debug.Print "DocTipo:", WSFEv1.ObtenerTagXml("DocTipo")
    Debug.Print "DocNro:", WSFEv1.ObtenerTagXml("DocNro")

    ' ejemplos con arreglos (primer elemento = 0, segundo = 1, etc.):
    Debug.Print "Primer IVA (alci id):", WSFEv1.ObtenerTagXml("Iva", "AlicIva", 0, "Id")
    Debug.Print "Primer IVA (importe):", WSFEv1.ObtenerTagXml("Iva", "AlicIva", 0, "Importe")
    Debug.Print "Segundo IVA (alic id):", WSFEv1.ObtenerTagXml("Iva", "AlicIva", 1, "Id")
    Debug.Print "Segundo IVA (importe):", WSFEv1.ObtenerTagXml("Iva", "AlicIva", 1, "Importe")
    Debug.Print "Primer Tributo (ds):", WSFEv1.ObtenerTagXml("Tributos", "Tributo", 0, "Desc")
    Debug.Print "Primer Tributo (importe):", WSFEv1.ObtenerTagXml("Tributos", "Tributo", 0, "Importe")
    Debug.Print "Segundo Tributo (ds):", WSFEv1.ObtenerTagXml("Tributos", "Tributo", 1, "Desc")
    Debug.Print "Segundo Tributo (importe):", WSFEv1.ObtenerTagXml("Tributos", "Tributo", 1, "Importe")
    Debug.Print "Tercer Tributo (ds):", WSFEv1.ObtenerTagXml("Tributos", "Tributo", 2, "Desc")
    Debug.Print "Tercer Tributo (importe):", WSFEv1.ObtenerTagXml("Tributos", "Tributo", 2, "Importe")
End If
```

Servicio Web de Factura Electrónica Mercado Interno Programa Matrix (WSMTXCA)

EL WSMTXCA es un Servicio Web de la AFIP para Facturas Electrónicas de Mercado Interno, correspondiente al Artículo 4 Opción A (con detalle de factura) de la Resolución General 2904/2010,.

Para poder acceder, se debe obtener un Ticket de Acceso (TA) previamente (para utlizar el Token y Sign y enviarlos a los métodos de facturación electrónica).

Esta interfase recibe los valores y crea los mensaje en XML, lo envía y analiza la respuesta XML, por lo que no es necesario modificar XML. Se recomienda pasar los valores en formato string, números separados por punto y fechas YYYY-MM-DD.

La operatoria es bastante similar al método de autorización del WSFE (pero **SIN** un ID secuencial ni reproceso), teniendo en cuenta esta mayor complejidad por tener que informar el detalle de los artículos, los impuestos (IVA) y tributos (similar a WSFEX y WSBFE).

A su vez, el WSMTXCA devuelve mensajes de eventos (mantenimiento programado, advertencias, etc.), los que deben ser capturados e informados al usuario.

A diferencia del WSFE, este nuevo servicio funciona con tablas dinámicas de parámetros para los códigos de comprobante, moneda, alícuotas de iva, tributos, unidades de medida. Estas tablas pueden sufrir modificaciones realizadas por la AFIP, con altas y bajas lógicas, por lo que tienen una fecha de vigencia (desde, hasta) y se proveen métodos para consultarlas por el mismo servicio web (a diferencia del WSFE, que las tablas eran documentadas estáticamente en el sitio web).

Para más información ver:

http://www.sistemasagiles.com.ar/trac/wiki/FacturaElectronicaMTXCAService

La programación de la interfase WSMTXCA es similar a WSFE, WSFEX, WSBFE y WSFEv1, se recomienda revisarlos.

Objeto

- El objeto COM se crea invocando a CreateObject("WSMTXCA")

Métodos

Métodos principales:

- **`Conectar(cache, wsdl, proxy)`**: realiza la conexión a los servidores de la AFIP (primer paso esencial). Si no se especifica url del wsdl, se utiliza servidores de homologación. El parámetro cache es un directorio donde se almacenan internamente la descripción del servicio (archivo WSDL) para mayor optimización. Proxy es un string con la información del servidor intermedio: "usuario:clave@servidor:puerto"
- **`Dummy()`**: servicio de prueba para obtener el estado de los servidores de la AFIP.
- **`CrearFactura(concepto, tipo_doc, nro_doc, tipo_cbte, punto_vta, cbt_desde, cbt_hasta, imp_total, imp_tot_conc, imp_neto, imp_subtotal, imp_trib, imp_op_ex, fecha_cbte, fecha_venc_pago, fecha_serv_desde, fecha_serv_hasta, moneda_id, moneda_ctz, obs, cancela_misma_moneda_ext, condicion_iva_receptor_id)`**: crea internamente una factura para luego poder autorizarla, recibe los datos de la factura a emitir. Ver ejemplo para el detalle de los parámetros.
- **`EstablecerCampoFactura(campo, valor)`**: establece individualmente el valor de un campo del encabezado de la factura, devuelve True si el campo pertenece al encabezado y se ha actualizado correctamente (ver ejemplo)
- **`AgregarIva(id, base_imp, importe)`**: agrega internamente un subtotal de IVA a una factura para luego poder autorizarla, recibe los datos del tipo de alícuota, base imponible e importe. Ver ejemplo para el detalle de los parámetros.
- **`AgregarTributo(id, Desc, base_imp, alic, importe)`**: agrega internamente un subtotal de tributo a una factura para luego poder autorizarla, recibe los datos del impuesto nacional, provincial o municipal (descripción), base imponible e importe. Ver ejemplo para el detalle de los parámetros.
- **`AgregarCmpAsoc(tipo_cbte_asoc, punto_vta_asoc, cbte_nro_asoc, cbte_cuit, fecha)`**: agrega internamente un comprobante asociado a una factura para luego poder autorizarla, recibe tipo de comprobante, punto de venta y número. Ver ejemplo para el detalle de los parámetros. El campo `cuit` fué agregado según Release v0.5 15-03-2017, *disponible desde actualización 1.13a*. El campo fecha fue agregado en Realse v0.10 01/07/2019,

- **`AgregarOpcional(opcional_id, valor)`**: agrega internamente un dato opcional a una factura para luego poder autorizarla, recibe el id del tipo de dato opcional y valor
- **`AgregarItem(u_mtx, cod_mtx, codigo, ds, qty, umed, precio, bonif, cod_iva, imp_iva, imp_subtotal)`**: agrega internamente un item (linea de factura) a una factura para luego poder autorizarla, recibe los datos del item a factura a emitir. Ver ejemplo para el detalle de los parámetros.
- **`EstablecerCampoItem(campo, valor)`**: establece individualmente el valor de un campo del detalle de la factura (último item agregado), devuelve True si el campo pertenece al encabezado y se ha actualizado correctamente (ver ejemplo)
- **`AutorizarComprobante()`**: autoriza la emisión de factura electrónica, devuelve el Código de Autorización Electrónico (CAE). Ver ejemplo.
- **`AutorizarAjusteIVA()`**: autoriza un ajuste de IVA (notas de crédito / débito). Se debe informar el código 7790001001139 e informar solo el importe de IVA ajustado.
- **`SolicitarCAEA(periodo, orden)`**: permite obtener un CAEA y su respectivo período de vigencia (fecha de validez desde y fecha de validez hasta)
- **`ConsultarCAEA(periodo, orden, caea)`**: permite consultar la información correspondiente a un CAEA previamente otorgado (recibe período/orden o CAEA)
- **`InformarComprobanteCAEA()`**: envía la información del comprobante emitido y asociado a un CAEA (ver `AutorizarComprobante`).
- **`InformarAjusteIVACAEA()`**: envía la información del comprobante de ajuste de IVA emitidos para un CAEA (ver `AutorizarAjusteIVA`)
- **`AgregarPeriodoComprobantesAsociados(fecha_desde, fecha_hasta)`**: agrega internamente un periodo asociado a un comprobante N/C o N/D para luego poder autorizarla, recibe fecha desde y hasta. *Disponible desde actualización 1.25a 20/06/20*
- **`AgregarActividad(actividad_id)`**: agrega internamente las actividades asociadas a un comprobante. Disponible desde actualización v 1.16a 11/12/22 ***Nuevo!***

Métodos secundarios:

- **`ConsultarComprobante(tipo_cbte, punto_vta, cbte_nro)`**: recupera los datos de una factura autorizada, recibe tipo de comprobante, punto de venta y número de comprobante original, y devuelve el Código de Autorización Electrónico (CAE) obtenido en su momento. A su vez, establece los datos de la factura (Cae, FechaCbte, ImpTotal, ImpNeto). Ver ejemplo para el detalle de los parámetros y valores devueltos.
- **`CompUltimoAutorizado(tipo_cbte, punto_vta)`**: recupera el último número de factura autorizada, recibe tipo de comprobante y punto de venta. Ver WSFE.RecuperaLastCMP
- **`ConsultarMonedas()`**, **`ConsultarTiposComprobante()`**, **`ConsultarTiposDocumento()`**, **`ConsultarAlicuotasIVA()`**, **`ConsultarCondicionesIVA()`**, **`ConsultarUnidadesMedida()`**, **`ConsultarTiposTributo()`**: recupera valores referenciales de códigos de las tablas de parámetros, devuelve una lista de strings con el id/código, descripción del parámetro y vigencia -si corresponde- (ver ejemplos). Más información en Tablas de Parámetros
- **`ConsultarCotizacionMoneda(moneda_id)`**: devuelve cotización y fecha de la moneda indicada como parámetro
- **`ConsultarPuntosVentaCAE()`**: permite consultar los puntos de venta habilitados para CAE en este WS, devuelve una lista (array de strings) con los datos con numero_punto_venta, bloqueado, fecha_baja
- **`ConsultarPuntosVentaCAEA()`**: permite consultar los puntos de venta habilitados para CAEA en este WS, devuelve una lista (array de strings) con los datos con numero_punto_venta, bloqueado, fecha_baja

- **`ConsultarPtosVtaCAEANoInformados(caea)`**: permite consultar que puntos de venta aún no fueron informados para un CAEA determinado, devuelve una lista (array de strings) con los datos del numero_punto_venta
- **`ConsultarActividadesVigentes()`**: permite consultar las actividades vigentes para el contribuyente, devuelve una lista (array de strings) con las actividades vigentes en las bases del organismo. Disponible desde v 1.16a 11/12/2022 ***Nuevo!***

Atributos

- **`Token`**: es el código de autorización generado por la AFIP (WSAA)
- **`Sign`**: es la firma de autorización generado por la AFIP (WSAA)
- **`Cuit`**: es el número de CUIT del emisor de facturas, formato string sin guiones.
- **`AppServerStatus`**, **`DbServerStatus`**, **`AuthServerStatus`**: estados de los servidores de AFIP (string ?OK? en caso de estar funcionales)
- **`XmlRequest`**, **`XmlResponse`**: requerimiento y respuesta XML (para depuración)
- **`InstallDir`** ***Nuevo!*** : directorio de instalación (ej. `C:\Archivos de Programa\WSAA`)
- **`Excepcion`**, **`Traceback`** ***Nuevo!*** : mensaje de error y traza de rastreo (para depuración)
- **`Respuesta`**, **`Obs`**, **`Reproceso`**: valores complementarios que retornan los métodos
- **`CAE`**, Vencimiento: CAE y Fecha de vencimiento autorización
- **`Version`**: versión de la interfase (ej. ?1.11?)
- **`FechaCbte`**: fecha del comprobante (del comprobante recuperado devuelto por GetCmp)
- **`ImpTotal`**: importes del comprobante
- **`ErrCode`**: código de error (si corresponde)
- **`ErrMsg`**: mensaje de error (si corresponde)
- **`Errores`**: lista de errores (si corresponde)
- **`Eventos`**: lista de eventos (si corresponde)

Errores

La interfase procesa los errores WSMTXCA y los almacena en las proipedades !Errores y ErrCode/ErrMsg. Ver especificaciones técnicas del WSMTXCA para más información sobre el uso del servicio web, su formato, condiciones y errores.

Ejemplo

Autorizar una factura de mercado interno (version 1) a modo de ejemplo (ficticia) CAE normal:

```
im WSAA As Object, WSMTXCA As Object

n Error GoTo ManejoError

 Crear objeto interface Web Service Autenticación y Autorización
et WSAA = CreateObject("WSAA")

 Generar un Ticket de Requerimiento de Acceso (TRA) para WSMTXCA
ra = WSAA.CreateTRA("wsmtxca")
ebug.Print tra

 Especificar la ubicacion de los archivos certificado y clave privada
ath = CurDir() + "\"
 Certificado: certificado es el firmado por la AFIP
 ClavePrivada: la clave privada usada para crear el certificado
ertificado = "..\..\reingart.crt" ' certificado de prueba
```

```
lavePrivada = "..\..\reingart.key" ' clave privada de prueba

 Generar el mensaje firmado (CMS)
ms = WSAA.SignTRA(tra, Path + Certificado, Path + ClavePrivada)
ebug.Print cms

 Llamar al web service para autenticar:
a = WSAA.CallWSAA(cms, "https://wsaahomo.afip.gov.ar/ws/services/LoginCms") ' Homologación (cambiar para produc

 Imprimir el ticket de acceso, ToKen y Sign de autorización
ebug.Print ta
ebug.Print "Token:", WSAA.Token
ebug.Print "Sign:", WSAA.Sign

 Una vez obtenido, se puede usar el mismo token y sign por 24 horas
 (este período se puede cambiar)

 Crear objeto interface Web Service de Factura Electrónica de Mercado Interno
et WSMTXCA = CreateObject("WSMTXCA")
ebug.Print WSMTXCA.version

 Setear tocken y sing de autorización (pasos previos)
SMTXCA.Token = WSAA.Token
SMTXCA.Sign = WSAA.Sign

 CUIT del emisor (debe estar registrado en la AFIP)
SMTXCA.Cuit = "20267565393"

 Conectar al Servicio Web de Facturación
SDL = "" '"https://serviciosjava.afip.gov.ar/wsmtxca/services/MTXCAService?wsdl"
k = WSMTXCA.Conectar("", WSDL)   ' producción

 Llamo a un servicio nulo, para obtener el estado del servidor (opcional)
SMTXCA.Dummy
ebug.Print "appserver status", WSMTXCA.AppServerStatus
ebug.Print "dbserver status", WSMTXCA.DbServerStatus
ebug.Print "authserver status", WSMTXCA.AuthServerStatus

 Establezco los valores de la factura a autorizar:
ipo_cbte = 1
unto_vta = 4000
bte_nro = WSMTXCA.CompUltimoAutorizado(tipo_cbte, punto_vta)
echa = Format(Date, "yyyy-mm-dd")
oncepto = 3
ipo_doc = 80: nro_doc = "30000000007"
bte_nro = CLng(cbte_nro) + 1
bt_desde = cbte_nro: cbt_hasta = cbte_nro
mp_total = "122.00": imp_tot_conc = "0.00": imp_neto = "100.00"
mp_trib = "1.00": imp_op_ex = "0.00": imp_subtotal = "100.00"
echa_cbte = fecha: fecha_venc_pago = fecha
 Fechas del período del servicio facturado (solo si concepto = 1?)
echa_serv_desde = fecha: fecha_serv_hasta = fecha
oneda_id = "PES": moneda_ctz = "1.000"
bs = "Observaciones Comerciales, libre"


k = WSMTXCA.CrearFactura(concepto, tipo_doc, nro_doc, tipo_cbte, punto_vta, _
    cbt_desde, cbt_hasta, imp_total, imp_tot_conc, imp_neto, _
    imp_subtotal, imp_trib, imp_op_ex, fecha_cbte, fecha_venc_pago, _
    fecha_serv_desde, fecha_serv_hasta, _
    moneda_id, moneda_ctz, obs)
```

```
' Agrego los comprobantes asociados:
f False Then ' solo si es nc o nd
   tipo = 19
   pto_vta = 2
   nro = 1234
   ok = WSMTXCA.AgregarCmpAsoc(tipo, pto_vta, nro)
nd If

' Agrego impuestos varios
d = 99
esc = "Impuesto Municipal Matanza'"
ase_imp = "100.00"
lic = "1.00"
mporte = "1.00"
k = WSMTXCA.AgregarTributo(id, Desc, base_imp, alic, importe)

' Agrego subtotales de IVA
d = 5 ' 21%
ase_im = "100.00"
mporte = "21.00"
k = WSMTXCA.AgregarIva(id, base_imp, importe)

_mtx = 123456
od_mtx = "1234567890"
odigo = "P0001"
s = "Descripcion del producto P0001"
ty = "1.0000"
med = 7
recio = "100.00"
onif = "0.00"
od_iva = 5
mp_iva = "21.00"
mp_subtotal = "121.00"
k = WSMTXCA.AgregarItem(u_mtx, cod_mtx, codigo, ds, qty, _
           umed, precio, bonif, cod_iva, imp_iva, imp_subtotal)
k = WSMTXCA.AgregarItem(u_mtx, cod_mtx, codigo, ds, qty, _
           umed, precio, bonif, cod_iva, imp_iva, imp_subtotal)
k = WSMTXCA.AgregarItem(u_mtx, cod_mtx, "DESC", "Descuento", 0, _
           "99", 0#, 0, cod_iva, "-21.00", "-121.00")

' Solicito CAE:
ae = WSMTXCA.AutorizarComprobante()

' verifico que no haya errores
or Each er In WSMTXCA.Errores
   MsgBox er, vbInformation, "Error:"
ext

' Imprimo pedido y respuesta XML para depuración (errores de formato)
ebug.Print WSMTXCA.XmlRequest
ebug.Print WSMTXCA.XmlResponse

sgBox "Resultado:" & WSMTXCA.Resultado & " CAE: " & cae & _
      " Venc: " & WSMTXCA.Vencimiento & " Obs: " & WSMTXCA.obs, _
      vbInformation + vbOKOnly
```

Ejemplo para obtener códigos referenciales (tabla moneda):

```
' recupero tabla de parámetros de moneda ("id: descripción")
For Each x In WSMTXCA.ConsultarMonedas()
    Debug.Print x
```

```
Next

' busco la cotización del dolar (ver Param Mon)
ctz = WSMTXCA.ConsultarCotizacionMoneda("DOL")
MsgBox "Cotización Dólar: " & ctz
```

Descripción de la obtención de CAE / CAEA

Como primer paso, se debe crear una factura (utilizada internamente por la interfase para contener los valores a autorizar), llamando al método CrearFactura con los siguiente parámetros:

- tipo_doc, nro_doc: Tipo (80 CUIT, 96 DNI, etc.) y número de Documento
- tipo_cbte: Tipo de comprobante (según tabla de parámetros)
- punto_vta: Nº de punto de venta (debe estar autorizado para WSMTX: factura electrónica con detalle)
- cbt_desde, cbt_hasta: Nº de comprobante (desde, hasta, ambos el mismo número si es una factura individual)
- fecha_cbte: Fecha del comprobante (no puede ser mayor o menor a 10 días)
- concepto: tipo de factura (1: productos, 2: servicios, etc.)
- imp_total: Importe total de la factura
- imp_tot_conc: Importe total de conceptos no gravados por el IVA
- imp_neto: Importe neto (gravado por el IVA) de la factura
- imp_subtotal: subtotal sin iva
- imp_trib: Importe de otros tributos
- imp_op_ex: Importe de operaciones exentas
- moneda_id: Moneda de la factura (según tabla de parámetros)
- moneda_ctz: Cotización de la moneda de la factura
- fecha_vto_pago: Fecha de vencimiento de pago (si es de servicios)
- fecha_serv_desde, fecha_serv_hasta: fecha del servicios
- obs: observaciones comerciales (campo libre)
- caea: debe completarse si se informa una factura en esta modalidad (obtenido con SolicitarCAEA)

Luego, por cada artículo vendido (ítem), se debe llamar al método AgregarItem con los siguientes parámetros:

- u_mtx: cantidad de unidades(ej. 123456)
- cod_mtx: codigo mtx del producto (ej. "1234567890")
- codigo: codigo opcional del producto (ej. "P0001")
- ds: Descripción completa
- qty: Cantidad
- umed: Unidad de medida (según tabla de parámetros)
- precio: Precio Neto Unitario (facturas A) o precio final (facturas B)
- bonif: Bonificación
- cod_iva: Alícuota de IVA (según tabla de parámetros)
- imp_iva: importe liquidado de IVA
- imp_subtotal = Importe total (incluyendo descuentos e IVA)

Para items descriptivos, utilizar umed = 0. Para anticipos o descuentos, utilizar umed = 99.

Por cada alicuota de IVA, se debe llamar al método AgregarIva con los siguientes parámetros:

- id: código Alícuota de IVA (según tabla de parámetros)
- base_imp: base imponible (importe)

- importe: importe liquidado

De existir otros tributos, se debe llamar al método AgregarTributo con los siguientes parámetros:

- id: código tipo de impuesto (según tabla de parámetros)
- desc: descripción del tributo (por ej. "Impuesto Municipal Matanza")
- base_imp: base imponible (importe)
- alic: alicuota (porcentaje)
- importe: importe liquidado

También se puede llamar al método AgregarCmpAsoc para detallar los comprobantes asociados a una nota de crédito, con los siguientes parámetros:

- tipo: Código de tipo de comprobante.
- pto_vta: Punto de venta
- nro: Numero de comprobante

Como último paso, se debe llamar al método AutorizarComprobante o InformarComprobanteCAEA

Si no hubiere inconvenientes, la llamada debe devolver el CAE / CAEA y se establece el atributo WSMTXCA.Resultado = "A" (Aceptado) y WSMTXCA.Vencimiento. Sino, devuelve:

- cae = "": La interfase no pudo procesar la respuesta del WebService, o bien la conexión a internet esta caída, las direcciones de los servidores son incorrectas, o el servicio web rechazo los datos de la factura a generar. Se establece el atributo WSBFE.Resultado = "R" (Rechazado) y WSMTXCA.Obs con los diversos motivos de rechazo proporcionados por el webservice (ver WSFE.Motivo):

Aún devolviendo el CAE y con el resultado Aceptado, puede haber un motivo (ej "10017: Factura individual, DocTipo?: 80, DocNro? 33693450239 no se encuentra en condicion activa en los padrones de AFIP."), que funcionan como advertencia para revisar la base de datos y corregir posibles irregularidades.

Advertencia: Este servicio web no tiene reproceso (a diferencia de WSFE). En caso de falla o pérdida de comunicación, usar el método para recuperar un comprobante emitido (ConsultarComprobante)

Para más información ver especificaciones técnicas del WSMTX en el sitio de la AFIP:

- Manual Desarrollador: Información importante sobre la operatoria

Establecer campos individualmente (ejemplo WSMTXCA)

Para lenguajes que no soporten el pasaje de varios parametros a un método (por ej. SERVOY por su limitación de 6 argumentos), desde la versión 1.06a es posible crear la factura en blanco (todos los campos vacios) y luego establecer los campos individualmente:

```
ok = WSMTXCA.CrearFactura()
ok = WSMTXCA.EstablecerCampoFactura("concepto", concepto)
ok = WSMTXCA.EstablecerCampoFactura("tipo_doc", tipo_doc)
ok = WSMTXCA.EstablecerCampoFactura("nro_doc", nro_doc)
ok = WSMTXCA.EstablecerCampoFactura("tipo_cbte", tipo_cbte)
ok = WSMTXCA.EstablecerCampoFactura("punto_vta", punto_vta)
ok = WSMTXCA.EstablecerCampoFactura("cbt_desde", cbt_desde)
ok = WSMTXCA.EstablecerCampoFactura("cbt_hasta", cbt_hasta)
```

```
ok = WSMTXCA.EstablecerCampoFactura("imp_total", imp_total)
ok = WSMTXCA.EstablecerCampoFactura("imp_tot_conc", imp_tot_conc)
ok = WSMTXCA.EstablecerCampoFactura("imp_neto", imp_neto)
ok = WSMTXCA.EstablecerCampoFactura("imp_subtotal", imp_subtotal)
ok = WSMTXCA.EstablecerCampoFactura("imp_trib", imp_trib)
ok = WSMTXCA.EstablecerCampoFactura("imp_op_ex", imp_op_ex)
ok = WSMTXCA.EstablecerCampoFactura("fecha_cbte", fecha_cbte)
ok = WSMTXCA.EstablecerCampoFactura("fecha_venc_pago", fecha_venc_pago)
ok = WSMTXCA.EstablecerCampoFactura("fecha_serv_desde", fecha_serv_desde)
ok = WSMTXCA.EstablecerCampoFactura("fecha_serv_hasta", fecha_serv_hasta)
ok = WSMTXCA.EstablecerCampoFactura("moneda_id", moneda_id)
ok = WSMTXCA.EstablecerCampoFactura("moneda_ctz", moneda_ctz)
ok = WSMTXCA.EstablecerCampoFactura("observaciones", Obs)
ok = WSMTXCA.EstablecerCampoFactura("caea", CAEA)
ok = WSMTXCA.EstablecerCampoFactura("vencimiento", vencimiento)
```

Para los items de detalle sería similar:

```
ok = WSMTXCA.AgregarItem()
ok = WSMTXCA.EstablecerCampoItem("u_mtx", u_mtx)
ok = WSMTXCA.EstablecerCampoItem("cod_mtx", cod_mtx)
ok = WSMTXCA.EstablecerCampoItem("codigo", codigo)
ok = WSMTXCA.EstablecerCampoItem("ds", ds)
ok = WSMTXCA.EstablecerCampoItem("qty", qty)
ok = WSMTXCA.EstablecerCampoItem("umed", umed)
ok = WSMTXCA.EstablecerCampoItem("precio", precio)
ok = WSMTXCA.EstablecerCampoItem("bonif", bonif)
ok = WSMTXCA.EstablecerCampoItem("iva_id", cod_iva)
ok = WSMTXCA.EstablecerCampoItem("imp_iva", imp_iva)
ok = WSMTXCA.EstablecerCampoItem("imp_subtotal", imp_subtotal)
```

Interfase por archivos de texto símil SIAP - RECE

Para lenguajes donde no es posible utilizar objetos COM, se desarrolló una interfaz de texto (command line interfase) para poder utilizar los Web Services de la AFIP, que funciona como un programa independiente, manteniendo las ventajas y características presentadas anteriormente.

Formatos de archivos de intercambio soportados:

- Archivos TXT de Texto plano estilo COBOL y aplicativos SIAP AFIP
- Tablas DBF para dBase III / Fox / Clipper, Harbour (xBase), etc.
- Formato JSON para lenguajes de programación más modernos (PHP, JAVA, JavaScript, etc.)

La herramienta es multiplataforma (Linux/Windows)

Ver secciones anteriores sobre la descripción de las opciones de configuración, campos y procedimientos:

- Consideraciones Generales
- Autenticación (WSAA): ticket de acceso, certificados y claves privadas
- Gestión de CAE (WSFEv1): mercado interno sin detalle (version 1)
- Gestión de CAE (WSMTXCA): mercado interno con detalle ("matrix")
- Gestión de CAE (WSFEX): exportación

Configuración

Editar el archivo RECE.INI en la carpeta de la intefase (C:\PYAFIPWS):

- CERT: ubicación del archivo certificado (ver WSAA)
- PRIVATEKEY: ubicación del archivo de la clave privada (ver WSAA)
- CUIT: CUIT del emisor
- ENTRADA: ubicación del archivo de texto de entrada (para cada webservice)
- SALIDA: ubicación del archivo de texto de salida (para cada webservice)
- URL: dirección de los servicios web de producción (para cada webservice)
- Seccion [DBF]: configura los nombres de archivos con las tablas requeridas

Ejemplo:

```
[WSAA]
CERT=C:\SISTEMA\empresa.crt
PRIVATEKEY= C:\SISTEMA\empresa.key
URL=https://wsaa.afip.gov.ar/ws/services/LoginCms?wsdl
[WSFEv1]
CUIT=30000000000
CBU=2222222222222222222222
ENTRADA=entrada.txt
SALIDA=salida.txt
URL=https://servicios1.afip.gov.ar/wsfev1/service.asmx?WSDL

[WSMTXCA]
CUIT=30000000000
ENTRADA=entrada.txt
SALIDA=salida.txt
Reprocesar= S
URL=https://serviciosjava.afip.gob.ar/wsmtxca/services/MTXCAService

[WSBFE]
# aplica también a WSBFEv1
CUIT=30000000000
ENTRADA= C:\SISTEMA\entrada.txt
SALIDA= C:\SISTEMA\salida.txt
URL=https://servicios1.afip.gov.ar/wsbfev1/service.asmx?WSDL

[WSFEXv1]
# aplica también a WSFEX
CUIT=30000000000
ENTRADA=entrada.txt
SALIDA=salida.txt
##URL=https://servicios1.afip.gov.ar/wsfexv1/service.asmx?WSDL

[DBF]
Encabezado = encabeza.dbf
Tributo = tributo.dbf
Iva = iva.dbf
Comprobante Asociado = cbteasoc.dbf
Detalles = detalles.dbf
Datos Opcionales = opcionales.dbf
```

Sección [WSAA]

Configuración referente al web service de autenticación:

- CERT: ubicación del archivo que contiene el certificado
- PRIVATEKEY: ubicación de la clave privada correspondiente al certificado
- URL: dirección del servidor de AFIP. Por ej., para producción:
  https://wsaa.afip.gov.ar/ws/services/LoginCms?wsdl

Sección [WSFEv1]

Configuración referente al web service de factura nacional:

- CUIT: nº de cuit del emisor (sin guiones). Por ej.: 20267565393
- ENTRADA: nombre de archivo con datos para autorizar/generar (si corresponde). Por ej.: entrada.txt
- SALIDA: nombre de archivo para almacenar el resultado (si corresponde). Por ej: salida.txt
- URL: dirección del servidor de AFIP. Por ej., para producción:
  URL= https://servicios1.afip.gov.ar/wsfev1/service.asmx?WSDL

Sección [WSFEX]

Configuración referente al web service de factura exportación:

- CUIT: nº de cuit del emisor (sin guiones). Por ej.: 20267565393
- ENTRADA: nombre de archivo con datos para autorizar/generar (si corresponde). Por ej.: entrada.txt
- SALIDA: nombre de archivo para almacenar el resultado (si corresponde). Por ej: salida.txt
- URL: dirección del servidor de AFIP. Por ej., para producción:
  URL= https://servicios1.afip.gov.ar/wsfex/service.asmx

Sección [DBF]

Configuración referente a las tablas DBF, por cada una se especifica la ruta y el nombre de archivo, ej:

- Encabezado = encabeza.dbf
- Tributo = tributo.dbf
- Iva = iva.dbf
- Comprobante Asociado = cbteasoc.dbf
- Detalles = detalles.dbf

Sección [PROXY]

Configuración referente al servidor intermedio de salida a internet (firewall, antivirus, proxy, etc.), ej:

- HOST = localhost
- PORT = 8000
- USER = mariano
- PASS = reingart

NOTA: solo debe configurarse esta sección si corresponde, contactar al administrador de red de la empresa para determinar los parámetros correctos de cada caso. Para más información ver Errores de Conectividad

Formato de archivos de entrada y salida

Factura electrónica normal (WSFE):

El formato utilizado es similar a RECE pero extendido con algunos campos necesarios para utilizar los Servicios Web. A continuación se detalla nombre del campo, posición inicial, longitud y tipo.

- fecha_cbte: (2,8,A)
- tipo_cbte: (10,2,N)
- punto_vta: (13,4,N)
- cbt_desde: (17,8,N)
- cbt_hasta: (25,8,N)
- tipo_doc: (36,2,N)
- nro_doc: (38,11,N)
- imp_total: (79,15,I)
- imp_tot_conc: (94,15,I)
- imp_neto: (109,15,I)
- impto_liq: (124,15,I),
- impto_liq_rni: (139,15,I)
- imp_op_ex: (154,15,I)
- cae: (261,14,N)
- fecha_vto: (275,8,A)
- resultado: (291,1,A)
- motivo: (292,2,A)
- reproceso: (294,1,A)
- fecha_venc_pago: (295,8,A)
- presta_serv: (303,1,N)
- fecha_serv_desde: (304,8,A)
- fecha_serv_hasta: (312,8,A)
- id: (320,15,N)

Tipos de campo:

- A: Alfanumerico
- N: Numerico
- I: Importes con 2 decimales (sin incluir punto)

Los espacios no definidos se ignoran (completar con un espacio en blanco)

Para más información sobre los campos, ver parámetros y valores que retorna el método Aut en el presente manual

Como formato de entrada es posible utilizar el mismo archivo generado para el aplicativo SIAP/RECE, pero no se podrán facturar servicios ya que los campo presta_serv y fecha_serv_desde y fecha_serv_hasta no están contemplados en dicho formato.

Factura electrónica bienes de capital (WSBFE)

El formato es similar al de factura electrónica normal, pero varía el tipo de registro (0: Encabezado, 1: Detalle de cada Item):

Encabezado:

- Campo: tipo_reg Posición: 1 Longitud: 1 Tipo: Numerico Valor: 0
- Campo: fecha_cbte Posición: 2 Longitud: 8 Tipo: Alfanumerico
- Campo: tipo_cbte Posición: 10 Longitud: 2 Tipo: Numerico
- Campo: punto_vta Posición: 12 Longitud: 4 Tipo: Numerico
- Campo: cbte_nro Posición: 16 Longitud: 8 Tipo: Numerico
- Campo: tipo_doc Posición: 24 Longitud: 2 Tipo: Numerico
- Campo: nro_doc Posición: 26 Longitud: 11 Tipo: Numerico
- Campo: imp_total Posición: 37 Longitud: 15 Tipo: Importe
- Campo: imp_tot_conc Posición: 52 Longitud: 15 Tipo: Importe
- Campo: imp_neto Posición: 67 Longitud: 15 Tipo: Importe
- Campo: impto_liq Posición: 82 Longitud: 15 Tipo: Importe
- Campo: impto_liq_rni Posición: 97 Longitud: 15 Tipo: Importe
- Campo: imp_op_ex Posición: 112 Longitud: 15 Tipo: Importe
- Campo: impto_perc Posición: 127 Longitud: 15 Tipo: Importe
- Campo: imp_iibb Posición: 142 Longitud: 15 Tipo: Importe
- Campo: impto_perc_mun Posición: 157 Longitud: 15 Tipo: Importe
- Campo: imp_internos Posición: 172 Longitud: 15 Tipo: Importe
- Campo: imp_moneda_id Posición: 187 Longitud: 3 Tipo: Alfanumerico
- Campo: imp_moneda_ctz Posición: 190 Longitud: 10 Tipo: Importe
- Campo: zona Posición: 200 Longitud: 5 Tipo: Alfanumerico
- Campo: cae Posición: 205 Longitud: 14 Tipo: Numerico
- Campo: fecha_vto Posición: 219 Longitud: 8 Tipo: Alfanumerico
- Campo: resultado Posición: 227 Longitud: 1 Tipo: Alfanumerico
- Campo: obs Posición: 228 Longitud: 2 Tipo: Alfanumerico
- Campo: reproceso Posición: 230 Longitud: 1 Tipo: Alfanumerico
- Campo: id Posición: 231 Longitud: 15 Tipo: Numerico
- Campo: cancela_misma_moneda_ext Posición: 246 Longitud: 1 Tipo: Alfanumerico
- Campo: condicion_iva_receptor_id Posición: 247 Longitud: 4 Tipo: Numerico

Detalle (por cada item):

- Campo: tipo_reg Posición: 1 Longitud: 1 Tipo: Numerico Valor: 1
- Campo: ncm Posición: 2 Longitud: 15 Tipo: Alfanumerico
- Campo: sec Posición: 17 Longitud: 15 Tipo: Alfanumerico
- Campo: qty Posición: 32 Longitud: 15 Tipo: Importe
- Campo: umed Posición: 47 Longitud: 5 Tipo: Numerico
- Campo: precio Posición: 52 Longitud: 15 Tipo: Importe
- Campo: bonif Posición: 67 Longitud: 15 Tipo: Importe
- Campo: imp_total Posición: 82 Longitud: 15 Tipo: Importe
- Campo: iva_id Posición: 97 Longitud: 5 Tipo: Numerico
- Campo: ds Posición: 102 Longitud: 200 Tipo: Alfanumerico

Si se informa el campo ID, la factura se genera con dicho número de secuencia. De lo contrario, para evitar problemas y poder reprocesar el CAE, se genera con un número de secuencia del formato TPPPPNNNNNNNN (tipo de comprobante, punto de venta y número de comprobante).

Factura electrónica exportación (WSFEX)

El formato es similar al de factura electrónica bienes de capital, pero varía el tipo de registro (0: Encabezado, 1: Detalle de cada Item, 2: Permiso, 3: Comprobante asociado):

Formato:

Encabezado:

- Campo: tipo_reg Posición: 1 Longitud: 1 Tipo: Numerico Decimales:
- Campo: fecha_cbte Posición: 2 Longitud: 8 Tipo: Alfanumerico Decimales:
- Campo: tipo_cbte Posición: 10 Longitud: 2 Tipo: Numerico Decimales:
- Campo: punto_vta Posición: 12 Longitud: 4 Tipo: Numerico Decimales:
- Campo: cbte_nro Posición: 16 Longitud: 8 Tipo: Numerico Decimales:
- Campo: tipo_expo Posición: 24 Longitud: 1 Tipo: Numerico Decimales:
- Campo: permiso_existente Posición: 25 Longitud: 1 Tipo: Alfanumerico Decimales:
- Campo: dst_cmp Posición: 26 Longitud: 3 Tipo: Numerico Decimales:
- Campo: cliente Posición: 29 Longitud: 200 Tipo: Alfanumerico Decimales:
- Campo: cuit_pais_cliente Posición: 229 Longitud: 11 Tipo: Numerico Decimales:
- Campo: domicilio_cliente Posición: 240 Longitud: 300 Tipo: Alfanumerico Decimales:
- Campo: id_impositivo Posición: 540 Longitud: 50 Tipo: Alfanumerico Decimales:
- Campo: imp_total Posición: 590 Longitud: 15 Tipo: Importe Decimales: 3
- Campo: moneda_id Posición: 605 Longitud: 3 Tipo: Alfanumerico Decimales:
- Campo: moneda_ctz Posición: 608 Longitud: 10 Tipo: Importe Decimales: 6
- Campo: obs_comerciales Posición: 618 Longitud: 1000 Tipo: Alfanumerico Decimales:
- Campo: obs Posición: 1618 Longitud: 1000 Tipo: Alfanumerico Decimales:
- Campo: forma_pago Posición: 2618 Longitud: 50 Tipo: Alfanumerico Decimales:
- Campo: incoterms Posición: 2668 Longitud: 3 Tipo: Alfanumerico Decimales:
- Campo: incoterms_ds Posición: 2671 Longitud: 20 Tipo: Alfanumerico Decimales:
- Campo: idioma_cbte Posición: 2691 Longitud: 1 Tipo: Alfanumerico Decimales:
- Campo: cae Posición: 2692 Longitud: 14 Tipo: Numerico Decimales:
- Campo: fecha_vto Posición: 2706 Longitud: 8 Tipo: Alfanumerico Decimales:
- Campo: resultado Posición: 2714 Longitud: 1 Tipo: Alfanumerico Decimales:
- Campo: reproceso Posición: 2715 Longitud: 1 Tipo: Alfanumerico Decimales:
- Campo: motivos_obs Posición: 2716 Longitud: 40 Tipo: Alfanumerico Decimales:
- Campo: id Posición: 2756 Longitud: 15 Tipo: Numerico Decimales:
- Campo: fch_venc_cae Posición: 2771 Longitud: 8 Tipo: Alfanumerico Decimales:

Detalle:

- Campo: tipo_reg Posición: 1 Longitud: 1 Tipo: Numerico Decimales:
- Campo: codigo Posición: 2 Longitud: 30 Tipo: Alfanumerico Decimales:
- Campo: qty Posición: 32 Longitud: 12 Tipo: Importe Decimales:
- Campo: umed Posición: 44 Longitud: 2 Tipo: Numerico Decimales:
- Campo: precio Posición: 46 Longitud: 12 Tipo: Importe Decimales: 3
- Campo: imp_total Posición: 58 Longitud: 14 Tipo: Importe Decimales: 3
- Campo: ds Posición: 72 Longitud: 4000 Tipo: Alfanumerico Decimales:

Permiso:

- Campo: tipo_reg Posición: 1 Longitud: 1 Tipo: Numerico Decimales:

- Campo: id_permiso Posición: 2 Longitud: 16 Tipo: Alfanumerico Decimales:
- Campo: dst_merc Posición: 18 Longitud: 3 Tipo: Numerico Decimales:

Comprobante Asociado:

- Campo: tipo_reg Posición: 1 Longitud: 1 Tipo: Numerico Decimales:
- Campo: cbte_tipo Posición: 2 Longitud: 3 Tipo: Numerico Decimales:
- Campo: cbte_punto_vta Posición: 5 Longitud: 4 Tipo: Numerico Decimales:
- Campo: cbte_nro Posición: 9 Longitud: 8 Tipo: Numerico Decimales:

Nota: se utilizan 3 decimales para los importes y 6 para la cotización de la moneda

Factura electrónica Mercado Interno Versión 1 (WSFEv1)

El formato es similar al de factura electrónica bienes de capital, pero varía el tipo de registro (0: Encabezado, 1: Tributo, 2: IVA, 3: Comprobante asociado, 6: Opcionales, 8: Periodo Comprobante asociado, A: Actividad) y no tiene detalle de artículos:

Encabezado:

- Campo: tipo_reg Posición: 1 Longitud: 1 Tipo: Numerico
- Campo: fecha_cbte Posición: 2 Longitud: 8 Tipo: Alfanumerico
- Campo: tipo_cbte Posición: 10 Longitud: 2 Tipo: Numerico
- Campo: punto_vta Posición: 12 Longitud: 4 Tipo: Numerico
- Campo: cbt_desde Posición: 16 Longitud: 8 Tipo: Numerico
- Campo: cbt_hasta Posición: 24 Longitud: 8 Tipo: Numerico
- Campo: concepto Posición: 32 Longitud: 1 Tipo: Numerico
- Campo: tipo_doc Posición: 33 Longitud: 2 Tipo: Numerico
- Campo: nro_doc Posición: 35 Longitud: 11 Tipo: Numerico
- Campo: imp_total Posición: 46 Longitud: 15 Tipo: Importe Decimales: 2
- Campo: no_usar Posición: 61 Longitud: 15 Tipo: Importe Decimales: 2
- Campo: imp_tot_conc Posición: 76 Longitud: 15 Tipo: Importe Decimales: 2
- Campo: imp_neto Posición: 91 Longitud: 15 Tipo: Importe Decimales: 2
- Campo: imp_iva Posición: 106 Longitud: 15 Tipo: Importe Decimales: 2
- Campo: imp_trib Posición: 121 Longitud: 15 Tipo: Importe Decimales: 2
- Campo: imp_op_ex Posición: 136 Longitud: 15 Tipo: Importe Decimales: 2
- Campo: moneda_id Posición: 151 Longitud: 3 Tipo: Alfanumerico
- Campo: moneda_ctz Posición: 154 Longitud: 10 Tipo: Importe Decimales: 6
- Campo: fecha_venc_pago Posición: 164 Longitud: 8 Tipo: Alfanumerico
- Campo: cae Posición: 172 Longitud: 14 Tipo: Alfanumerico
- Campo: fch_venc_cae Posición: 186 Longitud: 8 Tipo: Alfanumerico
- Campo: resultado Posición: 194 Longitud: 1 Tipo: Alfanumerico
- Campo: motivos_obs Posición: 195 Longitud: 1000 Tipo: Alfanumerico
- Campo: err_code Posición: 1195 Longitud: 6 Tipo: Alfanumerico
- Campo: err_msg Posición: 1201 Longitud: 1000 Tipo: Alfanumerico
- Campo: reproceso Posición: 2201 Longitud: 1 Tipo: Alfanumerico
- Campo: emision_tipo Posición: 2202 Longitud: 4 Tipo: Alfanumerico
- Campo: fecha_serv_desde Posición: 2206 Longitud: 8 Tipo: Alfanumerico
- Campo: fecha_serv_hasta Posición: 2214 Longitud: 8 Tipo: Alfanumerico
- Campo: tipo_cbte Posición: 2222 Longitud: 3 Tipo: Numerico
- Campo: punto_vta Posición: 2225 Longitud: 5 Tipo: Numerico

- Campo: fecha_hs_gen Posición: 2230 Longitud: 14 Tipo: Alfanumerico formato yyyymmddhhmiss (p/CAEA contingencia)
- Campo: cancela_misma_moneda_ext Posición: 2244 Longitud: 1 Tipo: Alfanumerico Decimales
- Campo: condicion_iva_receptor_id Posición: 2245 Longitud: 4 Tipo: Numerico Decimales

Tributo:

- Campo: tipo_reg Posición: 1 Longitud: 1 Tipo: Numerico
- Campo: tributo_id Posición: 2 Longitud: 16 Tipo: Numerico
- Campo: desc Posición: 18 Longitud: 100 Tipo: Alfanumerico
- Campo: base_imp Posición: 118 Longitud: 15 Tipo: Importe Decimales: 2
- Campo: alic Posición: 133 Longitud: 15 Tipo: Importe Decimales: 2
- Campo: importe Posición: 148 Longitud: 15 Tipo: Importe Decimales: 2

Iva:

- Campo: tipo_reg Posición: 1 Longitud: 1 Tipo: Numerico
- Campo: iva_id Posición: 2 Longitud: 16 Tipo: Numerico
- Campo: base_imp Posición: 18 Longitud: 15 Tipo: Importe Decimales: 2
- Campo: importe Posición: 33 Longitud: 15 Tipo: Importe Decimales: 2

Comprobante Asociado:

- Campo: tipo_reg Posición: 1 Longitud: 1 Tipo: Numerico
- Campo: tipo Posición: 2 Longitud: 3 Tipo: Numerico
- Campo: pto_vta Posición: 5 Longitud: 4 Tipo: Numerico
- Campo: nro Posición: 9 Longitud: 8 Tipo: Numerico
- Campo: fecha Posición: 17 Longitud: 8 Tipo: Numerico
- Campo: cuit Posición: 25 Longitud: 11 Tipo: Numerico

Opcionales: *Disponible desde actualización 1.33b*

- Campo: tipo_reg Posición: 1 Longitud: 1 Tipo: Numerico
- Campo: opcional_id Posición: 2 Longitud: 4 Tipo: Alfanumerico
- Campo: valor Posición: 6 Longitud: 250 Tipo: Alfanumerico

Compradores: *Disponible desde actualización 1.37a* (RG-4109-E Bienes Muebles Registrables)

- Campo: tipo_reg Posición: 1 Longitud: 1 Tipo: Numerico Decimales:
- Campo: doc_tipo Posición: 2 Longitud: 3 Tipo: Numerico Decimales:
- Campo: doc_nro Posición: 5 Longitud: 80 Tipo: Numerico Decimales:
- Campo: porcentaje Posición: 85 Longitud: 6 Tipo: Importe Decimales: 2

Periodo Cbte Asoc: *Disponible desde actualización 1.38a*

- Campo: tipo_reg Posición: 1 Longitud: 1 Tipo: Numerico
- Campo: fecha_desde Posición: 2 Longitud: 8 Tipo: Numerico
- Campo: fecha_hasta Posición: 10 Longitud: 8 Tipo: Numerico

Actividades: *Disponible desde actualización 1.* (RG 5259/2022 y RG5259/2022)

- Campo: tipo_reg Posición: 1 Longitud: 1 Tipo: Alfanumerico
- Campo: actividad_id Posición: 2 Longitud: 6 Tipo: Alfanumerico

Factura electrónica Mercado Interno Matrix (WSMTXCA)

El formato es similar al de factura electrónica mercado interno version 1, pero varía el tipo de registro (0: Encabezado, 1: Tributo, 2: IVA, 3: Comprobante asociado, 4: Detalle de cada item) y tiene detalle de artículos:

Encabezado:

- Campo: tipo_reg Posición: 1 Longitud: 1 Tipo: Numerico Decimales:
- Campo: fecha_cbte Posición: 2 Longitud: 10 Tipo: Alfanumerico Decimales:
- Campo: tipo_cbte Posición: 12 Longitud: 2 Tipo: Numerico Decimales:
- Campo: punto_vta Posición: 14 Longitud: 4 Tipo: Numerico Decimales:
- Campo: cbt_desde Posición: 18 Longitud: 8 Tipo: Numerico Decimales:
- Campo: cbt_hasta Posición: 26 Longitud: 8 Tipo: Numerico Decimales:
- Campo: concepto Posición: 34 Longitud: 1 Tipo: Numerico Decimales:
- Campo: tipo_doc Posición: 35 Longitud: 2 Tipo: Numerico Decimales:
- Campo: nro_doc Posición: 37 Longitud: 11 Tipo: Numerico Decimales:
- Campo: imp_total Posición: 48 Longitud: 15 Tipo: Importe Decimales: 2
- Campo: imp_tot_conc Posición: 63 Longitud: 15 Tipo: Importe Decimales: 2
- Campo: imp_neto Posición: 78 Longitud: 15 Tipo: Importe Decimales: 2
- Campo: imp_subtotal Posición: 93 Longitud: 15 Tipo: Importe Decimales: 2
- Campo: imp_trib Posición: 108 Longitud: 15 Tipo: Importe Decimales: 2
- Campo: imp_op_ex Posición: 123 Longitud: 15 Tipo: Importe Decimales: 2
- Campo: moneda_id Posición: 138 Longitud: 3 Tipo: Alfanumerico Decimales:
- Campo: moneda_ctz Posición: 141 Longitud: 10 Tipo: Importe Decimales: 6
- Campo: fecha_venc_pago Posición: 151 Longitud: 10 Tipo: Alfanumerico Decimales:
- Campo: fecha_serv_desde Posición: 161 Longitud: 10 Tipo: Alfanumerico Decimales:
- Campo: fecha_serv_hasta Posición: 171 Longitud: 10 Tipo: Alfanumerico Decimales:
- Campo: cae Posición: 181 Longitud: 14 Tipo: Numerico Decimales:
- Campo: fch_venc_cae Posición: 195 Longitud: 10 Tipo: Alfanumerico Decimales:
- Campo: resultado Posición: 205 Longitud: 1 Tipo: Alfanumerico Decimales:
- Campo: motivos_obs Posición: 206 Longitud: 1000 Tipo: Alfanumerico Decimales:
- Campo: err_code Posición: 1206 Longitud: 6 Tipo: Alfanumerico Decimales:
- Campo: err_msg Posición: 1212 Longitud: 1000 Tipo: Alfanumerico Decimales:
- Campo: reproceso Posici?n: 2212 Longitud: 1 Tipo: Alfanumerico Decimales:
- Campo: emision_tipo Posici?n: 2213 Longitud: 4 Tipo: Alfanumerico Decimales:
- Campo: observaciones Posición: 2217 Longitud: 1000 Tipo: Alfanumerico Decimales:

Tributo:

- Campo: tipo_reg Posición: 1 Longitud: 1 Tipo: Numerico Decimales:
- Campo: tributo_id Posición: 2 Longitud: 3 Tipo: Alfanumerico Decimales:
- Campo: desc Posición: 5 Longitud: 100 Tipo: Alfanumerico Decimales:
- Campo: base_imp Posición: 105 Longitud: 15 Tipo: Importe Decimales: 2
- Campo: alic Posición: 120 Longitud: 15 Tipo: Importe Decimales: 2
- Campo: importe Posición: 135 Longitud: 15 Tipo: Importe Decimales: 2

Iva:

- Campo: tipo_reg Posición: 1 Longitud: 1 Tipo: Numerico Decimales:
- Campo: iva_id Posición: 2 Longitud: 3 Tipo: Alfanumerico Decimales:
- Campo: base_imp Posición: 5 Longitud: 15 Tipo: Importe Decimales: 2
- Campo: importe Posición: 20 Longitud: 15 Tipo: Importe Decimales: 2

Comprobante Asociado:

- Campo: tipo_reg Posición: 1 Longitud: 1 Tipo: Numerico Decimales:
- Campo: tipo Posición: 2 Longitud: 3 Tipo: Numerico Decimales:
- Campo: pto_vta Posición: 5 Longitud: 4 Tipo: Numerico Decimales:
- Campo: nro Posición: 9 Longitud: 8 Tipo: Numerico Decimales:

Opcionales:

- Campo: tipo_reg Posición: 1 Longitud: 1 Tipo: Numerico
- Campo: opcional_id Posición: 2 Longitud: 4 Tipo: Alfanumerico
- Campo: valor Posición: 6 Longitud: 250 Tipo: Alfanumerico

Detalle de item:

- Campo: tipo_reg Posición: 1 Longitud: 1 Tipo: Numerico Decimales:
- Campo: u_mtx Posición: 2 Longitud: 10 Tipo: Numerico Decimales:
- Campo: cod_mtx Posición: 12 Longitud: 30 Tipo: Alfanumerico Decimales:
- Campo: codigo Posición: 42 Longitud: 30 Tipo: Alfanumerico Decimales:
- Campo: qty Posición: 72 Longitud: 15 Tipo: Importe Decimales: 3 (hasta 6 decimales con .)
- Campo: umed Posición: 87 Longitud: 3 Tipo: Numerico Decimales:
- Campo: precio Posición: 90 Longitud: 15 Tipo: Importe Decimales: 3 (hasta 6 decimales con .)
- Campo: bonif Posición: 105 Longitud: 15 Tipo: Importe Decimales: 3 (hasta 6 decimales con .)
- Campo: iva_id Posición: 120 Longitud: 3 Tipo: Numerico Decimales:
- Campo: imp_iva Posición: 123 Longitud: 15 Tipo: Importe Decimales: 2
- Campo: imp_subtotal Posición: 138 Longitud: 15 Tipo: Importe Decimales: 2
- Campo: ds Posición: 153 Longitud: 4000 Tipo: Alfanumerico Decimales:

Factura electrónica Exportación Version 1 (WSFEXv1)

El formato es similar al de factura electrónica bienes de capital, pero varía el tipo de registro (0: Encabezado, 1: Detalle de cada Item, 2: Permiso, 3: Comprobante asociado, A: Actividades):

Encabezado:

- Campo: tipo_reg Posición: 1 Longitud: 1 Tipo: Numerico Decimales:
- Campo: fecha_cbte Posición: 2 Longitud: 8 Tipo: Alfanumerico Decimales:
- Campo: tipo_cbte Posición: 10 Longitud: 2 Tipo: Numerico Decimales:
- Campo: punto_vta Posición: 12 Longitud: 4 Tipo: Numerico Decimales:
- Campo: cbte_nro Posición: 16 Longitud: 8 Tipo: Numerico Decimales:
- Campo: tipo_expo Posición: 24 Longitud: 1 Tipo: Numerico Decimales:
- Campo: permiso_existente Posición: 25 Longitud: 1 Tipo: Alfanumerico Decimales:
- Campo: pais_dst_cmp Posición: 26 Longitud: 3 Tipo: Numerico Decimales:
- Campo: nombre_cliente Posición: 29 Longitud: 200 Tipo: Alfanumerico Decimales:
- Campo: cuit_pais_cliente Posición: 229 Longitud: 11 Tipo: Numerico Decimales:
- Campo: domicilio_cliente Posición: 240 Longitud: 300 Tipo: Alfanumerico Decimales:

- Campo: id_impositivo Posición: 540 Longitud: 50 Tipo: Alfanumerico Decimales:
- Campo: imp_total Posición: 590 Longitud: 15 Tipo: Importe Decimales: 2
- Campo: moneda_id Posición: 605 Longitud: 3 Tipo: Alfanumerico Decimales:
- Campo: moneda_ctz Posición: 608 Longitud: 10 Tipo: Importe Decimales: 6
- Campo: obs_comerciales Posición: 618 Longitud: 4000 Tipo: Alfanumerico Decimales:
- Campo: obs_generales Posición: 4618 Longitud: 1000 Tipo: Alfanumerico Decimales:
- Campo: forma_pago Posición: 5618 Longitud: 50 Tipo: Alfanumerico Decimales:
- Campo: incoterms Posición: 5668 Longitud: 3 Tipo: Alfanumerico Decimales:
- Campo: incoterms_ds Posición: 5671 Longitud: 20 Tipo: Alfanumerico Decimales:
- Campo: idioma_cbte Posición: 5691 Longitud: 1 Tipo: Alfanumerico Decimales:
- Campo: cae Posición: 5692 Longitud: 14 Tipo: Numerico Decimales:
- Campo: fecha_vto Posición: 5706 Longitud: 8 Tipo: Alfanumerico Decimales:
- Campo: resultado Posición: 5714 Longitud: 1 Tipo: Alfanumerico Decimales:
- Campo: reproceso Posición: 5715 Longitud: 1 Tipo: Alfanumerico Decimales:
- Campo: motivos_obs Posición: 5716 Longitud: 1000 Tipo: Alfanumerico Decimales:
- Campo: id Posición: 6716 Longitud: 15 Tipo: Numerico Decimales:
- Campo: fch_venc_cae Posición: 6731 Longitud: 8 Tipo: Alfanumerico Decimales:
- Campo: excepcion Posición: 6739 Longitud: 100 Tipo: Alfanumerico Decimales:
- Campo: err_code Posición: 6839 Longitud: 100 Tipo: Alfanumerico Decimales:
- Campo: err_msg Posición: 6939 Longitud: 1000 Tipo: Alfanumerico Decimales:
- Campo: fecha_pago Posición: 7939 Longitud: 8 Tipo: Alfanumerico Decimales:

Detalle:

- Campo: tipo_reg Posición: 1 Longitud: 1 Tipo: Numerico Decimales:
- Campo: codigo Posición: 2 Longitud: 50 Tipo: Alfanumerico Decimales:
- Campo: qty Posición: 52 Longitud: 12 Tipo: Importe Decimales: 6
- Campo: umed Posición: 64 Longitud: 2 Tipo: Numerico Decimales:
- Campo: precio Posición: 66 Longitud: 12 Tipo: Importe Decimales: 6
- Campo: importe Posición: 78 Longitud: 13 Tipo: Importe Decimales: 2
- Campo: bonif Posición: 91 Longitud: 12 Tipo: Importe Decimales: 6
- Campo: ds Posición: 103 Longitud: 4000 Tipo: Alfanumerico Decimales:

Permiso:

- Campo: tipo_reg Posición: 1 Longitud: 1 Tipo: Numerico Decimales:
- Campo: id_permiso Posición: 2 Longitud: 16 Tipo: Alfanumerico Decimales:
- Campo: dst_merc Posición: 18 Longitud: 3 Tipo: Numerico Decimales:

Comprobante Asociado:

- Campo: tipo_reg Posición: 1 Longitud: 1 Tipo: Numerico Decimales:
- Campo: cbte_tipo Posición: 2 Longitud: 3 Tipo: Numerico Decimales:
- Campo: cbte_punto_vta Posición: 5 Longitud: 4 Tipo: Numerico Decimales:
- Campo: cbte_nro Posición: 9 Longitud: 8 Tipo: Numerico Decimales:
- Campo: cbte_cuit Posición: 17 Longitud: 11 Tipo: Numerico Decimales:

Actividades:

- Campo: tipo_reg Posición: 1 Longitud: 1 Tipo: Alfanumerico
- Campo: actividad_id Posición: 2 Longitud: 6 Tipo: Numerico Decimales:

Formato Tablas xBase DBF (dBase III / Fox / Clipper)

Las ultimas versiones de la interfaz, ademas de manejo por archivo de texto, soporta manejo por tablas DBF. Estas tablas pueden ser generadas con cualquier librería o aplicación compatible (incluso pueden ser abiertas con planillas de cálculo).

Los tipos de campos son:

- C(l): caracter (longitud)
- N(l.d): numerico (longitud y decimales)
- M: camos memo (usado para errores y mensajes extensos >250 caracteres)
- D: campos fecha

Los nombres de los campos son los mismos que se manejan para el archivo de texto, pero sin el guión bajo y recortados a 10 caracteres.

Se utiliza la librería Python dBase (dbf-0.88.19), que soporta los siguientes tipos de archivos:

- dBase III Plus
- dBase III Plus w/memos
- dBase IV
- dBase IV SQL
- dBase IV w/memos
- dBase IV w/memos
- dBase IV w/SQL table
- dBase V
- FoxBASE
- FoxPro w/memos
- Visual FoxPro
- Visual FoxPro (auto increment field)

Reconoce archivos con extensión .DBF, .DBT, .FPT, entre otros.

Factura electrónica mercado interno (WSFEv1)

El formato utilizado es similar al formato de texto de ancho fijo explicado anteriormente, y cada tipo de registro se almacena en una tabla separada.

IMPORTANTE: algunos campos han cambiado su denominación en las últimas versiones para hacerlos compatibles con las tablas PDF (solo se muestra el nombre de campo perteneciente a la última actualización)

Encabezado (encabeza.dbf)

Encabezado (datos generales de la factura):

- Campo: tiporeg N(1,0)
- Campo: fechacbte C(8)
- Campo: tipocbte N(2,0)
- Campo: puntovta N(4,0)
- Campo: cbtdesde N(8,0)
- Campo: cbthasta N(8,0)

- Campo: concepto N(1,0)
- Campo: tipodoc N(2,0)
- Campo: nrodoc N(11,0)
- Campo: imptotal N(15,2)
- Campo: nousar N(15,2)
- Campo: imptotconc N(15,2)
- Campo: impneto N(15,2)
- Campo: impiva N(15,2)
- Campo: imptrib N(15,2)
- Campo: impopex N(15,2)
- Campo: monedaid C(3)
- Campo: monedactz N(10,2)
- Campo: fechavencp C(8)
- Campo: cae C(14)
- Campo: fchvenccae C(8)
- Campo: resultado C(1)
- Campo: motivosobs M
- Campo: errcode C(6)
- Campo: errmsg M
- Campo: reproceso C(1)
- Campo: emisiontip C(4)
- Campo: fechaservd C(8)
- Campo: fechaservh C(8)

Tributo (tributo.dbf)

Detalle de otros tributos:

- Campo: tiporeg N(1,0)
- Campo: tributoid N(16,0)
- Campo: desc C(100)
- Campo: baseimp N(15,2)
- Campo: alic N(15,2)
- Campo: importe N(15,2)

Iva (iva.dbf)

Detalle de subtotales por alícuota de IVA:

- Campo: tiporeg N(1,0)
- Campo: ivaid N(16,0)
- Campo: baseimp N(15,2)
- Campo: importe N(15,2)

Comprobante Asociado (comproba.dbf)

Comprobante Asociado:

- Campo: tiporeg N(1,0)
- Campo: tipo N(3,0)
- Campo: ptovta N(4,0)

- Campo: nro N(8,0)

Opcionales (opcional.dbf)

Datos opcionales (adicionales según cada nueva RG 3411, 3668, 4004-E, etc.):

- Campo: tiporeg N(1,0)
- Campo: opcionalid C(4)
- Campo: valor C(250)

Compradores (comprado.dbf)

Datos para Venta de bienes muebles registrables (RG 4109-E)

- Campo: tiporeg N(1,0)
- Campo: doctipo N(3,0)
- Campo: docnro N(80,0)
- Campo: porcentaje N(6,2)

Factura electrónica mercado interno matrix (WSMTXCA)

El formato utilizado es similar al formato de texto de ancho fijo explicado anteriormente, y cada tipo de registro se almacena en una tabla separada.

Tabla Encabeza.dbf: Encabezado (datos generales de la factura):

- tiporeg N(1,0)
- fechacbte C(10)
- tipocbte N(2,0)
- puntovta N(4,0)
- cbtdesde N(8,0)
- cbthasta N(8,0)
- concepto N(1,0)
- tipodoc N(2,0)
- nrodoc N(11,0)
- imptotal N(15,2)
- imptotconc N(15,2)
- impneto N(15,2)
- impsubtota N(15,2)
- imptrib N(15,2)
- impopex N(15,2)
- monedaid C(3)
- monedactz N(10,6)
- fechavencp C(10)
- fechaservd C(10)
- fechaservh C(10)
- cae N(14,0)
- fchvenccae C(10)
- resultado C(1)
- motivosobs M
- errcode C(6)

- errmsg M
- reproceso C(1)
- emisiontip C(4)
- observacio M

Tabla Tributo.dbf : Detalle de otros tributos:

- tiporeg N(1,0)
- tributoid C(3)
- desc C(100)
- baseimp N(15,2)
- alic N(15,2)
- importe N(15,2)

Tabla Iva.dbf : Detalle de subtotales por alícuota de IVA:

- tiporeg N(1,0)
- ivaid C(3)
- baseimp N(15,2)
- importe N(15,2)

Tabla Comproba.dbf : Comprobante Asociado:

- tiporeg N(1,0)
- tipo N(3,0)
- ptovta N(4,0)
- nro N(8,0)

Tabla Detalles.dbf : Detalles del item:

- tiporeg N(1,0)
- umtx N(10,0)
- codmtx C(30)
- codigo C(30)
- qty N(15,3)
- umed N(3,0)
- precio N(15,3)
- bonif N(15,3)
- ivaid N(3,0)
- impiva N(15,2)
- impsubtota N(15,2)
- ds M

Factura electrónica exportación (WSFEXv1)

El formato utilizado es similar al formato de texto de ancho fijo explicado anteriormente, y cada tipo de registro se almacena en una tabla separada.

Encabezado (encabeza.dbf)

- Campo: tiporeg N(1,0)
- Campo: fechacbte C(8)
- Campo: tipocbte N(2,0)
- Campo: puntovta N(4,0)
- Campo: cbtenro N(8,0)
- Campo: tipoexpo N(1,0)
- Campo: permisoexi C(1)
- Campo: paisdstcmp N(3,0)
- Campo: nombreclie C(200)
- Campo: cuitpaiscl N(11,0)
- Campo: domicilioc M
- Campo: idimpositi C(50)
- Campo: imptotal N(15,2)
- Campo: monedaid C(3)
- Campo: monedactz N(10,2)
- Campo: obscomerci M
- Campo: obsgeneral M
- Campo: formapago C(50)
- Campo: incoterms C(3)
- Campo: incotermsd C(20)
- Campo: idiomacbte C(1)
- Campo: cae N(14,0)
- Campo: fechavto C(8)
- Campo: resultado C(1)
- Campo: reproceso C(1)
- Campo: motivosobs M
- Campo: id N(15,0)
- Campo: fchvenccae C(8)
- Campo: excepcion C(100)
- Campo: errcode C(100)
- Campo: errmsg M

Detalle (detalle.dbf)

- Campo: tiporeg N(1,0)
- Campo: codigo C(50)
- Campo: qty N(12,2)
- Campo: umed N(2,0)
- Campo: precio N(12,2)
- Campo: importe N(13,2)
- Campo: bonif N(12,2)
- Campo: ds M

Permiso (permiso.dbf)

- Campo: tiporeg N(1,0)
- Campo: idpermiso C(16)
- Campo: dstmerc N(3,0)

Comprobante Asociado (comproba.dbf)

- Campo: tiporeg N(1,0)
- Campo: cbtetipo N(3,0)
- Campo: cbtepuntov N(4,0)
- Campo: cbtenro N(8,0)
- Campo: cbtecuit N(11,0)

Formato JSON

Factura electrónica mercado interno (WSFEv1)

Para lenguajes modernos (PHP, JAVA, JavaScript, etc.) se pueden utilizar archivos de intercambio en formato JSON.

Ejecutar `rece1.exe /json`

Ejemplo archivo de entrada `faactura_electronica.json`:

```
[
    {
        "tipo_cbte": 3, "punto_vta": 4002, "cbt_desde": 447, "cbt_hasta": 447,
        "fecha_cbte": "20190511",
        "tipo_doc": 80, "nro_doc": "30500010912",

        "concepto": 1, "fecha_venc_pago": null, "fecha_vto": "20190521",

        "imp_neto": "100.00",
        "imp_iva": "21.00",
        "imp_op_ex": "0.00",
        "imp_tot_conc": "0.00",
        "imp_trib": "1.00",
        "imp_total": "122.00",

        "moneda_id": "PES", "moneda_ctz": "1.000",

        "ivas": [
            {
                "iva_id": 5, "base_imp": 100, "importe": 21
            }
        ],
       "tributos": [
            {
                "tributo_id": 99, "alic": 1, "base_imp": 100, "importe": 1,
                "desc": "Impuesto Municipal Matanza"
            }
        ],
        "cbtes_asoc": [
            {
                "nro": 1234, "pto_vta": 2, "tipo": 1
            }
        ],
        "opcionales": [],
        "compradores": []
    }
]
```

La resupuesta del webservice de AFIP es agregada al mismo formato (ejemplo editado):

```
[
    {
        "resultado": "A",
        "cae": "69193742426586",
        "fch_venc_cae": "20190521", "emision_tipo": "CAE",
        "motivos_obs": "",
        "err_code": "",  "err_msg": "",
        "reproceso": ""
    }
]
```

Forma de uso

Llamar al ejecutable RECE.EXE (factura electrónica normal, WSFE), RECEB.EXE (factura electrónica Bienes de Capital, WSBFE), RECEX.EXE (factura electrónica Exportación), RECE1.EXE (factura electrónica mercado interno WSFE versión 1) en la carpeta de la intefase (C:\PYAFIPWS)

En caso de ejecución correcta, informara por pantalla los ID y CAE obtenidos y el código de retorno es 0:

```
ID: 1000200000063 CAE: 59033567692234 Motivo: 00 Reproceso: N
ID: 1000200000064 CAE: 59033693441244 Motivo: 00 Reproceso: N
ID: 1000200000065 CAE: 59033880026543 Motivo: 00 Reproceso: N
```

El CAE obtenido, fecha de vencimiento y demás valores devueltos por WSFE (resultado, reproceso, motivo de rechazo u observación) son escritos en el archivo de salida, con la misma información de la factura que en el archivo de entrada.

En el caso de error, informa por pantalla el motivo y el código de retorno es distinto de 0:

ns1:coe.notAuthorized Computador no autorizado a acceder los servicio de AFIP

1000 FERecuperaLastCMPRequest@SRV_RA: Usuario no autorizado a realizar esta operacion. ValidacionDeToken: No apareció CUIT en lista de relaciones: 30000000000

Parámetros

RECE.EXE, RECEB.EXE, RECEX.EXE, RECE1.EXE y RECEM.EXE reciben los siguientes argumentos por línea de comando:

- /ayuda: lista los parámetros habilitados
- /prueba: Teniendo los certificados instalados, se puede realizar una prueba donde la interface generará un archivo de entrada para las tres próximas facturas, obteniendo los últimos números de transacción y comprobante.
- /ult: Solicita Tipo de comprobante y Punto de Venta y devuelve el último numero de comprobante registrado
- /dummy: consulta estado de servidores (deberían ser OK los 3 servidores)
- /ptosventa: devuelve los puntos de venta habilitados para emitir facturas electrónicas (solo WSFEv1/RECE1)
- /debug: modo depuración (detalla y confirma las operaciones)
- /formato: muestra el formato de los archivos de entrada/salida

- /get: recupera datos de un comprobante autorizado previamente (verificación, solo para RECEX, RECE1 y RECEM)
- /xml: almacena los requerimientos y respuestas XML (útil para depuración y registro)
- /dbf: modo de trabajo con tablas Dbase III/FoxPro o similares (solo RECE1 y RECEM)
- /json: modo de trabajo con formato JSON (JavaScript?, PHP, Java) (solo RECE1)

Se puede especificar como primer parámetro un nombre de archivo RECE.INI alternativo, para cargar distintas configuraciones, por ej:

```
RECE1.EXE rece-empresax.ini ....
```

Si no se especifica accion, por defecto se envía la información del archivo de intercambio para autorizar la emisión de factura electrónica, devuelve el Código de Autorización Electrónico (CAE) y demás datos que responde AFIP.

Para CAEA (CAE Anticipado) aplican las siguientes opciones (RECEM/RECE1):

- /solicitarcaea: obtiene un CAEA y su respectivo período de vigencia (fecha de validez desde y fecha de validez hasta)
- /consultarcaea: consulta la información correspondiente a un CAEA previamente otorgado (recibe período/orden o CAEA)
- /informarcaeanoutilizado, /informarcaeanoutilizadoptovta: registra en AFIP un CAEA no utilizado
- /informarcaea: envía la información del comprobante emitido y asociado a un CAEA

Ejemplo RECE1 para autorizar una factura electrónica

Al ejecutar la herramienta sin parámetros, se autoriza la factura en el archivo de entrada, guardando el resultado de AFIP en el archivo de salida:

```
C:\PYAFIPWS> RECE1.EXE

Procesando FAC A 4002 00002067 00002067 CUIT 30628789661 $ 122.00 IVA: $ 21.00
NRO: 2067 Resultado: A CAE: 65253261906700 Obs:  Err:  Reproceso: S
```

Ante cualquier dificultad, agregar los parámetros --trace /debug para ver los mensajes de depuración y xml detallados.

Ejemplo RECE1 consulta ultimo comprobante

Ejemplo para obtener el último número de factura emitida:

```
C:\PYAFIPWS> RECE1.EXE /ult
Consultar ultimo numero:
Tipo de comprobante: 19
Punto de venta: 2
Ultimo numero:  62
```

Si se pasa el tipo de comprobante y punto de venta por linea de comando (luego de `/ult`), en el archivo de salida guardará el dato devuelto por AFIP:

```
C:\PYAFIPWS> RECE1.EXE /ult 1 4001
Consultar ultimo numero:
```

```
Ultimo numero:  222
```

Ejemplo RECE1 consulta de comprobante

Es posible recuperar los datos de un comprobante indicando /get y tipo de comprobante, punto de venta y número de comprobante:

```
C:\PYAFIPWS> RECE1.EXE /get 1 5 478
Recuperar comprobante:
FechaCbte =  20150703
CbteNro =  478
PuntoVenta =  5
TipoDoc =  80
NroDoc =  27269434894
ImpTotal = 2268.75
CAE =  65273821773505
Vencimiento =  20150713
EmisionTipo =  CAE
```

Es posible agregar /xml y/o --trace para ver el mensaje enviado y recibido desde AFIP, con datos completos. Los datos del comprobante recuperado también se guardarán en el archivo de intercambio (salida).

Ejemplo RECE1 consulta puntos de venta

Ejemplo para consultar los puntos de venta habilitados por AFIP:

```
C:\PYAFIPWS> RECE1.EXE /ptosventa

=== Puntos de Venta ===
3|EmisionTipo:CAE|Bloqueado:N|FchBaja:NULL
4|EmisionTipo:CAE|Bloqueado:N|FchBaja:NULL
```

Ejemplo RECEX1 consulta monedas con cotización

Retorna la cotización a la fecha consultada de la base de datos aduanera de todas las monedas que posean cotización a la fecha indicada.

```
C:\PYAFIPWS> RECEX1.EXE /monctz
Fecha (AAAAMMDD):
|024|1.757370|2/20/2019|
|030|10.991355|2/20/2019|
|034|2.833349|2/20/2019|
|009|39.777268|2/20/2019|
|012|10.663978|2/20/2019|
|028|21.977839|2/20/2019|
|014|6.073274|2/20/2019|
|023|0.012042|2/20/2019|
|018|30.185896|2/20/2019|
|019|0.358860|2/20/2019|
|032|0.012749|2/20/2019|
|DOL|39.67|2/20/2019|
|016|4.295104|2/20/2019|
|026|28.471159|2/20/2019|
|033|0.060760|2/20/2019|
|035|11.955998|2/20/2019|
|063|27.265191|2/20/2019|
|015|4.659463|2/20/2019|
```

```
|060|45.1564|2/20/2019|
|062|0.605559|2/20/2019|
|010|2.068376|2/20/2019|
|029|0.006557|2/20/2019|
|011|1.214078|2/20/2019|
|021|52.0470|2/20/2019|
|025|0.381832|2/20/2019|
|031|5.761801|2/20/2019|
|061|5.902659|2/20/2019|
0: OK
```

Entorno

Algunas versiones de COBOL cambian el directorio u otras variables del entorno (codificación, zona horaria, etc.). En este caso se recomienda armar un archivo RECE.BAT:

```
@ECHO OFF
REM limpiar la zona horaria y cambiar al directorio de la interfase
SET TZ=
CD C:\PYAFIPWS
RECE.EXE
```

PyFEPDF: generador de PDF de facturas electrónicas

La herramienta PyFEPDF permite generar archivos PDF con el contenido de las Facturas Electrónicas, para ponerlos a disposición de los clientes o ser enviados electrónicamente vía email, con cualquier servicio de correos o por web.

Utiliza un enfoque similar a las interfaces para webservices, por lo que su uso es muy simple, totalmente automatizado (sin intervención del usuario) y no necesita herramientas externas (ni impresoras PDF ni tipografías para el código de barras).

Cubre toda la operatoria de Factura Electrónica según AFIP dada la Resolición General 2485/2006 y normativa relacionada (RG1415/03 y RG1361), aplicable a:

- merado interno (WSFEv1 y WSMTXCA, incluyendo importación, con y sin detalle)
- exportación (WSFEX)
- bono fiscal electrónico (WSBFE)

Ver muestra básica en factura.pdf

La interfaz maneja automáticamente:

- Logotipos e información particular de cada empresa
- Diseño en PDF libre* modificable (incluye diseñador visual fácil e intuitivo)
- Paginado y transporte, con múltiples copias (Original, Duplicado, etc.)
- Textos ilimitados* (detalles, observaciones, etc)
- IVA discriminado (subtotales facturas A)
- Código de barras Interleaved 2 of 7 con dígito verificador (reemplazado por código QR)
- Código QR (Obligatorio a partir de Marzo 2021 - ver cronograma de implementación)
- Metadatos en las propiedades del PDF para búsquedas y archivamiento
- Impresión directa* sin intervención del usuario

Nota *: algunas características pueden no estar disponible dado limitaciones del lenguaje de programación y/o sistema operativo.

De manera similar a las otras interfaces y herramientas, el instalador de PyFEPDF para homologación (evaluación y pruebas) está disponible en la sección Descargas

Importante:' PyFEPDF es una herramienta adicional independiente del resto de los webservices y herramientas, y los costos del soporte comercial varían dependiendo de las necesidades para implementarla. Ver: Costos y Condiciones

Aclaraciones PDF

- **Código de Barras:** es recomendable generarlo, especialmente si se imprime (dependiendo del caso sería obligatorio). Este componente contempla internamente la generación directa de la imágen de manera vectorial (escalable en cualquier resolución) según el formato que requiere AFIP (*Interleaved 2 of 5*), y no es necesario utilizar ninguna tipografia ni cuestión adicional. Simplemente debe completarse el CUIT emisor, punto de venta, tipo de comprobante, CAE y fecha de vencimiento del CAE (esta última en formato AFIP AAAAMMDD). El componente contempla también el cálculo del dígito verificador. De no estar todos los datos correctos, no se generará el código de barras. Para información general ver nuestro componente auxiliar PyI25
- **Observaciones AFIP:** Si esta presente `motivos_obs` (respuesta de AFIP en `WSFEv1.Obs` o similar) debería agregarse la leyenda *"El IVA discriminado no puede computarse como Crédito Fiscal..." (RG2485/08 Art. 27)* -esto está contemplado internamente en el componente-. Por ej., un mensaje de observación de AFIP sería: *"Factura individual, DocTipo: 80, DocNro 30000000007 no se encuentra registrado en los padrones de AFIP."*. Anteriormente AFIP solo informaba el código (09 en este caso), para más información ver WSFE F136
- **Discriminación de IVA**: el programador es responsable de enviar los datos correctos según las diferentes normativas de AFIP. Generalmente el importe de IVA solo se debe discriminar para facturas clase A o M. En las facturas clase B por lo general solo se muestra la tasa, pero no el neto ni IVA liquidado. En las facturas clase C y E generalmente no se muestra la columna de IVA ni NETO. Esta lógica esta contemplada internamente pero es responsabilidad de la aplicación que utilice el componente el revisar por la consistencia de los datos. Salvo en la Factura A, para el resto se debe indicar el precio unitario IVA Incluido (de corresponder), y el importe total de cada artículo también debería ser IVA Incluido en todos los casos (de manera similar a como se emiten los comprobantes por el sitio de clave fiscal de AFIP). Recomendamos utilizar el campo `imp_subtotal` para evitar diferencias de calculo y redondeo (debería ser la suma del neto + importes no gravado y exento para comprobantes clase A, e incluir el importe de IVA para los comprobantes clase B).
- **Descuentos**: si bien se provee un campo `descuento` para los importes generales, es recomendable que se utilice un artículo de bonificación general en el detalle, con código de unidad de medida `umed=99` y el importe en negativo, para un correcto cálculo de los subtotales (especialmente si la factura se extiende por varias páginas) y contemplar la normativa de almacenamiento/información y formatos de otros webservices como WSMTXCA, WSBFEv1 y WSFEXv1.
- **Logotipo AFIP:** en las últimas actualizaciones (1.07q), se incluye el logotipo de AFIP (similar al que se incluye en las facturas emitidas por clave fiscal). Para que muestre la leyenda "Comprobante Autorizado", debe establecerse el campo `resultado = "A"` y no debe ser un instalador de homologación. Próximamente se agregarán más validaciones en este sentido (la constatación del comprobante es responsabilidad de la aplicación que utilice el componente, ver WSCDC). Ver ejemplos y formatos a continuación para mayor información.
- **Requisitos formato de Factura:**ver http://www.afip.gov.ar/afip/259IIcap1b.htm

Interfaz COM PyFEPDF.DLL

PyFEPDF.DLL es una interfaz COM (Automatizacion/ActiveX) similar a un Control OCX, pero que no necesita un formulario visual, por lo que se puede usar desde desde cualquier lenguaje compatible en Windows (VB, VFP, Delphi, etc.). También puede utilizarce desde aplicaciones como MS Excel, MS Access, etc.

Objeto COM PyFEPDF: propiedades y métodos

El objeto COM se crea invocando a CreateObject("PyFEPDF").

Las propiedades del objeto son:

- **`CUIT`**: cuit del emisor
- **`Version`**: versión de la herramienta (ej. 1.01b)
- **`InstallDir`**: directorio de instalación

Los métodos generales del objeto para crear una factura son los siguientes (ver ejemplo en VB a continuación para más detalles):

- **`CrearFactura(concepto, tipo_doc, nro_doc, tipo_cbte, punto_vta, cbte_nro, imp_total, imp_tot_conc, imp_neto, imp_iva, imp_trib, imp_op_ex, fecha_cbte, fecha_venc_pago, fecha_serv_desde, fecha_serv_hasta, moneda_id, moneda_ctz, cae, fecha_vto_cae, id_impositivo, nombre_cliente, domicilio_cliente, pais_dst_cmp, obs_comerciales, obs_generales, forma_pago, incoterms, idioma_cbte, motivos_obs)`**: recibe toda la información del encabezado de una factura según la legislación vigente.
- **`AgregarDetalleItem(u_mtx, cod_mtx, codigo, ds, qty, umed, precio, bonif, iva_id, imp_iva, importe, despacho)`**: agrega un item de detalle de artículo. Adicionalmente, puede pasarse los parámetros `dato_a`, `dato_b`, `dato_c`, `dato_d`, `dato_e`. `despacho` también es opcional.
- **`AgregarIva(iva_id, base_imp, importe)`**: agrega un subtotal de iva (por cada alícuota)
- **`AgregarTributo(tributo_id, Desc, base_imp, alic, importe)`**: agrega otros impuestos (uno por cada ingresos brutos, retenciones y percepciones, etc.)
- **`AgregarCmpAsoc(tipo, pto_vta, nro)`**: agrega un comprobante asociado a la factura.

Una vez creada la factura con todos los items, se debe especificar el formato mediante los siguientes métodos (ver ejemplo en VB a continuación para más detalles):

- **`CargarFormato(archivo_csv)`**: crea todos los campos del diseño de la factura (layout) leyendolos desde el archivo especificado.
- **`AgregarCampo(nombre, tipo, x1, y1, x2, y2, font, size, bold, italic, underline, foreground, background, align, text, priority)`**: agrega un campo manualmente al diseño de la factura (layout)
- **`AgregarDato(campo, valor, pagina)`**: con campo = "nombre" usado en plantilla, el valor para cada hoja, y en pagina el nro de página o "T" todas, "P" primera, "U" ultima.

Luego se debe crear la plantilla que será rellenada con los datos de la factura y por último se generará y mostrará el PDF (ver ejemplo en VB a continuación para más detalles):

- **`CrearPlantilla(papel, orientacion)`**: crea una plantilla con el papel (A4, legal o letter) y orientación (portrait, landscape)
- **`ProcesarPlantilla(num_copias, lineas_max, qty_pos)`**: procesa los datos de la factura dentro de la plantilla, indicando la cantidad de copioas (1: original, 2: duplicado, 3:triplicado), la cantidad de líneas máximas por página.
- **`GenerarPDF(salida)`**: genera el archivo PDF terminado con el nombre dado en salida
- **`MostrarPDF(salida, imprimir)`**: muestra el contenido del PDF generado (usando Adobe Acrobat Reader o similar) y opcionalmente lo envía directo a la impresora.

Atributos/Propiedades específicos de configuración:

- **`CUIT`**: cuit del emisor (para generar el código de barras)
- **`FmtCantidad`**: formato numérico para cantidades, por ej. "0.4" (4 decimales)
- **`FmtPrecio`**: formato numérico para precios unitarios, por ej. "0.2" (2 decimales)
- **`Locale`**: configuración regional a utilizar para formatear importes y fechas, por ej. "Spanish_Argentina.1252" (Windows), "es_AR.utf8" (Linux)

Atributos genéricos (similares a otros componentes):

- **`Version`**: versión de la interfase (ej. ?1.07q?)
- **`InstallDir`**: directorio de instalación (ej. `C:\Archivos de Programa\PyAfipWs`)
- **`Excepcion`**, **`Traceback`**: mensaje de error y traza de rastreo (para depuración)
- **`LanzarExcepciones`**: establece si se deben emitir errores al lenguaje de programación (habilitado por defecto), o serán controlados por el programa (revisando el atributo Excepcion luego de cada método)

Ver Formato de Intercambio PDF y los otros componentes a los webservices para información adicional sobre cada campo.

Ejemplo PyFEPDF en VB

A modo de demostración a continuación se detalla un ejemplo simple (lenguaje Visual Basic), comentando el funcionamiento básico de la interfaz para generación de Facturas Electrónicas en formato PDF:

```
' Crear objeto interface para generación de F.E. en PDF
Set PyFEPDF = CreateObject("PyFEPDF")
Debug.Print PyFEPDF.Version
Debug.Print PyFEPDF.InstallDir

' CUIT del emisor
PyFEPDF.CUIT = "33693450239"

' establezco formatos (cantidad de decimales):
PyFEPDF.FmtCantidad = "0.4"
PyFEPDF.FmtPrecio = "0.2"

' Datos del encabezado de la factura:
tipo_cbte = 1       ' Factura A
punto_vta = 40000    ' prefijo
cbte_nro = 12345678 ' número de factura
fecha = "20110327"
concepto = 3
' datos del cliente:
tipo_doc = 80: nro_doc = "30000000007"
```

```
nombre_cliente = "Joao Da Silva"
domicilio_cliente = "Rua 76 km 34.5 Alagoas"
pais_dst_cmp = 16 ' código para exportación
id_impositivo = "PJ54482221-l"
' totales del comprobante:
imp_total = "122.00": imp_tot_conc = "0.00"
imp_neto = "100.00": imp_iva = "21.00"
imp_trib = "1.00": imp_op_ex = "0.00": imp_subtotal = "100.00"
fecha_cbte = fecha: fecha_venc_pago = fecha
' Fechas del período del servicio facturado
fecha_serv_desde = fecha: fecha_serv_hasta = fecha
moneda_id = "PES": moneda_ctz = "1.000"
obs_generales = "Observaciones Generales, texto libre"
obs_comerciales = "Observaciones Comerciales, texto libre"
moneda_id = "012"
moneda_ctz = 0.5
forma_pago = "30 dias"
incoterms = "FOB" ' termino de comercio exterior para exportación
idioma_cbte = 1   ' idioma para exportación (no usado por el momento)
motivos_obs = "11" ' motivo de observación (F136 y otros)

' Código de Autorización Electrónica y fecha de vencimiento:
' (para facturas tradicionales, no imprimir el CAE ni código de barras)
cae = "61123022925855"
fecha_vto_cae = "20110320"

' Creo la factura (internamente en la interfaz)
ok = PyFEPDF.CrearFactura( _
    concepto, tipo_doc, nro_doc, tipo_cbte, punto_vta, _
    cbte_nro, imp_total, imp_tot_conc, imp_neto, _
    imp_iva, imp_trib, imp_op_ex, fecha_cbte, fecha_venc_pago, _
    fecha_serv_desde, fecha_serv_hasta, _
    moneda_id, moneda_ctz, cae, fecha_vto_cae, id_impositivo, _
    nombre_cliente, domicilio_cliente, pais_dst_cmp, _
    obs_comerciales, obs_generales, forma_pago, incoterms, _
    idioma_cbte, motivos_obs)

' Parametros adicionales del encabezado:

ok = PyFEPDF.EstablecerParametro("localidad_cliente", "Hurlingham")
ok = PyFEPDF.EstablecerParametro("provincia_cliente", "Buenos Aires")

' Leyenda "Comprobante Autorizado" (verificar previamente con WSCDC o similar)

ok = PyFEPDF.EstablecerParametro("resultado", "A")

' Agregar comprobantes asociados (si es una NC/ND):
'tipo = 19
'pto_vta = 2
'nro = 1234
'pyfepdf.AgregarCmpAsoc(tipo, pto_vta, nro)

' Agrego subtotales de IVA (uno por alicuota)
iva_id = 5      ' código para alícuota del 21%
base_imp = 100  ' importe neto sujeto a esta alícuota
importe = 21    ' importe liquidado de iva
ok = PyFEPDF.AgregarIva(iva_id, base_imp, importe)

' Agregar cada impuesto (por ej. IIBB, retenciones, percepciones, etc.):
tributo_id = 99         ' codigo para 99-otros tributos
Desc = "Impuesto Municipal Matanza"
base_imp = "100.00"     ' importe sujeto a este tributo
```

```
alic = "1.00"           ' alicuota (porcentaje) de este tributo
importe = "1.00"        ' importe liquidado de este tributo
ok = PyFEPDF.AgregarTributo(tributo_id, Desc, base_imp, alic, importe)

' Agrego detalles de cada item de la factura:
u_mtx = 123456              ' unidades
cod_mtx = 1234567890123#    ' código de barras
codigo = "P0001"            ' codigo interno a imprimir (ej. "articulo")
ds = "Descripcion del producto P0001"
qty = 2                     ' cantidad
umed = 7                    ' código de unidad de medida (ej. 7 para "unidades")
precio = 100                ' precio neto (A) o iva incluido (B)
bonif = 0                   ' importe de descuentos
iva_id = 5                  ' código para alícuota del 21%
imp_iva = 42                ' importe liquidado de iva
importe = 242               ' importe total del item
despacho = "N° 123456"      ' numero de despacho de importación
ok = PyFEPDF.AgregarDetalleItem(u_mtx, cod_mtx, codigo, ds, qty, umed, _
        precio, bonif, iva_id, imp_iva, importe, despacho)

ds = "Descuento general"
umed = 99
precio = -100
imp_iva = -21
importe = -121
ok = PyFEPDF.AgregarDetalleItem("", "", "", ds, 0, umed, _
         precio, 0, iva_id, imp_iva, importe)

' Agrego datos adicionales fijos:
ok = PyFEPDF.AgregarDato("logo", PyFEPDF.InstallDir + "\fpdf.png")
ok = PyFEPDF.AgregarDato("EMPRESA", "Empresa de Prueba")
ok = PyFEPDF.AgregarDato("MEMBRETE1", "Direccion de Prueba")
ok = PyFEPDF.AgregarDato("MEMBRETE2", "Capital Federal")
ok = PyFEPDF.AgregarDato("ObservacionesGenerales1", "Nota al pie1")

' Cargo el formato desde el archivo CSV (opcional)
' (carga todos los campos a utilizar desde la planilla)
ok = PyFEPDF.CargarFormato(PyFEPDF.InstallDir + "\factura.csv")

' Agrego campos manualmente (opcional):
nombre = "prueba": tipo = "T"           ' "T" texto, "L" lineas, "I" imagen, etc.
X1 = 50: Y1 = 150: X2 = 150: Y2 = 255   ' coordenadas (en milimetros)
Font = "Arial": Size = 20: Bold = 1: Italic = 1: Underline = 1 ' tipo de letra
foreground = "000000": background = "FFFFFF"    ' colores de frente y fondo
Align = "C" ' Alineación: Centrado, Izquierda, Derecha
prioridad = 2 ' Orden Z, menor prioridad se dibuja primero (para superposiciones)
Text = "¡prueba!"
ok = PyFEPDF.AgregarCampo(nombre, tipo, X1, Y1, X2, Y2, _
                    Font, Size, Bold, Italic, Underline, _
                    foreground, background, _
                    Align, Text, priority)

' completo campos personalizados de la plantilla:
ok = PyFEPDF.AgregarDato("custom-nro-cli", "Cod.123")
ok = PyFEPDF.AgregarDato("custom-pedido", "1234")
ok = PyFEPDF.AgregarDato("custom-remito", "12345")
ok = PyFEPDF.AgregarDato("custom-transporte", "Camiones Ej. SRL")

' Creo plantilla para esta factura (papel A4 vertical):
papel = "A4" ' o "letter" para carta, "legal" para oficio
orientacion = "portrait" ' o landscape (apaisado)
ok = PyFEPDF.CrearPlantilla(papel, orientacion)
```

```
num_copias = 3  ' original, duplicado y triplicado
lineas_max = 24 ' cantidad de linas de items por página
qty_pos = "izq" ' (cantidad a la izquierda de la descripción del artículo)
' Proceso la plantilla
ok = PyFEPDF.ProcesarPlantilla(num_copias, lineas_max, qty_pos)
' Genero el PDF de salida según la plantilla procesada
salida = "factura.pdf"
ok = PyFEPDF.GenerarPDF(salida)

' Abro el visor de PDF y muestro lo generado
' (es necesario tener instalado Acrobat Reader o similar)
imprimir = False ' cambiar a True para que lo envie directo a la impresora
ok = PyFEPDF.MostrarPDF(salida, imprimir)
```

Ver pyfepdf.bas (ejemplo completo)

Herramienta PyFEPDF.EXE por línea de comando (consola)

La herramienta dispone de un programa `PyFEPDF.EXE` que puede ser utilizado desde la consola y generar facturas en formato PDF dado archivos de entada en texto o DBF.

Los parámetros son:

- --ayuda: este mensaje
- --licencia: muestra la licencia del programa
- --debug: modo depuración (detalla y confirma las operaciones)
- --formato: muestra el formato de los archivos de entrada/salida
- --prueba: genera y autoriza una factura de prueba (no usar en producción!)
- --cargar: carga un archivo de entrada (txt o dbf) con los datos del comprobante
- --grabar: graba un archivo de salida (txt o dbf) con los datos de los comprobantes
- --pdf: genera la imágen de factura en PDF
- --dbf: utiliza tablas DBF en lugar del archivo de entrada TXT

Ver rece.ini para parámetros de configuración.

Ejemplos:

1. Ver el formato de archivos de entrada:

   ```
   PYFEPDF.EXE --formato
   ```
2. Realizar una prueba (genera `factura.txt` y `factura.pdf`):

   ```
   PYFEPDF.EXE --prueba --mostrar --grabar
   ```
3. Realizar una prueba de DBF (genera `encabeza.dbf`, `detalles.dbf`, `iva.dbf`, `permiso.dbf`, `tributo.dbf`, `cbteasoc.dbf`, `dato.dbf` y `factura.pdf`):

   ```
   PYFEPDF.EXE --prueba --dbf --mostrar --grabar
   ```
4. Cargar archivo de entrada (leer `factura.txt` y generar `factura.pdf`), mostrando la factura:

   ```
   PYFEPDF.EXE --cargar --mostrar
   ```
5. Cargar tablas DBF (leer `encabeza.dbf`, `detalles.dbf`, `iva.dbf`, `permiso.dbf`, `tributo.dbf`, `cbteasoc.dbf`, `dato.dbf` y generar `factura.pdf`), imprimiendo la factura:

```
PYFEPDF.EXE --cargar --dbf --mostrar --imprimir
```

NOTA: dependiendo de la compilación del instalador, puede ser requerido ejecutar **`PYFEPDF_cli.EXE`** que esta preparado para ser ejecutado con interfaz de línea de comandos (consola/terminal, tanto en primer como segundo plano); de lo contrario podrá aparecer el mensaje *"This program host a COM object and is started automatically...(or maybe you want /register or /unregister)"* dado que el ejecutable genérico `PYFEPDF.EXE` puede estar compilado para ser registrado en Windows (componente para VB/VFP y similares).

Formato de Intercambio PDF

Para lenguajes donde no es posible utilizar objetos COM, como en algunas versiones de COBOL, se desarrolló una interfase de texto para poder generar PDF con esta herramienta. Adicionalmente, la herramienta para generación de PDF soporta manejo por tablas DBF. Estas tablas pueden ser generadas con cualquier librería o aplicación compatible (dBase, FoxPro, Clipper, incluso pueden ser abiertas con planillas de cálculo).

Los nombres de los campos y funcionamiento son los mismos entre los métodos y archivos de texto, pero recortados por compatibilidad con DBF.

En este caso se puede utilizar un único archivo de texto (`factura.txt`) o varias tablas DBF (ver método relacionado a la interfaz COM para más información). Los tipos de registros son:

- 0 para encabezado (`encabeza.dbf`) - método `CrearFactura` y `EstablecerParametro`
- 1 para detalles (`detalle .dbf`) - método `AgregarDetalleItem`
- 2 para permisos (`permisos.dbf`) - método `AgregarPermiso`
- 3 para comprobantes asociados (`comproba.dbf`) - método `AgregarCmpAsoc`
- 4 para iva (`iva.dbf`) - método `AgregarIva`
- 5 para tributos (`tributo.dbf`) - método `AgregarTributo`
- 6 para datos opcionales (AFIP) -no utilizado-
- 9 para datos adicionales (`datos.dbf`) -sólo PDF- - método `AgregarDato`

A continuación se detalla el formato de cada registro y una breve descripción:

Encabezado PDF

Nombre	Tipo	Long.	Pos.	DBF	Descripción
tipo_reg	Numerico	1	1	tiporeg N(1,0)	valor: 0 (encabezado)
webservice	Alfanumerico	6	2	webservice C(6)	"wsfe", "wsbfe", "wsfex", etc.
fecha_cbte	Alfanumerico	8	8	fechacbte C(8)	ej. "20100308"
tipo_cbte	Numerico	2	16	tipocbte N(2,0)	ej. 1 para Facturas A (según tabla de parámetros de AFIP: Mercado Interno / Exportación)
punto_vta	Numerico	5	18	puntovta N(4,0)	ej. 0001
cbte_nro	Numerico	8	23	cbtenro N(8,0)	ej. 00000001

tipo_expo	Numerico	1	31	tipoexpo N(1,0)	tipo de exportación (1: bienes, 2: servicios, 3: otros)
permiso_existente	Alfanumerico	1	32	permisoexi C(1)	permiso de exportaión: 'S' o 'N' o nulo
pais_dst_cmp	Numerico	3	33	paisdstcmp N(3,0)	código de país, según tabla de parámetros de AFIP (200: 'ARGENTINA', 203: 'BRASIL', 212: 'ESTADOS UNIDOS', etc.)
nombre_cliente	Alfanumerico	200	36	nombreclie C(200)	
tipo_doc	Numerico	2	236	tipodoc N(2,0)	código de tipo de documento según tabla AFIP WSFEv1 (80: CUIT)
nro_doc	Numerico	11	238	nrodoc N(11,0)	número de documento (DNI, CUIT, etc.) -o corresponde a cuit_pais_cliente en exportación-
domicilio_cliente	Alfanumerico	300	249	domicilioc M	
id_impositivo	Alfanumerico	50	549	idimpositi C(50)	categoría frente al IVA (Resp. Inscripto, Monotributo, etc.) o identificación impositiva en país destino (por ej. CNJP, RUC, VAT ID, etc.)
imp_total	Importe	15.3	599	imptotal N(15,3)	importe total (con hasta 3 decimales)
imp_tot_conc	Importe	15.3	614	imptotconc N(15,3)	importe conceptos no gravados (con hasta 3 decimales)
imp_neto	Importe	15.3	629	impneto N(15,3)	importe neto sujeto a cálculo de IVA (con hasta 3 decimales)
impto_liq	Importe	15.3	644	imptoliq N(15,3)	importe de IVA liquidado (con hasta 3 decimales)
impto_liq_nri	Importe	15.3	659	imptoliqnr N(15,3)	importe de IVA liquidado a RNI (con hasta 3 decimales) -obsoleto-
imp_op_ex	Importe	15.3	674	impopex N(15,3)	importe de operaciones exentas de IVA (con hasta 3 decimales)
impto_perc	Importe	15	689	imptoperc N(15,2)	importe de percepciones (con hasta 3 decimales) -obsoleto, ver tributos-
imp_iibb	Importe	15.3	704	impiibb N(15,3)	mporte de ingresos brutos (con hasta 3 decimales) -obsoleto, ver tributos-
impto_perc_mun	Importe	15.3	719	imptopercm N(15,3)	importe de percepciones municipales (con hasta 3 decimales) -obsoleto, ver tributos-
imp_internos	Importe	15.3	734	impinterno N(15,3)	importe de impuestos internos (con hasta 3 decimales) -obsoleto, ver tributos-
imp_trib	Importe	15.3	749	imptrib N(15,3)	importe total de impuestos y tributos (no incluye IVA)
moneda_id	Alfanumerico	3	764	monedaid C(3)	código de moneda según tabla de parámetros de AFIP ('PES': peso, 'DOL': dólares, etc.)
moneda_ctz	Importe	10.6	767	monedactz N(10,6)	cotización de la moneda (con hasta 6 decimales)
obs_comerciales	Alfanumerico	1000	777	obscomerci M	observaciones comerciales

obs_generales	Alfanumerico	1000	1777	obsgeneral M	observaciones generales
forma_pago	Alfanumerico	50	2777	formapago C(50)	descripción de la forma de pago (ej. 'Efectivo')
incoterms	Alfanumerico	3	2827	incoterms C(3)	término de comercio exterior (ej. 'FOB')
incoterms_ds	Alfanumerico	20	2830	incotermsd C(20)	descripción de termino de comercio exterior ('Free on board')
idioma_cbte	Alfanumerico	1	2850	idiomacbte C(1)	idioma del comprobante según tabla de parámetros de AFIP (1: español, etc.)
zona	Alfanumerico	5	2851	zona C(5)	-no utilizado-
fecha_venc_pago	Alfanumerico	8	2856	fechavencp C(8)	fecha de vencimiento del pago (solo servicios)
presta_serv	Numerico	1	2864	prestaserv N(1,0)	prestación de servicio ('S' o 'N')
fecha_serv_desde	Alfanumerico	8	2865	fechaservd C(8)	fecha de inicio del servicio facturado
fecha_serv_hasta	Alfanumerico	8	2873	fechaservh C(8)	fecha de finalización del servicio facturado
cae	Alfanumerico	14	2881	cae C(14)	Código de Autorización Electrónico otorgado por AFIP
fecha_vto	Alfanumerico	8	2895	fechavto C(8)	fecha de vencimiento del CAE otorgado por AFIP
resultado	Alfanumerico	1	2903	resultado C(1)	resultado 'A': aceptado, 'R': rechazado (respuesta AFIP)
reproceso	Alfanumerico	1	2904	reproceso C(1)	'S' si hubo reprocesamiento, 'N' si es procesamiento original (respuesta AFIP)
motivos_obs	Alfanumerico	1000	2905	motivosobs M	código o descripción del motivo de rechazo u observación del comprobante, ej: 11 - numeración, 02: cuit no autorizada (respuesta AFIP).
id	Numerico	15	3905	id N(15,0)	identificador secuencial de la factura (usar 0)
telefono_cliente	Alfanumerico	50	3920	telefonocl C(50)	
localidad_cliente	Alfanumerico	50	3970	localidadc C(50)	
provincia_cliente	Alfanumerico	50	4020	provinciac C(50)	
formato_id	Numerico	10	4070	formatoid N(10,0)	-no utilizado-
email	Alfanumerico	100	4080	email C(100)	destinatario del PDF
pdf	Alfanumerico	100	4180	pdf C(100)	ruta y nombre de archivo PDF a generar
err_code	Alfanumerico	6	4280	errcode C(6)	código de error informado por AFIP (Ej. fexerror.ErrCode='505')
err_msg	Alfanumerico	1000	4286	errmsg M	mensaje de error informado por AFIP (Ej. fexerror.ErrMsg='Error de Lockeo')

Dato_adicional1	Alfanumerico	30	5286	datoadic01 C(30)	datos extra optativos
Dato_adicional2	Alfanumerico	30	5316	datoadic02 C(30)	datos extra optativos
Dato_adicional3	Alfanumerico	30	5346	datoadic03 C(30)	datos extra optativos
Dato_adicional4	Alfanumerico	30	5376	datoadic04 C(30)	datos extra optativos

Campos extra (por compatibilidad con otras herramientas y formatos):

Nombre	Tipo	Long.	Pos.	DBF	Descripción
descuento	Importe	15.3	5405	descuento N(15,3)	importe general de descuento (no recomendado, usar descuento por item/detalle)
cbt_desde	Numerico	8	5420	cbtdesde N(8,0)	número de comprobante inicial del lote (usar cbte_nro)
cbt_hasta	Numerico	8	5428	cbthasta N(8,0)	número de comprobante final del lote (usar cbte_nro)
concepto	Numerico	1	5436	concepto N(1,0)	1 productos, 2 servicios, 3 productos + servicios (ver presta_serv)
no_usar	Importe	15.3	5437	nousar N(15,3)	-reservado-
imp_iva	Importe	15.3	5452	impiva N(15,3)	ídem impto_liq
emision_tipo	Alfanumerico	4	5467	emisiontip C(4)	CAE o CAEA
imp_subtotal	Importe	15.3	5471	impsubtota N(15,3)	se calcula internamente si no es pasado: imp_neto + imp_tot_conc + imp_op_ex - descuento (subtotal sin IVA ni otros tributos)
cat_iva	Numerico	2	5486	cativa N(2,0)	-reservado-

NOTA: estos campos pueden no utilizarse, o tener el mismo valor que los campos originales; están para contemplar casos especiales y en general no deben utilizarse.

Detalle Item PDF

Nombre	Tipo	Long.	Pos.	DBF	Descripción
id	Numerico			id N(15,0)	identificador secuencial de la factura (*clave foránea*) - usar 0, no presente en el TXT -
tipo_reg	Numerico	1	1	tiporeg N(1,0)	valor 1
codigo	Alfanumerico	30	2	codigo C(30)	código del artículo
qty	Importe	12.2	32	qty N(12,2)	cantidad (con 2 decimales)
umed	Numerico	2	44	umed N(2,0)	código de unidad de medida según tabla de parámetros AFIP WSMTXCA (ej. 1 - kg)
precio	Importe	12.3	46	precio N(12,3)	importe unitario (con hasta 3 decimales)

importe	Importe	14.3	58	importe N(14,3)	importe subtotal (con hasta 3 decimales)
iva_id	Numerico	5	72	ivaid N(5,0)	código de alícuota de IVA según tabla de parámetros AFIP WSFEv1 / WSMTXCA (ej. 5 - IVA RI Tasa General 21 %)
ds	Alfanumerico	4000	77	ds M	descripción del artículo (usar " " para salto de linea)
ncm	Alfanumerico	15	4077	ncm C(15)	código habilitado del Nomenclador Común de Mercosur -sólo WSBFEv1-
sec	Alfanumerico	15	4092	sec C(15)	código habilitado de la Secretaría de Comercio -reservado-
bonif	Importe	15	4107	bonif N(15,2)	importe de descuento (con 2 decimales) para este articulo
imp_iva	Importe	15	4122	impiva N(15,2)	importe liquidado de IVA para este articulo
despacho	Alfanumerico	20	4137	despacho C(20)	-sólo importación-
u_mtx	Numerico	10	4157	umtx N(10,0)	unidades MTX
cod_mtx	Alfanumerico	30	4167	codmtx C(30)	código de barras MTX
dato_a	Alfanumerico	15	4197	datoa C(15)	dato extra optativo
dato_b	Alfanumerico	15	4212	datob C(15)	dato extra optativo
dato_c	Alfanumerico	15	4227	datoc C(15)	dato extra optativo
dato_d	Alfanumerico	15	4242	datod C(15)	dato extra optativo
dato_e	Alfanumerico	15	4257	datoe C(15)	dato extra optativo

Nota: en los campos numéricos (archivo de texto), para usar más de los decimales especificados de manera predeterminada, indicar el punto explicitamente, por ej. `cantidad = "0.123456"` para 6 decimales en cantidad. En la estructura del DBF, cada programa puede cambiar la cantidadad de decimales.

Importante: para Facturas B, de manera similar a las facturas tradicionales, el IVA no debe estar discriminado (enviar el precio unitario y subtotal final, IVA incluido), y se debe enviar NULL en `imp_iva`

IVA PDF

Nombre	Tipo	Long.	Pos.	DBF	Descripción
id	Numerico			id N(15,0)	identificador secuencial de la factura (*clave foránea*) - usar 0, no presente en el TXT -
tipo_reg	Numerico	1	1	tiporeg N(1,0)	valor 4
iva_id	Numerico	5	2	ivaid N(5,0)	tipo de alícuota (ver tablas de parámetros de WSFEv1, por ej. 5 para 21%)
base_imp	Importe	15.3	7	baseimp N(15,3)	base imponible (neto gravado) para esta alícuota
importe	Importe	15.3	22	importe N(15,3)	importe IVA liquidado para esta alícuota

Tributo PDF

Nombre	Tipo	Long.	Pos.	DBF	Descripción
id	Numerico			id N(15,0)	identificador secuencial de la factura (*clave foránea*) - usar 0, no presente en el TXT -
tipo_reg	Numerico	1	1	tiporeg N(1,0)	valor 5
tributo_id	Numerico	5	2	tributoid N(5,0)	tipo de tributo (ver tablas de parámetros de WSFEv1, por ej. 2 para impuestos provinciales)
desc	Alfanumerico	100	7	desc C(100)	descripción (por ej. 'IIBB prov. de Bs.As.')
base_imp	Importe	15.3	107	baseimp N(15,3)	base imponible para este impuesto (pude ser 0 / nulo)
alic	Importe	15	122	alic N(15,2)	alícuota (porcentaje, por ej. 3%)
importe	Importe	15.3	137	importe N(15,3)	importe liquidado para este impuesto

Comprobante Asociado PDF

Nombre	Tipo	Long.	Pos.	DBF	Descripción
id	Numerico			id N(15,0)	identificador secuencial de la factura (*clave foránea*) - usar 0, no presente en el TXT -
tipo_reg	Numerico	1	1	tiporeg N(1,0)	valor 3
cbte_tipo	Numerico	3	2	cbtetipo N(3,0)	tipo de comprobante asociado (ídem encabezado)
cbte_punto_vta	Numerico	4	5	cbtepuntov N(4,0)	punto de venta
cbte_nro	Numerico	8	9	cbtenro N(8,0)	número de comprobante

Permisos PDF

Nombre	Tipo	Long.	Pos.	DBF	Descripción
id	Numerico			id N(15,0)	identificador secuencial de la factura (*clave foránea*) - usar 0, no presente en el TXT -
tipo_reg	Numerico	1	1	tiporeg N(1,0)	valor 2
id_permiso	Alfanumerico	16	2	idpermiso C(16)	identificador del permiso de exportación, ej: '99999AAXX999999A'
dst_merc	Numerico	3	18	dstmerc N(3,0)	código de país de destino según tabla de AFIP

Datos PDF

Nombre	Tipo	Long.	Pos.	DBF	Descripción
tipo_reg	Numerico	1	16	tiporeg N(1,0)	valor 9
campo	Alfanumerico	30	17	campo C(30)	nombre del campo en la plantilla PDF
valor	Alfanumerico	1000	47	valor M	contenido para el campo en la plantilla PDF
pagina	Alfanumerico	3	1047	pagina C(3)	'P': primera, 'U': ultima, 'T': todas

Este tipo de registro es opcional, resultados similares se pueden lograr con el archivo de configuración (rece.ini, sección [PDF]), método del Componente `AgregarDato` o directamente con el diseñador visual (modificando la plantilla factura.csv).

Configuración PyFEPDF: RECE.INI

De manera similar a las otras herramientas, PyFEPDF puede configurarse por el archivo de configuración `RECE.INI` con las siguientes opciones predeterminadas:

```
[FACTURA]
ARCHIVO=tipo,letra,numero
FORMATO=factura.csv
DIRECTORIO=.
PAPEL=letter
ORIENTACION=portrait
DIRECTORIO=.
SUBDIRECTORIO=
LOCALE=Spanish_Argentina.1252
FMT_CANTIDAD=0.6
FMT_PRECIO=0.3
CANT_POS=izq
ENTRADA=factura.txt
SALIDA=factura.pdf

[PDF]
LOGO=fpdf.png
EMPRESA=Empresa de Prueba
MEMBRETE1=Direccion de Prueba
MEMBRETE2=Capital Federal
CUIT=CUIT 30-00000000-0
IIBB=IIBB 30-00000000-0
IVA=IVA Responsable Inscripto
INICIO=Inicio de Actividad: 01/04/2006

[DBF]
Encabezado = encabeza.dbf
Tributo = tributo.dbf
Iva = iva.dbf
Comprobante Asociado = cbteasoc.dbf
Detalle = detalles.dbf
Permiso = permiso.dbf
Dato = dato.dbf
```

Sección [FACTURA]

Controla la generación del PDF (si no se especifica campo PDF en tabla encabezado):

- ARCHIVO: nombre de archivo PDF a generar, por ej. incluyendo tipo,letra,numero
- FORMATO: formato del PDF a generar, por ej. en planilla factura.csv
- DIRECTORIO: directorio donde almacenar los archivos PDF.
- SUBDIRECTORIO: nombre de la carpeta a generar (una por día de manera predeterminada)
- LINEAS_MAX: cantidad de lineas para el detalle (artículos), por defecto 24
- PAPEL: legal, A4, letter
- ORIENTACION: portrait, landscape (apaisada)
- LOCALE: configuración regional ("Spanish_Argentina.1252" en Windows, "es_AR.utf8" en Linux)
- FMT_CANTIDAD: formato de cantidad (0.4 para 4 decimales)
- FMT_PRECIO: formato de precio unitario (0.3 para 3 decimales)
- CANT_POS: posición de la columna cantidad (izq o der)
- ENTRADA: nombre del archivo de intercambio (factura.txt)

- SALIDA: nombre del archivo a generar (dejar en blanco para generar uno distinto según la configuración ARCHIVO)
- COPIAS: cantidad de copias en el PDF (Original, Duplicado, Triplicado, etc.)

Sección [PDF]

Controla campos del PDF a personalizar (si no se especifican los campo texto en tabla formato del PDF):

- LOGO: archivo con imagen logotipo, por ej. serpiente.png
- EMPRESA: ej. Empresa de Prueba
- MEMBRETE1: ej. Direccion de Prueba
- MEMBRETE2: ej. Capital Federal
- CUIT: CUIT 20-26756539-3
- IIBB: IIBB 20-26756539-3
- IVA: IVA Responsable Inscripto
- INICIO: Inicio de Actividad: 01/04/2006

Sección [DBF]

Configuración referente a las tablas DBF, por cada una se especifica la ruta y el nombre de archivo, ej:

- Encabezado = encabeza.dbf
- Tributo = tributo.dbf
- Iva = iva.dbf
- Comprobante Asociado = cbteasoc.dbf
- Detalles = detalles.dbf
- Dato = dato.dbf

Campo Subtotal

Dado que en general la columna importe debería mostrar "Subtotal c/IVA", por similitud con el PDF que genera AFIP por comprobantes en linea, por diseño (actualización "1.08b"):

- Si es A o M, se imprime el subtotal SIN IVA
- Si es B o C, se imprime el subtotal CON IVA

En el caso de factura A, debe sumar los items sin el IVA ya que en la ultima hoja se le suma el IVA al subtotal.

Igualmente el campo "subtotal" podría "sobre-escribirse" utilizando el método `AgregarDato(campo, valor, pagina)` o el tipo de registro para datos adicionales. En campo = "subtotal", el valor a imprimir en cada hoja, y pagina el nro de página ("T" todas, "P" primera, "U" ultima) Se podría agregar el dato para todas las paginas vacio, y para la última con el valor que sea necesario (en ese orden).

Diseñador Visual PyFEPDF

La herramienta incluye el progama `designer.exe` para modificar visualmente los diseños de factura.

Los archivos de entrada están son planillas CSV describiendo el diseño. Al abrilos el diseñador muestra la plantilla con los elementos donde se ubicarán los futuros valores de cada factura.

La barra de herramientas cuenta con botones para:

- Abrir, guardar e imprimir (vista previa) las plantillas
- Agregar, eliminar y duplicar elementos
- Buscar elementos por nombre o por texto

Sobre un elemento, con doble click izquierdo se abre un dialogo para editar el texto, y con click derecho se abre un dialogo con la ventana de propiedades. Es posible seleccionar múltiples elementos manteniendo presionado shift y click izquierdo. Para mover los elementos se puede usar las teclas de flechas o arrastrar los elementos.

Ver a continuación los tipos de elementos y su definición.

A modo de ejemplo se muestra un pantallazo del Diseñador Visual, con el elemento logo seleccionado, editando sus propiedades:

Para ejecutarlo, simplemente ir al directorio de la interfaz y correr `designer.exe`

Tipos de elementos para PyFEPDF

Para el diseño del PDF es posible establecer los siguientes tipos de campo (elementos gráficos):

- Texto (T), con tipo de letra (fuente), tamaño, formato (italico, negrita, subrayado), tamaño y alineación
- Líneas (L) y Cuadros (B)
- Imágenes (I) en formato PNG o JPG
- Código de Barras (CB) en formato Entrelazado 2 de 5
- Codigo QR (I) (Requerido por AFIP a partir del 01/03/2021, ver cronograma)

Definición de elementos para PyFEPDF

Las columnas de la planilla (factura.csv) o estructura de la tabla formato para definir los elementos gráficos es:

- name (alfanumérico): nombre del campo
- type (alfanumérico): tipo del campo: T, L, I, B, CB
- x1 (numérico): coordenada horizontal izquierda (en mm)
- y1 (numérico): coordenada vertical superior (en mm)
- x2 (numérico): coordenada horizontal derecha (en mm)
- y2 (numérico): coordenada vertical inferior (en mm)
- font (alfanumérico): nombre de la tipografía (fuente)
- size (numérico): tamaño en puntos del texto
- bold (verdadero/falso): estilo negrita para el texto (1 o 0)
- italic (verdadero/falso): estilo cursiva (itálica) para el texto (1 o 0)
- underline (verdadero/falso): estilo subrayado para el texto (1 o 0)
- foreground (numérico): color RGB de dibujo
- backgroud (numérico): color RGB de relleno del fondo (si aplica al elemento)
- align (alfanumérico): alineación: I: izquierda, D: derecha, C: centrado
- text (alfanumérico): texto estático o fórmula
- priority (numérico): orden z en el que se dibujan los elementos (a menor prioridad se dibuja primero)

Estilos HTML: Adicionalmente, los estilos negrita, itálica y subrayado pueden establecerse en tiempo de ejecución, encerrando todo el texto con el tag html correspondiente (en el orden indicado). Ej:

- `<B>texto en negrita</B>`
- `<I>texto en cursiva</I>`
- `<U>texto subrayado</U>`
- `<B><I><U>texto en negrita, cursiva y subrayado</U></I></B>`

PyI25: Generador de Códigos de Barras

Utilitario `PYI25.EXE` (`py25.py`) para generar los códigos de barras Interleaved 2 of 5 (Entrelazado 2 de 5) según Resolución General 1702/04.

Ver Descargas para descargar el instalador.

Los datos a consignar son (ver RG1702/04 Art. 1 Anexo 1):

- Clave Unica de Identificación Tributaria (C.U.I.T.) del emisor de la factura (11 caracteres)
- Código de tipo de comprobante (3 caracteres)
- Punto de venta (5 caracteres)
- Código de Autorización de Electrónica (C.A.E.) o Código de Autorización de Impresión (C.A.I.) (14 caracteres)
- Fecha de vencimiento del CAE / CAI (8 caracteres)
- Dígito verificador (1 carácter), ver rutina

El utilitario genera una imágen en formato PNG o JPG, funciona tanto de línea de comando / consola (DOS) como por interfaz COM (automatización, DLL similar a OCX), y es útil para facturas electrónicas o tradicionales (autoimpresores o imprentas).

Imágen de Ejemplo de Código de Barras para CAE:

Importante: De generar imágenes para imprimir, es recomendado respetar la resolución de la impresora (Láser o Chorro de tinta, no pueden usarse matriz de punto), por ej., con una resolución por lo menos tres veces mayor que la utilizada en pantalla.

Nota: Nuestro generador de PDF ya contempla generar los códigos de barra para cualquier resolución de impresión y pantalla, no es necesario generar imágenes.

PYI25.EXE (utilitario por linea de comando)

Opciones:

- --barras: a continuación se indica el código de barras en formato numérico
- --noverificador: deshabilitar cálculo de digito verificador
- --archivo: a continuación se indica el nombre del archivo de salida

Ejemplo de uso (generar el archivo prueba.png):

```
PYI25.EXE --barras 20267565393024001612030347390422011052 8 --mostrar --archivo prueba.png
```

NOTA: dependiendo de la compilación del instalador, puede ser requerido ejecutar **`PYI25_cli.EXE`** que esta preparado para ser ejecutado con interfaz de línea de comandos (consola/terminal, tanto en primer como segundo plano); de lo contrario podrá aparecer el mensaje *"This program host a COM object and is started automatically...(or maybe you want /register or /unregister)"* dado que el ejecutable genérico `PYI25.EXE` puede estar compilado para ser registrado en Windows (componente para VB/VFP y similares).

Objeto PYI25 (interfaz COM)

Objeto: PyI25 (registrar PYI25.DLL)

Métodos:

- **`DigitoVerificadorModulo10(codigo)`**: recibe el codigo de barras en formato numérico, devuelve el dígito verificador en base 10
- **`GenerarImagen(codigo, archivo, anchobase, ancho, alto, extension )`**: recibe el codigo de barras en formato numérico, nombre de archivo de salida, tamaños y formato de imagen (PNG o JPEG) y genera el codigo de barras especificado.

Ejemplo de uso (interfaz COM desde visual Basic):

```
Dim PyI25 As Object

Set PyI25 = CreateObject("PyI25")

' cuit, tipo_cbte, punto_vta, cae, fch_venc_cae
barras = "20267565393024001612030347390422011052 9"
' calculo digito verificador:
barras = barras + PyI25.DigitoVerificadorModulo10(barras)

' genero imagen en png, aspecto 1x para ver en pantalla o por mail
ok = PyI25.GenerarImagen(barras, "C:\barras.png")


' formato en jpg, aspecto 3x más ancho para imprimir o incrustar:
ok = PyI25.GenerarImagen(barras, "c:\barras.jpg", 9, 0, 90, "JPEG")
```

PyQR: Generador de Códigos QR

Utilitario `PYQR.EXE` (`pyqr.py`) para generar los códigos QR según RG4892/2020.

Cronograma de Implementación:

Para responsables inscriptos en el impuesto al valor agregado que hayan efectuado operaciones (gravadas, exentas y no gravadas) declaradas en el impuesto durante el año calendario 2020 por un importe total neto de impuestos y tasas:

1.1. Superior a PESOS DIEZ MILLONES ($10.000.000.-): desde el 1 de marzo de 2021.

1.2. Superior a PESOS DOS MILLONES ($2.000.000.-) e inferior o igual a PESOS DIEZ MILLONES ($ 10.000.000.-): desde el 1 de abril de 2021.

1.3. Superior a PESOS QUINIENTOS MIL ($500.000.-) e inferior o igual a PESOS DOS MILLONES ($ 2.000.000.-): desde el 1 de mayo de 2021.

2. Para el resto de los responsables inscriptos en el impuesto al valor agregado, sujetos exentos ante dicho gravamen y pequeños contribuyentes inscriptos en el Régimen Simplificado (Monotributo): a partir del 1 junio de 2021.

Los datos a consignar son (ver QR especificaciones):

- Fecha de emisión
- CUIT del emisor
- Punto de venta
- Tipo de Comprobantes
- Número de Comprobante
- Importe total
- Moneda
- Cotización
- Tipo Documento Receptor (de corresponder)
- Número de Documento de Receptor (de corresponder)
- Código del Tipo de Autorización
- Código de Autorización

Ejemplo Visual Basic script: pyafipws/blob/develop/ejemplos/pyqr/pyqr.vbs

Ejemplo Visual Basic clásico

```
Dim pyqr As Object

Set pyqr = CreateObject("PyQR")

' Establecer tipo de imagen (PNG o JPEG):
PyQR.Extension = "JPEG"

' Generar un nombre de archivo temporal para la imagen:
archivo = pyqr.CrearArchivo

' Alternativo: Especificar ubicación y extensión
pyqr.archivo = "C:\QRgenerado\pyqr.jpg"

Debug.Print archivo

ver = 1
fecha = "2020-10-13"
cuit = 30000000007#
pto_vta = 10
tipo_cmp = 1
nro_cmp = 94
importe = 12100
moneda = "DOL"
ctz = 65#
tipo_doc_rec = 80
nro_doc_rec = 20000000001#
tipo_cod_aut = "E"
cod_aut = 70417054367476#

' genero imagen en png con el codigo qr
```

```
url = pyqr.GenerarImagen(ver, fecha, cuit, pto_vta, tipo_cmp, nro_cmp, _
                         importe, moneda, ctz, tipo_doc_rec, nro_doc_rec, _
                         tipo_cod_aut, cod_aut)

Debug.Print url
```

Parámetros por línea de comandos:

- --datos: recibe los datos a procesar version, fecha, cuit, pto_vta, tipo_cmp, nro_cmp, importe, moneda, ctz, tipo_doc_rec, nro_doc_rec, tipo_cod_aut, cod_aut
- --prueba: genera un QR de muestra
- --archivo: permite especificar una ruta (de lo contrario usa un archivo temporal)
- --mostrar: abre el archivo para visualizar la imagen

Ejemplo por línea de comando:

*Linux:

```
python pyqr.py --datos 1 2020-10-13 30000000007 10 1 94 12100 PES 1 80 20000000001 E 70417054367476
```

*Windows:

```
pyqr.exe --archivo qr.jpg --datos 1 2020-10-13 30000000007 10 1 94 12100 PES 1 80 20000000001 E 70417054367
```

En windows usar pyqr.exe en vez de python pyqr.py

Imágen de Ejemplo de Código QR:

Nota: Nuestro generador de PDF ya contempla generar los códigos QR para cualquier resolución de impresión y pantalla, no es necesario generar imágenes.

PyEmail: Envio de correo electrónico

Herramienta para enviar correos electrónicos, incluyendo por ej, adjunto PDF de con factura electrónica.

Soporta servidores SMTP con y sin autenticación.

Próximamente soporte para correos con contenido HTML e imagenes de factura embebida

PYEMAIL.EXE: utilitario de correo

Para utilizar esta herramienta debe tener configurado la sección [EMAIL] en el `rece.ini`

Ejemplo desde linea de comando:

```
PYEMAIL.EXE "motivo" "reingart@gmail.com" "mensaje" FacturaA0002-00117826.pdf
```

NOTA: dependiendo de la compilación del instalador, puede ser requerido ejecutar **`PYEMAIL_cli.EXE`** que esta preparado para ser ejecutado con interfaz de línea de comandos (consola/terminal, tanto en primer

como segundo plano); de lo contrario podrá aparecer el mensaje *"This program host a COM object and is started automatically...(or maybe you want /register or /unregister)"* dado que el ejecutable genérico `PYEMAIL.EXE` puede estar compilado para ser registrado en Windows (componente para VB/VFP y similares).

Objeto PyEmail (interfaz COM)

Objeto: PyEmail (registrar PyEmail.DLL)

Métodos:

- Conectar(servidor, usuario, clave): recibe el nombre del servidor a conectarse, usuario y contraseña (opcional)
- Enviar(remitente, motivo, destinatario, mensaje, archivo): envia el correo electrónico con los datos proporcionados

Ejemplo de uso (interfaz COM desde visual Basic):

```
Dim PyEmail As Object

Set PyEmail = CreateObject("PyEmail")

' Primer paso: conexión al servidor (por unica vez)
servidor = "mail.sistemasagiles.com.ar"
usuario = "no.responder@nsis.com.ar"
clave = "1238478"
puerto = "465"
ok = PyEmail.Conectar(servidor, usuario, clave)

' Envio el o los correos (repetir por cada FE)
remitente = "no.responder@sistemasagiles.com.ar"
destinatario = "mariano@sistemasagiles.com.ar"
mensaje = "Se envia factura electronica adjunta"
archivo = "C:\FACTURA.PDF"

ok = PyEmail.Enviar(remitente, motivo, destinatario, mensaje, archivo)
```

Régimen de Almacenamiento de Duplicados Digitales (RG1361)

Con la interfaz se provee un utilitario para generar los archivos requeridos por el aplicativo SIRED (SIAP) de la Resolución General 1361/02, referente al almacenamiento digital de los comprobantes emitidos (Libro Ventas, Detalle y Cabeceras de Factura).

El formato de entrada es una planilla csv similar al PyRece, respetando los tipos de datos y códigos requeridos por los webservices, con las siguientes columnas: tipo_cbte, punto_vta, cbt_numero, fecha_cbte, tipo_doc, nro_doc, imp_neto, impto_liq, imp_total, cae, fecha_vto, nombre.

Ejemplo de uso:

```
CD C:\PYAFIPWS
RG1361.EXE planilla.csv
```

Generará los archivos:

- VENTAS_200902.txt
- CABECERA_200902.txt
- DETALLE_200902.txt

Para más información ver RG1361/02 de AFIP:

- Diseño de registro (archivos de texto)
- Guia Temática

Ejemplo

Ejemplo en Visual Basic clásico:

```
sired = CreateObject("SIRED")

' creo una factura de ejemplo
tipo_cbte = 2
punto_vta = 4000
fecha = "20150721"
concepto = 3
tipo_doc = 80
nro_doc = "30000000007"
cbte_nro = 12345678
imp_total = "122.00"
imp_tot_conc = "3.00"
imp_neto = "100.00"
imp_iva = "21.00"
imp_trib = "1.00"
imp_op_ex = "2.00"
imp_subtotal = "100.00"
fecha_cbte = fecha
fecha_venc_pago = fecha
' Fechas del período del servicio facturado (solo si concepto = 1?)
fecha_serv_desde = fecha
fecha_serv_hasta = fecha
moneda_id = "PES"
moneda_ctz = "1.000"
obs_generales = "Observaciones Generales, texto libre"
obs_comerciales = "Observaciones Comerciales, texto libre"

nombre_cliente = "Joao Da Silva"
domicilio_cliente = "Rua 76 km 34.5 Alagoas"
pais_dst_cmp = 16
id_impositivo = "Responsable Inscripto"
forma_pago = "30 dias"
incoterms = "FOB"
idioma_cbte = 1
motivo = "11"

cae = null
fch_venc_cae = null

sired.CrearFactura(concepto, tipo_doc, nro_doc, tipo_cbte, punto_vta, _
    cbte_nro, imp_total, imp_tot_conc, imp_neto, _
    imp_iva, imp_trib, imp_op_ex, fecha_cbte, fecha_venc_pago, _
    fecha_serv_desde, fecha_serv_hasta, _
```

```
    moneda_id, moneda_ctz, cae, fch_venc_cae, id_impositivo, _
    nombre_cliente, domicilio_cliente, pais_dst_cmp, _
    obs_comerciales, obs_generales, forma_pago, incoterms, _
    idioma_cbte, motivo)

' agrego comprobantes asociado:

tipo = 91
pto_vta = 2
nro = 1234
sired.AgregarCmpAsoc(tipo, pto_vta, nro)
tipo = 5
pto_vta = 2
nro = 1234
sired.AgregarCmpAsoc(tipo, pto_vta, nro)

tributo_id = 99
desc = "Impuesto Municipal Matanza"
base_imp = "100.00"
alic = "1.00"
importe = "1.00"
sired.AgregarTributo(tributo_id, desc, base_imp, alic, importe)

iva_id = 5 ' 21%
base_imp = 100
importe = 21
sired.AgregarIva(iva_id, base_imp, importe)

u_mtx = 123456
cod_mtx = 1234567890123
codigo = "P0001"
ds = "Descripcion del producto P0001"
qty = 1.00
umed = 7
precio = 100.00
bonif = 0.00
iva_id = 5
imp_iva = 21.00
importe = 121.00
despacho = u'N° 123456'
sired.AgregarDetalleItem(u_mtx, cod_mtx, codigo, ds, qty, umed, _
        precio, bonif, iva_id, imp_iva, importe, despacho)

sired.AgregarDato("prueba", "1234")

id_factura = sired.GuardarFactura()

sired.EstablecerParametro("cae", "61123022925855")
sired.EstablecerParametro("fch_venc_cae", "20110320")
sired.EstablecerParametro("motivo_obs", "")
ok = sired.ActualizarFactura(id_factura)
```

Otros Servicios WEB y herramientas similares de AFIP

Constatación de Comprobantes

WSCDC es la Interfaz para los Servicios Web para verificar en forma dinámica si los comprobantes recibidos se encuentran autorizados por la AFIP. Ver manual específico en ConstatacionComprobantes

Padrón de Contribuyentes

Herramienta para consultar el archivo completo de la condición tributaria de los contribuyentes y responsables (RG1817) Ver manual específico en PadronContribuyentesAFIP

Código de Trazabilidad de Granos

WSCTGv2 es la Interfaz para generar Código de Trazabilidad de Granos AFIP (RG2806, RG3113, RG3493). Ver manual específico en CodigoTrazabilidadGranos

Liquidación Primaria de Granos

WSLPG es la Interfaz para autorizar y ajustar Liquidaciones Primarias de Granos (RG3419/2012). Ver manual específico en LiquidacionPrimariaGranos

Consulta de Operaciones Cambiarias

WSCOC es la Interfaz para Consultas de Operaciones (moneda extranjera). Ver manual en ConsultaOperacionesCambiarias

wDigDepFiel: Depositario Fiel

Servicio Web de AFIP para Aviso de Recepción y Aceptación de la Digitalización de Documentos por parte de los Prestadores de Servicios de Archivos y Digitalización (PSAD) y Aviso de Digitalizacion de Documentos por parte de los Despachantes o de los Prestadores de Servicios de Archivos y Digitalización (PSAD) -

Sujetos Alcanzados: ver Resolución General 2570/2009

Actualmente la interfaz expone las funcionalidades de este webservice:

- Aviso de Digitalización
- Aviso de Recepción y Aceptación
- Método Dummy para verificación de funcionamiento

Ejemplo Interfaz COM wDigDepFiel

De igual manera que el resto de los servicios web, el web service de depositario fiel puede ser invocado desde lenguajes visuales y no visuales (en windows) utilizando la interfaz COM:

Ejemplo en Visual Basic (similar para Visual FoxPro?, Delphi, etc.):

```
' Crear objeto interface Web Service de Factura Electrónica de Exportación
Set wDigDepFiel = CreateObject("wDigDepFiel")
' Setear tocken y sing de autorización (pasos previos)
wDigDepFiel.token = WSAA.token
wDigDepFiel.sign = WSAA.sign

' CUIT del emisor (debe estar registrado en la AFIP)
wDigDepFiel.cuit = "20267565393"
```

```
' Conectar al Servicio Web de Depositario Fiel (homologación)
ok = wDigDepFiel.Conectar("https://testdia.afip.gov.ar/Dia/Ws/wDigDepFiel/wDigDepFiel.asmx")

' Llamo a un servicio nulo, para obtener el estado del servidor (opcional)
wDigDepFiel.Dummy
Debug.Print "appserver status", wDigDepFiel.AppServerStatus
Debug.Print "dbserver status", wDigDepFiel.DbServerStatus
Debug.Print "authserver status", wDigDepFiel.AuthServerStatus

tipo_agente = "PSAD" '"DESP"
rol = "EXTE"
nro_legajo = "0000000000000000"
cuit_declarante = "20267565393"
cuit_psad = "20267565393"
cuit_ie = "20267565393"
codigo = "000" ' carpeta completa, "001" carpeta adicional
' Fecha/Hora en formato ISO: "2010-06-07T00:23:51.750000"
fecha_hora_acept = Format(Now(), "yyyy-MM-dd") & "T" & _
                   Format(Now(), "hh:mm:ss") & ".000000"
ticket = "1234"
errCode = wDigDepFiel.AvisoRecepAcept(tipo_agente, rol, _
                     nro_legajo, cuit_declarante, cuit_psad, cuit_ie, _
                     codigo, fecha_hora_acept, ticket)
Debug.Print wDigDepFiel.XmlResponse

MsgBox wDigDepFiel.DescError, vbInformation, _
       "AvisoRecepAcept Código Error: " & wDigDepFiel.CodError

tipo_agente = "PSAD" ' "DESP"
rol = "EXTE"
nro_legajo = "0000000000000000" ' "12345678901234567"
cuit_declarante = "20267565393"
cuit_psad = "20267565393"
cuit_ie = "20267565393"
cuit_ata = "20267565393"
codigo = "000" ' carpeta completa, "001" carpeta adicional
ticket = "1234"
url = "http://www.example.com"
hashing = "db1491eda47d78532cdfca19c62875aade941dc2"

' inicializo aviso: limpio datos (familias)
wDigDepFiel.IniciarAviso
codigo = "02"
cantidad = 1
wDigDepFiel.AgregarFamilia codigo, cantidad
codigo = "03"
cantidad = 3
wDigDepFiel.AgregarFamilia codigo, cantidad

cantidad_total = 4

errCode = wDigDepFiel.AvisoDigit(tipo_agente, rol, _
                    nro_legajo, cuit_declarante, cuit_psad, cuit_ie, cuit_ata, _
                    codigo, url, ticket, hashing, cantidad_total):

Debug.Print wDigDepFiel.XmlResponse

MsgBox wDigDepFiel.DescError, vbInformation, _
       "AvisoDigit Código: " & wDigDepFiel.CodError
```

Otros Servicios Web relacionados de otros organismos

Remito Electrónico (COT ARBA)

Interfaz para Servicio Web Código de Operaciones de Traslado (COT) "Remito Electronico" correspondiente al articulo 41 del Código Fiscal que establece la obligación de amparar el traslado o transporte de bienes en el territorio de la provincia de Bs. As. Ver manual específico en RemitoElectronicoCotArba

Trazabilidad de Medicamentos (ANMAT)

Interfaz para Servicio Web Código de Trazabilidad de Medicamentos (SOAP) correspondiente a la Resolución 435/2011 del Ministerio de Salud y Disposición 3683/2011 de A.N.M.A.T.: Sistema Nacional de Trazabilidad de Medicamentos que deberán implementar las personas físicas o jurídicas que intervengan en la cadena de comercialización, distribución y dispensación de especialidades medicinales incluidas en el Registro de Especialidades Medicinales. SNT Especificación Técnica V2. Ver manual específico en TrazabilidadMedicamentos

Trazabilidad de Precursores Químicos (SEDRONAR)

Interfaz para Servicio Web Trazabilidad de Precursores Químicos TrazaMed?.SDRN (SOAP) para informar movimientos de Sustancias Químicas Controladas ? Decreto 1095/96, modificado por Decreto 1161/00: Sistema Nacional de Trazabilidad modulo de Precursores Químicos (RENPRE SEDRONAR PAMI INSSJP) que deberán implementar el operador de precursores químicos obtención de número de CUFE (Código de Ubicación Física de Establecimiento). Entrenamiento y Alineación de Datos. Resolución 900/12 RENPRE Ver manual específico en TrazabilidadPrecursoresQuimicos

Certificados

Generación

Para poder utilizar la interfase se deben tramitar y asociar los certificados de homologación/producción en la AFIP. Para mas información ver Página principal de Factura Electrónica (AFIP)

Crear Pedido CSR con WSAA

A partir de la revisón 1632 (ver Actualizaciones Factura Electrónica, módulo WSAA.py versión 1.10b) es posible generar el pedido de certificado automáticamente utilizando la opción `--crear_pedido_cert`). La herramienta solicitará el CUIT, buscará la denominación del contribuyente en el padrón de AFIP y generará los archivos necesarios (clave privada y pedido de certificado).

Ejemplo:

```
C:\PyAfipWs> WSAA-cli.exe --crear_pedido_cert
reingart@S55t-B:~/pyafipws$ python wsaa.py --crear_pedido_cert
Ingrese un CUIT: 20267565393
Denominación según AFIP: REINGART MARIANO ALEJANDRO
Longitud clave 2048 (bits)
Se crearon los archivos:
clave_privada_20267565393_201609105952.key
pedido_cert_20267565393_201609105952.csr
```

También pueden especificarse esos datos por linea de comando (CUIT, nombre/alias, empresa y longitud de la clave -2048 bits predeterminados-), ejecutando:

```
WSAA-cli.exe --crear_pedido_cert 20267565393 "PyAfipWS SHA-2" "REINGART MARIANO ALEJANDRO" 2048
```

Nota: desde el código fuente ejecutar `python wsaa.py`.
También es posible realizar el procedimiento por programa (automatizadamente), llamando a los métodos `wsaa.CrearClavePrivada(clave_privada)` y `wsaa.CrearPedidoCertificado(cuit, empresa, nombre, pedido_cert)`.

La herramienta deja listos los archivos respectivos, pudiendo subir el pedido de certificado CSR a AFIP (tanto en homologación como en producción)

WSASS

Video Explicativo completo (incluyendo WSASS autogestión de certificados en homologación AFIP):
https://www.youtube.com/watch?v=ietDXvUzVIQ

OpenSSL

Pasos para crear el certificado (más información en Instructivo AFIP):

- Bajar e instalar OpenSSL para Windows: 0.9.8i (historico) o 1.0.2k o superior (recomendado). En caso de inconvenientes, instalar Redistribuible de Visual C++
- Ingresar por línea de comando al directorio de OpenSSL C:\OpenSSL\bin>
- Generar la clave privada:

```
openssl genrsa -out empresa.key 2048
```

- Generar el pedido (CSR: certificate signing request) por línea de comando:

```
openssl req -new -key empresa.key -subj "/C=AR/O=[empresa]/CN=[nombre]/serialNumber=CUIT [nro_cuit]" -out e
```

  - ♦ Reemplazar [empresa] por el nombre de su empresa. Ej. *Nombre de la empresa tal cual figura en la consulta de inscripción, ej.:* **Empresa S A**
  - ♦ Reemplazar [nombre] por su nombre o server hostname. *Nombre del servicio, aplicación u unidad operativa, ej:* **Interfaz PyAfipWs**
  - ♦ Reemplazar [nro_cuit] por la CUIT sin guiones de la empresa o programador. *Ej.* **CUIT xxxxxxxxxxx**
- Enviar el *empresa.csr* a la AFIP para que lo firmen y devuelvan el certificado *empresa.crt*.

Ejemplo concreto:

```
ssl req -new -key empresa.key -subj "/C=AR/O=Mariano Reingart/CN=PyAfipWs/serialNumber=CUIT 20267565393" -out e
```

Generar Certificado CRT

Una vez generado el pedido de certificado CSR (ver puntos anteriores), el trámite es:

- **Homologación**: ir a "WSASS - Autogestión Certificados Homologación" (solicitud de servicio)

nuevo procedimiento a partir del 1/06/2015

- Ir a Nuevo Certificado en el WSASS
- Nombre simbólico del DN: "un alias del computador fiscal" (nombre de fantasía, ej PC1)
- Solicitud de certificado en formato PKCS#10: abrir con un editor de texto el CSR y copiar aqui el contenido
- Presionar "Crear DN y obtener certificado"
- Abrir un archivo nuevo y copiar el campo "Resultado" (este es el certificado, usar extensión .CRT)

- **Producción**: Ingresar por clave fiscal al servicio online "Administración de Certificados Digitales"
  - Ir a "Agregar Alias", escribir el alias (nombre relacionado al certificado)
  - Subir el archivo .CSR presionar "Agregar"
  - Luego buscar y descargar el certificado CRT.

- Con esta interfase no es necesario convertir el certificado en formato pkcs12 ni importarlo al repositorio de Windows

Para más información ver instructivo oficial AFIP

Asociación

Para poder usar los servicios web se debe asociar el certificado al webservice y CUIT (empresa) que lo utilizará:

- **Homologación**: ir a "WSASS - Autogestión Certificados Homologación" (solicitud de servicio)
  nuevo procedimiento a partir del 1/06/2015
  - Ir a Crear autorización a servicio en el WSASS
  - Seleecionar el "Nombre simbólico del DN a autorizar" (igual que en el paso anterior)
  - Revisar el CUIT representado y quien genera la autorización
  - Seleccionar el "Servicio al que desea acceder" (por ej "wsfe factura electrónica")
  - Presionar "Crear autorización de Acceso" y revisar el resultado
- **Producción**: en el sitio de AFIP, ingresar por clave fiscal:
  - Entrar en "Administrador de Relaciones de Clave Fiscal"
  - Elegir "Nueva Relación"
  - Presionar "BUSCAR" para seleccionar el webservice de AFIP, por ej "ws - Facturación Electrónica"
  - Presionar "BUSCAR" para seleccionar el Representante, seleccionar el nombre del alias del "Computador Fiscal" (ver sección anterior)
  - "Confirmar".

Anteriormente, en producción debía realizarse utilizando el servicio de clave fiscal "ARFE - Gestionar Relaciones", yendo a "Realizar una nueva Asociación" completando Empresa, Servicio y computador -certificado- que utilizara los webservices.

Para más información en producción ver instructivos oficiales de la AFIP:

- obtener certificado AFIP
- asociar certificado AFIP

Es el mismo trámite para la renovación (`ns1:cms.cert.expired: Certificado expirado`): debe subirse el arcihvo CSR (pedido de certificado) y descargar el nuevo CRT. Recordamos que los certificados

tienen una vigencia de 2 años en producción, y 1 año en homologación.

Pre-requisitos para Producción (RECE)

Para poder usar Factura Electrónica en Producción (modo definitivo), además de los puntos mencionados anteriormente, dentro de Clave Fiscal, servicio online "Regimenes de facturación y registración (REAR/RECE/RFI)" es necesario:

- Adherirse al régimen (realizarlo con anterioridad, este trámite puede demorar varias semanas). Importante: en el "Rubro I. C) RECE / RCEL (Factura Electrónica y Factura Electrónica en Linea)" especificar *RECE* (no RCEL que es para "Comprobantes en linea" y Facturador Plus" no contemplados por los webservices)
- Dar de alta los nuevos puntos de ventas (uno por cada webservice a utilizar: factura nacional, exportación, bono fiscal, etc.)

Para más información ver demo interactiva oficial empadronamiento AFIP

Descargas

- OpenSSL para windows
- Redistribuibe VC++

Comprobación de Autenticación (WSAA)

Para descartar problemas técnicos de acceso en producción u homologación (y verificar la validez del certificado generado), se puede utilizar la herramienta `WSAA.EXE` / `WSAA-cli.EXE` en modo depuración (sin necesidad de autorizar una factura, solo solicitar acceso) y/o el mensaje XML devuelto por `WSAA.CallWSAA()`:

```
C:\PYAFIPWS>WSAA-cli.EXE reingart.crt reingart.key wsfe 2400 https://wsaa.afip.gov.ar/ws/services/LoginCms --

Usando CERT=reingart.crt PRIVATEKEY=reingart.key
       URL=https://wsaa.afip.gov.ar/ws/services/LoginCms
       SERVICE=wsfex TTL=2400
Creando TRA...
--------------------------------------------------------------------------------
<?xml version="1.0" encoding="UTF-8"?>
<loginTicketRequest version="1.0">
 <header>
  <uniqueId>1279058341</uniqueId>
  <generationTime>2010-07-13T18:19:01</generationTime>
  <expirationTime>2010-07-13T19:39:01</expirationTime>
 </header>
 <service>wsfex</service>
</loginTicketRequest>
--------------------------------------------------------------------------------
Frimando TRA...
Llamando WSAA...
--------------------------------------------------------------------------------
POST https://wsaa.afip.gov.ar/ws/services/LoginCms
SOAPAction: "http://ar.gov.afip.dif.facturaelectronica/loginCms"
Content-length: 3369
Content-type: text/xml; charset="UTF-8"

...
```

```
================================================================================
El archivo TA.xml se ha generado correctamente.
```

Nota: Cambiar `WSAA.EXE` por `WSAA-CLI.EXE` o viceversa, dependiendo del paquete de instalación o como este compilado (si aparece el mensaje "This program host a COM object an is started automatically..."). Para usarlo desde el código fuente, utilizar ?en su lugar `python wsaa.py`

Para descartar cualquier problema técnico se puede utilizar la salida que es la traza de depuración de la comunicación con AFIP a la mesa de ayuda (requerimiento y respuesta xml).

Si se estaría conectando correctamente a producción, el certificado y las URL son correctas, etc., se debería constatar que:

- El ticket de requerimiento de acceso (TRA) correctamente solicita servicio correcto (wsfex): `<service>wsfex</service>`
- Se está solicitando acceso a los servidores del ambiente correcto (`wsaa_url`): `POST https://wsaa.afip.gov.ar/ws/services/LoginCms` (producción) o `POST http://wsaahomo.afip.gov.ar/ws/services/LoginCms` (homologación)
- El equipo que responde sería del ambiente correcto: `pereza.afip.gov.ar`, `envidia.afip.gov.ar`, `ira.afip.gov.ar` (producción) o `avaricia.afip.gov.ar` (homologación). Nota: los nombres de los servidores pueden cambiar, y solo se informan en caso de error.

Si la autenticación es exitosa (`El archivo TA.xml se ha generado correctamente.`, en el mensaje XML contenido en dicho archivo se puede verificar los siguientes datos:

- Fuente: AFIP producción (`<source>CN=wsaa, O=AFIP, C=AR, SERIALNUMBER=CUIT 33693450239</source>`), sinó diría `CN=wsaahomo, O=AFIP, C=AR, SERIALNUMBER=CUIT 33693450239`
- Destino: datos del certificado de producción/homologación (`C=ar, O=pyafipws-sistemas agiles, SERIALNUMBER=CUIT 20267565393, CN=mariano reingart`)

Si el ambiente es el correcto (producción/homologación) y sigue devolviendo acceso denegado: `ns1:coe.notAuthorized: Computador no autorizado a acceder al servicio (gov.afip.desein.dvadac.sua.view.wsaa.LoginFault)`, revisar el certificado, ARFE (asociación de servicio), RECE/REAR/RFI sobre régimen habilitado, puntos de venta, y demás trámites ante AFIP. Si aparece error `600: ValidacionDeToken: Error al verificar hash: VerificacionDeHash: No valida la firma digital`, se está usando un sign y token en el ambiente incorrecto (de producción en homologación y viceversa).

Ejemplo Interactivo de Autenticación (ej-WSAA)

De manera similar a lo expuesto anteriormente, se provee una aplicación visual a modo de ejemplo, para comprobar el funcionamiento de la interfaz y los certificados tramitados desde AFIP.

El instalador para este aplicación puede descargarse de:

- instalar-ej-wsaa-202.zip (archivado en 1GoogleCode)

Por defecto se instala en `C:\Archivos de Programa\wsaa\ej-wsaa.exe`

Importante: Es necesario configurar correctamente el certificado y clave privada con la ruta completa a dichos archivos.

Para pruebas en producción, cambiar la URL a `https://wsaa.afip.gov.ar/ws/services/LoginCms` y configurar los certificados.

La version de la interfaz no debe decir Homologación, y debe permitir cambiar la URL, sino deben reinstalar los instaladores para producción, eliminando cualquier archivo anterior.

Importante: generalmente solo debe correr el ejemplo, no instalar ni actualizar el componente WSAA si tiene una versión superior (o incluida con otro webservice, como WSFEv1 o WSLPG) o esta usando el instalador para producción. Si no lo tiene instalado, puede usar instalador-WSAA-2.02c.exe o superior.

Errores Frecuentes

Los siguientes son posibles interpretaciones y soluciones a los errores que envían los servidores de AFIP.

Fallas SOAP

Falla SOAP: ns1:coe.notAuthorized o ns1:cms.cert.untrusted

En WSAA, descartando que se este usando el certificado del ambiente incorrecto (certificado de homologación en producción y viceversa), es necesario que el certificado y CUIT estén habilitados para acceder al servicio web especificado (wsfe, wsfex, wsbfe, wsctg, wDigDepFiel, etc.). No es suficiente generar el certificado (en homologación o en producción), hay que solicitar explicitamente acceso al servicio requerido: ver Asociación de Certificado

Los servicios no relacionados a factura electrónica, en producción tienen otros métodos para asociar los certificados.

Ver Comprobación de Autenticación WSAA con información para determinar o depurar el problema de acceso.

Mensajes de Error Original:

- Computador no autorizado a acceder los servicios de AFIP (ns1:coe.notAuthorized)
- Certificado no emitido por AC de confianza (ns1:cms.cert.untrusted)

Falla SOAP: ns1:coe.alreadyAuthenticated: El CEE ya posee un TA valido para el acceso al WSN solicitado

Este error es reportado por AFIP cuando se esta solicitan varios tickets de acceso para el mismo certificado y servicio.

Ver Reutilización de Ticket de Acceso

Falla SOAP: Server was unable to read request. ---> There is an error in XML document (4, 1339). ---> Input string was not in a correct format.

Este error correponde al protocolo de intercambio de datos (ver Errores por Tipos de Datos) y puede significar que:

- Un campo obligatorio no fue completado
- Se completó un campo con un tipo de datos inválido (float en lugar de int, coma en vez de punto, etc.)
- Se especificó un valor no esperado por el web service.

En cualquier caso, se puede revisar el mensaje XML de requermiento (atributos xml_request o XmlRequest) y buscar la etiqueta con datos incorrectos en la línea y caracter donde se informa el error (por ej., en este caso, línea 4, cerca del caracter 1339). Ver más información y rutinas en Manejo de Excepciones

Otras fallas SOAP

WSAA y otros webservices pueden devolver errores no documentados o capturados por AFIP, por ej:

- `Server was unable to process request. ---> This OracleTransaction has completed; it is no longer usable.`
- `SoapFault: soapenv:Server: Could not initialize class org.hibernate.transform.Transformers.#`

Nuestra interfaz no utiliza Java ni Hibernate, por lo que este tipo de problemas son cuestiones internas de AFIP. Los errores "Oracle" u ORA son un problema interno de la base de datos de AFIP.

Puede ocurrir en homologación, y se debe esperar a que AFIP solucione el problema.

En estos casos para producción, se puede consultar con la mesa de ayuda: sri@afip.gov.ar (seguramente es un tema que lo resolverán en el día y no requiera una intervención de programación o configuración especial)

Errores generales AFIP

600: ValidacionDeToken: Error al verificar hash: VerificacionDeHash: Error al convertir de Base64 al token

Posiblemente haya un problema interno en AFIP (frecuentemente se soluciona automáticamente ya que puede ser un problema de los servidores de AFIP, ver abajo) o se este enviando incorrectamente el `WSAA.Token` (por ej. el lenguaje de programación o la base de datos recortó el string). Revisar que dicho dato se esté copiando tal cual al webservice de negocio (`WSFEv1.Token`, `WSMTXCA.Token`, etc.) o utilizar los nuevos métodos como `WSFEv1.SetTicketAcceso(ta)` que realiza este paso de manera más automatizada.

En python se puede analizar el sign ejecutando

```
import base64
print base64.b64decode(wsaa.Token)
```

Que debería mostrar un XML (token del ticket de acceso) como el siguiente:

```
="UTF-8" standalone="yes"?>
```

```
" src="CN=wsaahomo, O=AFIP, C=AR, SERIALNUMBER=CUIT 33693450239" gen_time="1401116887" exp_time="1401160147" ds
d" type="login">
oyafipws-sistemas agiles, SERIALNUMBER=CUIT 20267565393, CN=mariano reingart" service="wsfe" regmethod="22" ent

ltype="4" key="20267565393"/>
```

600: ValidacionDeToken: Error al verificar hash: VerificacionDeHash: Error al convertir de Base64 a la firma:

Posiblemente se este enviando incorrectamente el WSAA.Sign (por ej. el lenguaje de programación o la base de datos recortó el string). Revisar que dicho dato se esté copiando tal cual al webservice de negocio (WSFEv1.Sign, WSMTXCA.Sign, etc.) o utilizar los nuevos métodos como `WSFEv1.SetTicketAcceso(ta)` que realiza este paso de manera más automatizada.

600: ValidacionDeToken: Error al verificar hash: VerificacionDeHash: No valid? la firma digital.

Este error es reportado por AFIP cuando no se está enviando al webservice de negocio (WSFEv1, WSMTXCA, WSFEXv1, etc.) el `Token` y `Sign` obtenido con WSAA (o se lo envía vacío).

Esto puede deberse porque no se esté validando correctamente el ticket de requerimiento de acceso (certificado inválido, fecha inválida, ambiente de producción / homologación incorrecto, ticket de acceso ya otorgado, etc.). Revisar que la URL de cada webservice sea correcta (homologación o producción, tanto de WSAA como del servicio web de negocio: WSFEv1, WSMTXCA, etc.), y que el certificado corresponda al ambiente correcto.

Ver Manejos de excepciones para detectar el inconveniente en WSAA.

600: ValidacionDeToken: No validaron las fechas del token GenTime, ExpTime, NowUTC

No están sincronizados la fecha y hora del equipo con los servidores de AFIP.

En linux, se actualiza con:

```
sudo ntpdate time.afip.gov.ar
```

En windows, se actualiza yendo a Propiedades del Reloj, Hora de Internet, introduciendo `time.afip.gov.ar`

Otros mensajes similares:

- Token vencido fecha y hora de vencimiento del token enviado - fecha y hora actual del servidor
- Validaciondetoken No validaron las fechas del token Gen Time, Exp Time Now Utc:
- WSAA xml.generationTime.invalid, xml.expirationTime.expired, xml.expirationTime.invalid: El tiempo de generación/expiración es inválido: generationTime posee formato o dato inválido (ej: en el futuro o más de 24 hs de antiguedad)

1000 Usuario no autorizado a realizar esta operacion. ValidacionDeToken: No aparece CUIT en lista de relaciones:

En los web services de negocio (WSFE, WSFEX, WSBFE, etc.), se está accediendo con un ticket de acceso válido, pero se intenta operar con un CUIT incorrecto y/o que no se corresponde al certificado. Revisar que el CUIT sea correcto y que esté asociado al servicio y certificado (ver "ARFE" en punto anterior)

1000 Usuario no autorizado a realizar esta operacion. ValidacionDeToken: Error al verificar hash: VerificacionDeHash: No valid la firma digital.

En los web services de negocio (WSFE, WSFEX, WSBFE, etc.), se está accediendo con un ticket de acceso inválido para el ambiente requerido (por ej., ticket de acceso de homologación usado en producción), o, no se envió correctamente el tocken y sign.

Usuario no autorizado para realizar esta operación. Validacionde Token: No valido id Sistema: wsfex (Id Sistema de token es: wsfe)

En los web services de negocio (WSFE, WSFEX, WSBFE, etc.), se está accediendo con un ticket de acceso inválido para el servicio web requerido (por ej., ticket de acceso de factura nacional -wsfe- usado en exportación -wsfex-).

Se debe revisar el valor del servicio al crear el requerimiento de acceso (CreateTRA)

Problemas con los certificados

Error inesperado: no existe el archivo o directorio de certificado/clave privada

Este error corresponde a que la interfaz no encuentra los archivos del certificado/clave privada o la ruta no tiene algún directorio. Revisar la ruta completa y reintentar.

Mensaje de Error Original (certificado):

> Unexpected Python Error: <type 'exceptions.IOError'>: [Errno 2] No such file or directory: 'reingart.crt'

Mensaje de Error Original (clave privada):

> Unexpected Python Error: 936:error:02001002:system library:fopen:No such file or directory:.\crypto\bio\bss_file.c:122:fopen('reingart.key','r') 936:error:2006D080:BIO routines:BIO_new_file:no such file:.\crypto\bio\bss_file.c:125: : None

Error inesperado: La clave privada no coincide con el certificado

Este error corresponde a que el certificado no fue creado con la clave privada dada, por lo que no se puede realizar la encriptación necesaria. Revisar que la clave privada sea con la que se creó el pedido de certificado original (CSR) y reintentar. Si no se cuenta con la clave privada original, es imposible utilizar el certificado por lo que es necesario generar y asociar uno nuevo.

Mensaje de Error Original:

> Unexpected Python Error: <class 'M2Crypto.SMIME.SMIME_Error'>:
> 936:error:0B080074:x509 certificate routines:X509_check_private_key:key values
> mismatch:.\crypto\x509\x509_cmp.c:399: 936:error:2107407F:PKCS7
> routines:PKCS7_sign:private key does not match certificate:.\crypto\pkcs7\pk7_smime.c:76:

Error Inesperado: La clave privada es inválida

Mensaje de Error Original:

> Unexpected Python Error: 936:error:0906D06C:PEM routines:PEM_read_bio:no start
> line:.\crypto\pem\pem_lib.c:647:Expecting: ANY PRIVATE KEY

Este error corresponde a que la clave privada no está en el formato correcto (texto plano: PEM), seguramente el archivo se encuentra en PKCS12 (binario: certificado x509 + clave privada con extensión `.p12` o `..pfx`) lo que no es necesario para esta interfaz.

Para convertir de pkcs12 a pem utilizar:

```
openssl pkcs12 -in empresa.p12 -out empresa.pem -nodes
```

Eso generará el archivo `empresa.pem` donde:

- La clave privada (`.key`) se encuentra entre: `-----BEGIN RSA PRIVATE KEY-----` y `-----END RSA PRIVATE KEY-----`
- El certificado (`.crt`) se encuentra entre: `-----BEGIN CERTIFICATE-----` y `-----END CERTIFICATE-----`

Copiar dichas lineas en archivos separados. Los archivos deben incluir el encabezado (`-----BEGIN ...`) y pie (`-----END ...`) para ser válidos.

En caso de ser necesario, para recorrer el camino inverso, generar archivo PKCS12 desde PEM (no necesario por esta interfaz), realizar:

```
openssl pkcs12 -export -inkey empresa.key  -in empresa.crt -out empresa.p12
```

Error inesperado: Mala Desencriptación

Este error corresponde a que el certificado fue creado usando una frase de seguridad (passphrase), por lo que en tiempo de ejecución solicita entrada dicha contraseña (bloqueandose si no se proporciona).

Mensaje de Error Original:

> M2Crypto.EVP.EVPError: 3079395008:error:06065064:digital envelope
> routines:EVP_DecryptFinal_ex:bad
> decrypt:evp_enc.c:330:3079395008:error:0906A065:PEM routines:PEM_do_header:bad
> decrypt:pem_lib.c:428:

Para remover la passphrase (contraseña), por ej. del archivo `privada.pem` y grabarlo en `privada.key`, ejecutar el siguiente comando:

```
openssl rsa -in privada.pem -out privada.key
```

Error inesperado: Esperando certificado

El certificado proporcionado es inválido.

Es posible leer los datos de un certificado utilizando el comando `openssl`:

```
openssl x509 -in reingart.crt -noout -text
```

Debería mostrar la información completa del certificado:

```
 L=Ciudad Autonoma de Buenos Aires, O=Administracion Federal de Ingresos Publicos, OU=Subdireccion General de

as Agiles, C=AR
```

Si no lo muestra, posiblemente el certificado esté en otro formato (ver como convertir desde pk7 en puntos anteriores)

Mensaje de Error Original:

> M2Crypto.X509.X509Error: 3078817472:error:0906D06C:PEM routines:PEM_read_bio:no start line:pem_lib.c:650:Expecting: CERTIFICATE

Error al dar de alta computador

A partir de Agosto de 2016, aparentemente AFIP está solicitando claves más largas, pudiendose presentar el siguiente error al tratar de generar un certificado por WSASS Autoservicio de Acceso a WebServices? (**TESTING/HOMOLOGACIÓN**):

```
***ERROR*** clsCrearComputador: (CUIT=20267565393, ALIAS=test_1024).
createComputer: Error al dar de alta al computador (CUIT='20267565393', ALIAS='test_1024')
La longitud de clave pública debe ser estar comprendida entre 2048 y 8192 bits
```

En este caso de debe generar la clave con mayor cantidad de bits, por ej. indicando el parámetro 2048 a Instructivo OpenSSL

La herramienta WSAA 1.10g ya crea de manera predeterminada claves de 4096 bits, ver Crear Pedido CSR

Se estima que estas cuestiones afectarán en PRODUCCIÓN cuando venzan la mayoría de los certificados para ese entorno el **21 de Febrero de 2016**. Es posible con este componente revisar el vencimiento de cada `.crt` con los Metodos para Certificados

Excepción: SSLError X509_load_cert_crl_file

No se puede cargar el archivo "afip_ca_info.crt" de la lista de autoridades de certificación de confiaza (CACERT), necesario para verificar el canal seguro como solicita AFIP.

Ese archivo en las ultimas actualizaciones se encuentra en la carpeta "conf", o puede ser descargado desde: el repositorio

El archivo debe ser copiado a la carpeta principal donde esté el programa WSAA, o configurar el parámetro CACERT con la ruta a la ubicación completa.

Mensaje de Error Original:

> [Errno 185090050] _ssl.c:345: error:0B084002:x509 certificate routines:X509_load_cert_crl_file:system lib

Falla SOAP WSAA (SoapFault)

En ciertas ocaciones WSAA puede denegar el acceso con los siguientes mensajes:

- ns1:cms.cert.expired: Certificado expirado
- ns1:xml.bad: No se ha podido interpretar el XML contra el SCHEMA
- cms.cert.untrusted: Certificado NO emitido por un AC de confianza
- cms.sign.invalid, cms.bad y cms.bad.base64: no se esta informando correctamente el mensaje criptográfico firmado ("Firma inválida o algorítmo no soportado")

Generalmente se esta tratando de usar un certificado inválido o expirado, se está pasando mal algún parámetro (servicio o tiempo de vida). o se está tratando de acceder al ambiente equivocado (el cerificado es de producción y el servidor de homologación, o viceversa.)

Ver consulta tema anterior para revisar el certificado, Modo Homologación / Producción y Comprobación de Autenticación WSAA

Errores de Conectividad

Antes las fallas de conexión, sobrecarga, caídas de internet u otros problemas de comunicación la interfaz puede devolver los siguientes errores (en su mayoría del lenguaje de programación Python o sistema operativo):

- `AttributeError: 'NoneType' object has no attribute 'makefile'`: no se puede establecer el canal de comunicación (socket), puede que internet no este funcionando en el cliente
- `socket.gaierror: [Errno -2] Name or service not known`: no se puede resolver el nombre del equipo, revise los servidores DNS asignados a su conexión de internet.
- `socket.error: [Errno 104] Connection reset by peer`: la conexión fue cerrada por la contraparte (servidor AFIP), posiblemente por sobrecarga.
- `<urlopen (11001 'getaddrinfo failed')>`: no se puede resolver el nombre de archivo, revise los DNS y si está utiliando algún servidor intermedio (proxy)
- `ExpatError: Not well formed token`: el servidor posiblemente no esta devolviendo XML correcto debido a un error interno.

- `BadStatusLine`, `ResponseNotReady`: significa que los servidores de AFIP no contestan algo coherente (compatible con el protocolo HTTP)
- `AttributeError: Tag not Found: Body (no elements found)`: significa que los servidores de AFIP no contestan un XML válido según el protocolo de webservices SOAP
- `SSLError: The read operation timed out`: la conexión SSL/TLS supero el tiempo de espera, posiblemente es un error de conectividad de Internet o los servidores de AFIP están caídos.
- `Service Unavailable`: El servidor de AFIP no está disponible y no devuelve un mensaje XML correcto.
- `ServerNotFoundError: Unable to find the server at servicios1.afip.gov.ar`: no está resolviendo el nombre del Servidor de AFIP a la dirección IP, revisar la configuración DNS con su proveedor de Internet
- `error: [Errno 111] Connection refused`, `error: [Errno 10061] No se puede establecer una conexión ya que el equipo de destino denegó expresamente dicha conexión`: revisar que no esté mal configurado el servidor proxy (comentar o no enviar parametro si no corresponde)

En todos estos casos se recomienda reintentar la operatoria luego de unos instantes, después de verificar la conexión de internet. Se puede probar ingresando a la URL del webserice (por ej. [https://servicios1.afip.gov.ar/wsfev1/service.asmx?WSDL

WSFEv1]) con un navegador y ver si les abre correctamente la página.

También puede haber un tema de antivirus o proxy que esté bloqueando la conexión saliente. En ese caso (servidor intermedio para poder salir a internet), se debe consultar con los administradores de la red y posiblemente se necesite configurar:

- El parámetro `proxy` en el método `Conectar`, formato "usuario:clave@servidor:puerto" (ver ejemplo)
- o sección [PROXY] en el archivo rece.ini (ver ejemplo)

Ver la siguiente sección si el problema persiste. Recordar que a veces funciona o no dependiendo del lugar desde donde se conecta el cliente, ya que AFIP tiene una granja de servidores, pueden probar reconectandose a internet desde otra IP para ver si los deriva a un servidor que funcione (o esperar a que baje la demanda y/o ver si lo solucionan).

Errores de Protocolo SSL

AFIP está desafectando los protocolos de seguridad obsoletos (SSLv3, requiriendo TLSv1+), por lo que en sistemas operativos antiguos o desactualizados (por ej. Windows XP), puede existir una incompatibilidad con que imposibilite la comunicación segura; es recomendable actualizar el sistema operativo, lenguaje de programación y componentes.

- `SSLEOFError: EOF occurred in violation of protocol (_ssl.c:590)`
- `SSLHandshakeError: [SSL: SSLV3_ALERT_HANDSHAKE_FAILURE] sslv3 alert handshake failure (_ssl.c:581)`
- `[Errno 1] _ssl.c:490: error:14094410:SSL routines:SSL3_READ_BYTES:sslv3 alert handshake failure`

Como mitigación puede utilizare un transporte alternativo basado en la biblioteca CURL, que en general tiene mejor cobertura para este tipo de cuestiones en Windows.

Para ello, establecer parámetro `wrapper="pycurl"` en el método `Conectar` si se utilizan los componentes:

```
...
wrapper = "pycurl"
...
ok = WSFEv1.Conectar(cache, wsdl, proxy, wrapper)
...
```

Si se utilizan las herramientas por linea de comando, configurar en el `rece.ini`:

```
[WSFEv1]
...
wrapper = pycurl
...
```

Errores AFIP CUIT Emisor

En caso de problemas con el CUIT emisor y no superar una de las siguientes validaciones excluyentes (verificación de datos registrales, inscripción en el régimen, autorización de emisión de comprobantes, domicilio fiscal. etc.), AFIP devolverá un Código de Error 10000, con los mensajes posibles:

- 01: LA CUIT INFORMADA NO CORRESPONDE A UN RESPONSABLE INSCRIPTO EN EL IVA ACTIVO
- 02: LA CUIT INFORMADA NO SE ENCUENTRA AUTORIZADA A EMITIR COMPROBANTES ELECTRONICOS ORIGINALES O EL PERIODO DE INICIO AUTORIZADO ES POSTERIOR AL DE LA GENERACION DE LA SOLICITUD
- 03: LA CUIT INFORMADA REGISTRA INCONVENIENTES CON EL DOMICILIO FISCAL
- 04: LA CUIT INFORMADA NO SE ENCUENTRA AUTORIZADA A EMITIR COMPROBANTES CLASE "A"
- 05: EL CUIT INFORMADO COMO EMISOR NO SE ENCUENTRA REGISTRADO DE FORMA ACTIVA EN LAS BASES DE LA ADMINISTRACIÓN.
- 06: DEBE POSEER AL MENOS UNA ACTIVAD ACTIVA.

En estos casos debe contactar la mesa de ayuda de AFIP para solucionar el inconveniente que esté bloqueando la emisión de facturas electrónicas (generalmente no es un tema técnico sino un trámite administrativo).

Errores AFIP Punto de Venta

Otros errores relacionados a la emisión de facturas electrónicas pueden ser:

- 1501: El Punto de Venta debe ser del tipo habilitado para el régimen CAE Codificación de Productos - Web Services o del régimen CAEA

En este caso se debe revisar que el tipo de punto de venta dado de alta coincida con el servicio web a consumir (campo "sistema de facturación asociado"):

- RECE para aplicativo y web services (CAE WSFEv1 factura electrónica mercado interno tradicional sin detalle)
- RECE para aplicativo y web services EXPORTACION (CAE WSFEXv1 factura electrónica exportación con detalle)

- Codificación de Productos Web Services (CAE WSMTXCA factura electrónica mercado interno con detalle)
- CAEA - Codificación de Productos (CAE anticipado WSMTXCA factura electrónica mercado interno con detalle)

No debe figurar "Factura en Linea" porque ese punto de venta solo permite operar por "Clave Fiscal", servicio interactivo de "Comprobantes en Línea"

Errores de tipos de datos

Se recuerda que este componente es dinámico y generalmente no realiza conversión ni ajustes en los tipos de datos, por lo que la aplicación que lo utilice debe revisarlos y verificar que sean correctos para ser enviados, respetando las especificaciones y validaciones de AFIP.

Debido a que los distintos lenguajes de programación pueden almacenar y pasar los valores de formas diferentes (eventualmente en registros recordsets, campos fields o similares), es recomendable que se conviertan todos los datos a string (cadenas de caracteres), para evitar errores de conversión y redondeo. También se pueden usar otros tipos de datos básicos como enteros (integer), flotantes (float), numérico/moneda (currency/decimal) que son los que en definitiva reconoce AFIP. Las fechas deben pasarse en formato string según el formato de cada webservice. Los importes deben pasarse con punto decimal (no coma, y sin separadores de miles ni signo pesos).

En VB se pueden convertir los valores con `CStr` (string), `CLng` (long), `CCur` (currency), etc. o incluso las funciones `str` o `val` en ciertos casos, ejemplo:

```
tipo_cbte = CInt(rs!tipo_cbte)
```

Para un ejemplo concreto, ver la implentación completa de referencia: factura_electronica.mdb (Access 2k o superior)

En caso de enviar los tipos de datos incorrectos, en `Traceback` (ver Manejo de Excepciones) podrá encontrarse algunos de los mensajes frecuentes:

- `AttributeError: 'PyIDispatch' object has no attribute 'strip'`: se envió un dato que no puede convertirse a string
- `PyIDispatch at 0x10531678 with obj at 0x0C997832`: se usó un tipo de datos no convertido/soportado, ver Mensajes XML
- `TypeError: int() argument must be a string or a number, not 'NoneType'`: se envió un nulo en un campo que necesitaba un número
- `ValueError: invalid literal for int() with base 10: 'A'`: se envió un string en un campo que necesitaba un número

Errores de Negocio

Error 10242: El campo de identificación de la Condición de IVA del receptor no es un valor valido.

Este campo es obligatorio. Para mayor detalle consular el método FEParamGetCondicionIvaReceptor.

A partir de 2025 es necesario enviar el nuevo campo `condicion_iva_receptor_id`, ver ManualPyAfipWs#Aclaraci%C3%B3nparaVersion4RG5616

Ver ProyectoWSFEv1#CondicionIvaReceptorA para la lista de condiciones de IVA. Para más información ver: ProyectoWSFEv1#RG56162024FEv4

Errores internos de AFIP

Ocasionalmente pueden producirse errores no documentados en las especificaciones técnicas de AFIP. Por ej, cuando se intenta utilizar la URL de un webservice obsoleto como WSBFE en vez de WSBFEv1 el servidor de AFIP ha devuelto el siguiente error:

- soap:Server: Server was unable to process request. ---> Object reference not set to an instance of an object.

Otros problemas internos de los servidores de AFIP, que los resuelve dicho organismo automáticamente (sin intervención necesaria, generalmente en el transcurso del día) son:

- 500: Error general DB (InsertandAuthorize - fex_pck.prc_ins_rcel_comprobantes OEX) - (1654) - ORA-01654: unable to extend index MIGRA.RCEL_CMP_UNC_INDEX by 4 in tablespace FISCO_DATOS ORA-06512: at "MIGRA.FEX_PCK", line 353 ORA-06512: at line 1
- 500: Error interno DB - fex_Rece_Pck.get_ctz
- 500: Error interno de aplicación: - Metodo FECompUltimoAutorizado. Timeout expired. The timeout period elapsed prior to obtaining a connection from the pool. This may have occurred because all pooled connections were in use and max pool size was reached.
- 500: Error general DB (InsertandAuthorize - prc_ws_ins_cab) BFEError
- 500: Error interno de aplicación: - Metodo FECompUltimoAutorizado. ORA-12535: TNS:operation timed out
- 500: Error interno de aplicación: - Metodo FECAESolicitar - Index was outside the bounds of the array.
- 501: Error interno de base de datos - CuitsEnPadronSet. ORA-00257: archiver error. Connect internal only, until freed. ORA-02063: preceding line from GHOST ORA-06512: at "ORA.RECE_V1_PCK", line 1651 ORA-06512: at line 1
- 501: Error interno de base de datos - CuitsEnPadronSet. ORA-28000: the account is locked ORA-02063: preceding line from E15K_PADR ORA-06512: at "ORA.RECE_V1_PCK", line 1806 ORA-06512: at line 1
- 501: Error interno de base de datos -! Metodo CuitsEnPadronSet. ORA-01033: ORACLE initialization or shutdown in progress
- 501: Error interno de base de datos - CAECabInsert. ORA-02068: following severe error from CONTRIB ORA-03113: end-of-file communication channel. ORA-06512: at "ORA.RECE_V1_PCK"...
- 501_ Error interno de base de datos - CAECabInsert. ORA-01001: invalid cursor ORA-06512: at "ORA.RECE_V1_PCK", line 580 ORA-06512: at "ORA.RECE_V1_PCK", line 1070 ORA-06512: at line 1
- 501: Error interno de base de datos - CAEDetInsert. ORA-03233: unable to extend table ORA.F136_DETALLES_FACTE subpartition F136_DET_2011_CLAVE_08 by 128 in tablespace USERS
- 501: Error interno de base de datos - CAESequenceGenerar.ORA-01502: index 'ORA.F136_DET_CAE_UK' or partition of such index is in unusable state ORA-06512: at "ORA.RECE_V1_PCK", line 1477 ORA-06512: at line 1

- 501: Error interno de base de datos - CAECabInsert. ORA-01552: cannot use system rollback segment for non-system tablespace 'FISCO_DATOS' ORA-06512: at "ORA.RECE_V1_PCK", line 1168 ORA-06512: at line 1
- 501: Error interno de base de datos - Metodo CuitsEnPadronSet. ORA-01033: ORACLE initialization or shutdown in progress
- 501: Error interno de base de datos - AlicIvaCAEInsert.ORA-14400: inserted partition key does not map to any partition ORA-06512: at "ORA.RECE_V1_PCK", line 1308 ORA-06512: at line 1
- 501: Error interno de base de datos - CuitsEnPadronSet. ORA-12541: TNS:no listener ORA-06512: at "ORA.RECE_V1_PCK", line 1831 ORA-06512: at line 1
- 501: Error interno de base de datos - CuitsEnPadronSet. ORA-04031: unable to allocate 20528 bytes of shared memory ("shared pool","unknown object","sga heap(1,0)","KTI SGA freeable small pool"): ORA-02062: preceding line from SEFI, ORA-06512: at "ORA.RECE_V1_PCK", line 1898; ORA-06512: at line 1
- 501: Error interno de base de datos - CAECabInsert. ORA-01552: cannot use system rollback segment for non-system tablespace 'USERS' at "ORA.RECE_V1_PCK", line 1202.
- 501: Error interno de base de datos - Metodo CuitsEnPadronSet. ORA-12505: TNS:listener could not resolve SID given in connect descriptor
- 501: Error interno de base de datos - CAECompConsultar.ORA-01502: index 'ORA.IX_F136_DETALLES_TPN' or partition of such index is in unusable state
- 501: Error interno de base de datos - CAECabInsert. ORA-01502: index 'ORA.F136_CAB_CUIT_ID' or partition of such index is in unusable state
- 501: DataBase - Resultado inesperado
- 501: Error interno de base de datos - CAESequenceGenerar.ORA-01654: unable to extend index ORA.F136_DET_CAE_UK by 1024 in tablespace FISCO_FE ORA-06512: at "ORA.RECE_V1_PCK", line 1781 ORA-06512: at line 1
- 502: Error interno de base de datos - Autorizador CAE - Transacción Activa
- Error interno de aplicación: - Metodo FECompConsultar.ORA-01034: ORACLE not available ORA-27101: shared memory realm does not exist
- Error interno de aplicación: - Metodo FECompConsultar.ORA-01035: ORACLE only available to users with RESTRICTED SESSION privilege
- Error interno de aplicación: - Metodo FECompUltimoAutorizado. Timeout expired. The timeout period elapsed prior to obtaining a connection from the pool. This may have occurred because all pooled connections were in use and max pool size was reached.
- Error interno de aplicación: - Metodo FECompUltimoAutorizado. ORA-28000: the account is locked
- soapenv:Server: Error de conexin DB [Cd. A001]
- soapenv:Client: Error de acceso a la base de datos
- soapenv:Client: business.0010
- soapenv:Client: org.apache.axis2.databinding.ADBException: unexpected subelement consultarCTGDatos

También pueden ocurrir Errores de Conectividad a nivel de sistema operativo: `socket error` / `io error`, `timeout`, `connection refused`, `ssl handshake`, `server not found`, etc. (ver arriba), especialmente cuando los servidores de AFIP se saturan por cambios y actualizaciones.

Realizar la consulta con la mesa de ayuda de AFIP: sri@afip.gov.ar (producción) y webservices@afip.gov.ar (homologacón) para dejar constancia del problema, y si corresponde, ver de utilizar los métodos tradicionales para emitir comprobantes según RG2485/08 Artículo 33:

> En el caso de inoperatividad del sistema se deberá emitir y entregar el comprobante respectivo, utilizando el controlador fiscal o los comprobantes manuales, según corresponda hasta tanto esta Administración Federal apruebe otro procedimiento alternativo de respaldo.

Importante: en general es obligatorio informar los comprobantes emitidos manualemente por formulario multinota en dependencia de AFIP, a través de servicios en linea o como AFIP disponga en casos particulares. Consultar con el contador la normativa y legislación vigente.

Errores de Permisos

Dependiendo de las políticas de seguridad, puede ser necesario dar permisos de acceso a la carpeta cache (archivos temporales), o aquellas donde se almacenen los archivos de intercambio (entrada, salida, PDF o XML)

IMPORTANTE: los certificados y claves privadas no deben ser expuestas en una carpeta pública del servidor web o similar. Utilizar passphrase (ver Métodos WSAA) para mayor protección y seguridad.

Linux

En GNU/Linux generalmente es necesario que el usuario del servidor web (www-data o similar) pueda acceder a la carpeta.

Por ejemplo, se podría utilizar los siguientes comandos:

```
chown www-data -R /opt/pyafipws
chmod a+xr -R /opt/pyafipws
chmod a+xwrt -R /opt/pyafipws/cache
```

Revisar los permisos con su administrador de sistemas.

Windows

Acceso Denegado (Archivos)

Error: `IOError: [Errno 13] Permission denied: 'c:\Archivos de Programa\PyAfipWS\Cache\TA.XML`

En general es recomendable utilizar una carpeta con permisos de escritura para la aplicación.

Si esto no es posible, se puede configurar los permisos de acceso:

- Explorador de Archivos
- Buscar carpeta de instalación (C:\Archivos de Programa\PyAfipWS)
- Propiedades (click derecho)
- Solapa Seguridad
- Seleccionar "Usuarios" que ejecutan la aplicación
- Editar
- Modificar (habilitar)
- Aceptar

Ejemplo:

Acceso Denegado (CreateObject/CreateInstance)

En ciertos servidores (Windows 7 o superior), dependiendo de la configuración, debe permitir inicio y activación de los componentes.

Error al llamar a `CreateObject`, `CreateInstance` o similar: `"Acceso Denegado(HRESULT 0x80070005 E_ACCESSDENIED"`

Solución:

- Inicio, buscar "Servicios de Componentes" (o via el "Panel de Control", "Herramientas Administrativas")
- Equipos, "Mi PC", "Configuración DCOM"
- Buscar WSAA, WSFEv1 o el componente que desee configurar
- Propiedades (Click Derecho)
- Solapa Seguridad
- Permisos de inicio y activación: Personalizado, Editar
- Agregar los usuarios que deben tener permisos necesarios
- Aceptar

Repetir para Permisos de Acceso y Configuración de ser necesario.

Ejemplo:

.

. . . . . .