INDICE

# Objeto

Abreviaciones

TA = Ticket Acceso

WSFE = Servicio Factura Electronica

WSAA = Servicio de conexión a AFIP

CAE = Codigo de Autorizacion Electronica

CAEA = Codigo de Autorizacion Electronica Anticipada

Q = Periodo + Quincena

Inicializar

El objeto tiene propiedades que indicaremos en el uso de cada servicio. Aca aclararemos al principio las que son generales

.tiempoExpiracion Alamecena la vigencia del TA, DATETIME

TA variable publica

.sw\_produccion indica que se maneja en modo producción BOOLEAN

.pathCertificacos ruta donde están los certificados CARÁCTER

.hkey / .hcrt Nombre de los archivos con extensión .key y .crt de homologación

.pkey / .pcrt Nombre de los archivos con extensión .key y .crt de producción

.traza almecena la traza para el desarrollador

.trazauser mensajes de alerta para el operador

.recuperarxml para el desarrollador, almacenamos los xml de envio y respuesta a los servicios

.p\_cuit CUIT de la Empresa

P\_passarba CLAVE DE ARBA

Manejo de Errores.

Tenemos un .prg que captura el error a partir del objeto syerrhand.prg

* ErrHand1 (WSAA)
* ErrHand2 (WSFE)
* ErrHand3 (PADRON)
* ErrHand\_iibb (IIBB)

Nosotros tenemos una función para almecenar las consultas o textos. SaveSQL() que también lo usamos para guardar el XML

\*----------------------------------------------------------------------------

\* FUNCION SaveSQL(lcCmd,lcArchivo,llgenera)

\*----------------------------------------------------------------------------

\* Guarda la consulta al motor

\*----------------------------------------------------------------------------

FUNCTION SaveSQL

PARAMETERS lcCmd,lcArchivo,llgenera

llgenera = IIF(PCOUNT()<3,.f.,llgenera)

IF LEN(LTRIM(lcCmd))=0

RETURN

ENDIF

IF LEN(LTRIM(lcArchivo))=0

RETURN

ENDIF

lldesarrollo=(\_vfp.startmode()#4)

lcRuta = SYS(5)+ "\tempsql\"+ALLTRIM(goapp.initcatalo)

IF VARTYPE(goapp.rutaaplicacion)$'C' AND NOT lldesarrollo

lcRutaApli = IIF(LEN(ALLTRIM(goapp.rutaaplicacion))#0,goapp.rutaaplicacion,"")

lcRutaApli = RTRIM(lcRutaApli) + IIF(RIGHT(lcRutaApli,1)="\" or LEN(LTRIM(lcRutaApli))=0,"","\") &&Si es vacio o tiene \. Mantiene lo mismo.

lcRuta = IIF(LEN(LTRIM(lcRutaApli))#0,lcRutaApli+ "tempsql",lcRuta)

ENDIF

IF llgenera &&Determinamos que queres que siempre se guarde

IF !DIRECTORY(lcRuta)

MKDIR &lcRuta

ENDIF

ENDIF

SET SAFETY OFF

= STRTOFILE(lccmd,lcRuta+"\"+lcArchivo+".txt")

SET SAFETY ON

RETURN

# Facturacion Electronica WSFE

Conectar al servicio

Se utiliza el método .sw\_wsfe\_conexion(), hay dos partes la conexión a afip y el acceso a wsfe para trabajar.

Primero .sw\_wsaa\_iniciar() Se establece la conexión a AFIP y nos devuelve un TA (TicketAcces) para trabajar, este posee una vigencia, por ende se puede volver a utilizar mientras sea valida la fecha. La primera vez que lo solicitemos, se cargaran los certificados y almacenaremos el TA para proseguir, para reutilizarlo usamos TA\_guardar() que lo almacena en la terminal. La siguiente vez, buscara el Ta en Ta\_cargar(), si esta vigente, lo cargara y no haremos los pasos anteriores.

Luego si salio correcto invocamos a .sw\_wsfe\_conectar() donde a partir del TA y CUIT de la Empresa, pediremos autorización para WSFE. Si todo salio correcto .sw\_estado =TRUE

Modalidades

Este servicio tiene dos modos, CAE o CAEA, la diferencia es que el segundo la AFIP nos autoriza un Nro CAEA (Similar CAE) con vigencia quincenal, se puede usar offline. Y el primero es online. La autorización de CAEA se hace desde 5 dias antes del comienzo de la quincena, transcurrido la fecha, solo se puede usar CAE hasta la siguiente quincena. Por lo tanto si el cliente esta autorizado a CAEA, se debe dar de alta un punto de venta tipo CAE, para eventualidades.

Los dos modalidades tiene muchas similitudes al momento de componer la factura para autorizarla en caso de CAE e informarla en caso de CAEA.

Solicitar CAEA

Empezamos indicando como se pide el CAEA, porque luego trataremos de englobar el proceso.

Para solcitar el CAEA, invocamos el método reg\_solicitarCAEA, a partir de una fecha previamente cargada en .p\_fecha, obtenemos la Q (periodo + quincena). Establecemos la conexión al servicio WSFE (.sw\_wsfe\_conexion). Luego debemos consultar a AFIP si ya nos autorizo el CAEA para Q. Ya que si la solicitamos y existe, da error. Esto son los pasos cuando se solicita antes de la quincena, si por algún error no se llego a almacenar o AFIP no nos llego a pasar la respuesta, pero si se autorizo, lo que hago es buscar en nuestra tabla de CAEA (Debes almacenar el CAEA y Q histórico).

Proceso

La diferencia entre las dos modalidades es el concepto uno es Autorizar CAE, donde autorizamos la factura y AFIP nos devuelve el Nro CAE; y el otro es Informar CAEA, donde informamos los comprobantes que emitimos con el CAEA.

A tener en cuenta en CAE, es que AFIP no valida la situación fiscal del Contribuyente, por ende si hacemos una Factura B a un RI, es Valida.

Antes de invocar al proceso, debemos cargar los datos de la factura en dos objetos. oCabeCAE y oImpCAE. El primero se invoca con .crearFactura()

Recuperamos el ultimo comprobante autorizado/informado en afip (Creo que el concepto es que EXISTA), y además la fecha del mismo. Ya que en el caso de CAE, podemos facturar hasta 5 dias atrasados, si es que no se emitio nada a posterior de la fecha atrasada y que pertenezcan al mismo periodo, AFIP NO PERMITE facturar entre periodos, eso lo valida a partir de la fecha de su propio servidor, es decir no se puede anticipar la factura, si las fechas actual y a facturar no pertenecen al mismo periodo.

Si estamos trabajando con CAE debemos chequear un caso. Y es que entre la autorización por parte de AFIP y la captura de esta, se produce un corte ya sea de la red/internet/sistema etc.

Por ese motivo, nosotros internamente guardamos como funcionaba el sistema antes de los servicios, la numeración del comprobante+punto venta. Entonces comparamos el ultimo numero autorizado por AFIP y el ultimo por sistema. Si ambos coinciden EXCELENTE y continuamos con el proceso. SINO consultamos los datos del ultimo sistema + 1 para saber si no se llego a imprimir y afip ya lo autorizo WSFE.CompConsultar(). Si la respuesta es ERROR. Hay un error interno con la numeración del sistema y el trabajo es manual en saber el porque. Si recuperamos el comprobante, tenemos dos posibles casos y un problema.

El caso puede ser que AFIP Autorizo y por algún X motivo no llegamos a almacenarlo e imprimirlo. O el primero motivo mas estemos tratando de autorizar uno nuevo, sin haber reintentado el primero. Por ese motivo chequeamos el Total y CUIT del autorizado. Si corresponde, estamos en el primer caso, y recuperamos el CAE y continuamos el proceso final, o si es el segundo abortamos y debemos autorizar el comprobante anterior, dándose el primer caso descripto.

Por ese motivo, recomiendo crear un proceso que busque todos los comprobantes sin autorizar y recorra del mas viejo. O caso que nosotros no usamos por el tipo de NEGOCIOS es sino se autoriza eliminar y que vuelvan a realizar la factura.

En el caso de de CAEA pasa similar, nosotros informamos todos los comprobantes de la quincena y luego los marcamos para no volver a enviarlos, pero si en el proceso algo falla, debemos volver a reintentar. Para marcar los comprobantes usamos la propiedad nInformado. Si este es 1, proseguimos con el siguiente. Ahora también controlamos cual fue el ultimo informado que es exactamente igual al ultimo autorizado .sw\_ultimocomproba(). Si el uiltimo AFIP es mayor al ultimo sistema, entonces los autorizamos, pero no llegamos a marcarlo. Por ende los marcamos y recuperamos el CAEA, porque asi lo hace el proceso al finalizar, se puede hacer mas sencillo y escribir menos código, solo marcando la autorización .nInformado = 1. Pero si el numero es mayor. Ahí AFIP dara error porque estaras consultando algo que no existe. Por ende hay algún problema en tu sistema.

Si ambos modos CAE y CAEA no tuvieron inconvenientes con la numeración pasamos al proceso similar que es generar la factura.

Nosotros usamos el concepto 1 Porque no facturamos servicios. Si fuera tu caso , debes poner en sFecha\_Serv\_Desde y sFecha\_Serv\_Hasta el rango del servicio.

Luego a partir de la colección o ImpFac, extraemos los impuestos.

Hay que tener en cuenta que para IG = IVA, “creo” que la diferencia entre base y tasa, puede dar una diferencia del 1%, sino da Error. En el caso de tributos (percepciones , i internos) hay que informar el % de la tasa/Alicuota.

Si el proceso salio correcto, lo que hicimos fue crear una estructura para la factura, ahora demos autorizarla/informar

El concepto es similar, excepto los servicios que se usan WSFE.CAESolcitar() o WSFE.CAEARegInformativo(). Si ambos son correctos tienen de respuesta=’A’ y almacenamos el CAE / CAEA enviado. Reconsultamos para corroborar al final que todo coincida.

Para el caso de CAE, almacenamos datos que debemos almacenar SI O SI.

Nro CAE = Obviamente en mcae\_nrocae

CAE TIPO = ‘CAE’ , es para saber si se debe informar o no, solo CAEA se informa. Mcae\_caetipo

Fecha Vto CAE = Requerido para informar en la impresión mcae\_fechavto

Fecha Emision = Pudo haberse cambiado si eran de periodos distintos. Mcae\_fechafac

En el caso de CAEA, como mencionamos anteriormente solo es fundamental el .nInformado = 1, para no volver a enviarlo.

Si todo salio correcto .sw\_estado TRUE

ACLARARION FUNDAMENTAL

Si se inscribe a WSFE, viene de la mano con el Regimen Informativo de Ventas y Compras. Se genera dos archivo por ventas y compras.

Este servicio WSFE es para mercado interno. No se informa los productos de la venta, solo los totales. Si se requiere el otro servicio. No hemos trabajado con el.

El método solicitar\_select() es un ejemplo de nuestra SQL para recuperar

Caso a tener en cuenta en CAEA, El cliente puede que no trabaje en cierta Q pero se solicito el CAEA, o que tenga mas de un punto de venta con CAEA y solo utilizo algunos.

Para esos caso SI O SI se debe informar sin movimientos le punto de venta para la Q. Para eso tenemos el método informar\_sinmov() donde le pasamos los datos de la Q que debemos informar, p\_periodo carácter, p\_quincena numérico. Dejo el nuestro como prueba, ya que esta la Queary para sacar si hubo una venta para ese punto de venta entre las fechas validadas de la Q.

# Padron ARBA

Conectar al servicio

Se utiliza únicamente .sw\_iibb\_conectar() con el cuit de la empresa (p\_cuit) y la clave de arba (p\_passarba), ya que esto es un servicio directo de ARBA, no requiere WSAA.

Modalidades

Aca tenemos dos opciones para dos trabajos distintos. Consultar un cliente o en lote, la principal diferencia es que en lote no aborta, sino que devuelve el error u prosigue con el siguiente, para en su sistema armar una lista de errores. El Form de en Lote o Individual escapa al objeto.

Independientemente se usa un objeto oContribuyente, que se crea en el init del objeto

PUBLIC oContribuyente

oContribuyente = CREATEOBJECT("custom")

oContribuyente.AddProperty('cuit','')

oContribuyente.AddProperty('periodo',0)

oContribuyente.AddProperty('perceiibb',0)

oContribuyente.AddProperty('reteiibb',0)

oContribuyente.AddProperty('cadenaerrores',0)

Primero analicemos el individual, el método es arba\_ibto\_consultarcuit(), que se conecta y luego genera el proceso. Este se le determina el rango a consultar, el rango debe ser un periodo, asi que se toma del objeto oContribuyente.periodo, también el cuit, alícuota percepción y alícuota de retención. Estas dos ultimas las tomamos para mostrar si hubo alguna diferencia entre las que ya estaban cargadas.

Entonces a partir de esto, consultas con el servicio IBB.ConsultarContribuyentes(). En teoría esto sirve para hacer todo en lote, pero prefiero individualemente para poder analizar el error. Si el servicio funciona correctamente, analizamos el XMLResponse, ya que encontré muchos errores en el objeto IIBB, con todos los objetos que trabajamos se si se devuelve correcto la consulta se puede usar AnalizarXML(“XmlResponse”), es del desarrollador de Python y recupera el atributo que necesites.

Si se capto un error, analizamos el error desde iibb.obtenerTagXml(‘mensajeError’)

Caso contrario, recuperamos los datos.

oContribuyente.cuit = Iibb.cuitContribuyente

oContribuyente.perceiibb iibb.AlicuotaPercepcion

oContribuyente.reteiibb = iibb.AlicuotaRetencion

Segundo caso en Lote, que serviría para actualizar un padrón de arba. Lo que hago es iniciar la conexión manualmente

.sw\_iibb\_iniciar()

If .sw\_estado

Recorro la lista de cuits, solo funciona con cuit validos, sino da error.

.arba\_ibto\_lote()

If .sw\_estado

Actualizo las alícuotas y fecha

Else

Marco el cuit con errores y los muestro al operador

Endif

Endif

Analizando el método arba\_ibto\_lote(), la diferencia es que anlizamos el CodigoError, que los principal error encontrado es el código 11, que el cuit no pertenezca al padrón. Luego sin similares, se captan los datos en las mismas del objeto oContribuyente.

# Consultar CUIT

Conectar al servicio

Primero invocamos el método .cuit\_consultar() esto no direcciona a primera instancia a sw\_padron5\_iniciar(). Usa el mismo que otros como WSFE para establecer la conexión con AFIP que es el .sw\_wsaa\_iniciar(), esto genera un TA.

Luego proseguimos pidiendo la autorización para usar los servicios de Padron\_A5 sw\_padronA5\_Conectar(). Que funciona similar tb al sw\_wsfe\_conectar() pero tiene otro link.

Si todo funciona correctamente, usamos el método cuit\_proceso\_2. (La diferencia con cuit\_proceso es que este ultimo en teoría toma los impuestos de los atributos de PADRON, pero en la practica esto me devuelve vacio, por esa razón en proceso\_2, los tomo de ObtenerTagXML)

La estructura es muy similar a ARBA, vale aclarar que solo funciona para CUIT, NI DOCUMENTO, NI CUIL.

Se cargan los valores en el objeto oConstanciaCUIT, que también la estrucutura se encuentra en el Init

PUBLIC oConstanciaCuit

oConstanciaCuit = CREATEOBJECT("custom")

oConstanciaCuit.AddProperty('cuit',0)

oConstanciaCuit.AddProperty('denominacion','')

oConstanciaCuit.AddProperty('tipo','')

oConstanciaCuit.AddProperty('tipo\_doc',0)

oConstanciaCuit.AddProperty('estado',"")

oConstanciaCuit.AddProperty('direccion',"")

oConstanciaCuit.AddProperty('categoiva',"CF")

oConstanciaCuit.AddProperty('localidad','')

oConstanciaCuit.AddProperty('provincia','')

oConstanciaCuit.AddProperty('cod\_postal','')

oConstanciaCuit.AddProperty('imp\_iva','')

oConstanciaCuit.AddProperty('catego\_mt','')

Un tema que observe es que de PADRON.Consultar(cDocumento) el objeto PADRON.cuit es distinto a PADRON.idPersona (este ultimo si es CDocumento).

El mayor problema viene para determinar en que impuestos esta inscripto el CUIT. Entonces vamos recuperando el XML los impuestos, Si el existe, nos devuelve un nIdImpuesto, si este es 0, no existe mas impuestos.

Ahora para recuperar lo que nos importa que es la categoría ante IVA y si existe el categoría de Monotributo. Podria servir de Ganancias? Aun no lo se.

Esta es una estructura de Respuesta de Monotributista

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">

<soap:Body>

<ns2:getPersonaResponse xmlns:ns2="http://a5.soap.ws.server.puc.sr/">

<personaReturn>

<datosGenerales>

<apellido>VILLARREAL</apellido>

<domicilioFiscal>

<codPostal>8000</codPostal>

<descripcionProvincia>BUENOS AIRES</descripcionProvincia>

<direccion>VIEYTES 1270</direccion>

<idProvincia>1</idProvincia>

<localidad>BAHIA BLANCA</localidad>

<tipoDomicilio>FISCAL</tipoDomicilio>

</domicilioFiscal>

<estadoClave>ACTIVO</estadoClave>

<idPersona>20302299936</idPersona>

<mesCierre>12</mesCierre>

<nombre>MARCOS EZEQUIEL</nombre>

<tipoClave>CUIT</tipoClave>

<tipoPersona>FISICA</tipoPersona>

</datosGenerales>

<datosMonotributo>

<actividadMonotributista>

<descripcionActividad>SERVICIOS DE CONSULTORES EN INFORMÁTICA Y SUMINISTROS DE PROGRAMAS DE INFORMÁTICA</descripcionActividad>

<idActividad>620100</idActividad>

<nomenclador>883</nomenclador>

<orden>1</orden>

<periodo>201602</periodo>

</actividadMonotributista>

<categoriaMonotributo>

<descripcionCategoria>D LOCACIONES DE SERVICIO</descripcionCategoria>

<idCategoria>38</idCategoria>

<idImpuesto>20</idImpuesto>

<periodo>201802</periodo>

</categoriaMonotributo>

<impuesto>

<descripcionImpuesto>MONOTRIBUTO</descripcionImpuesto>

<idImpuesto>20</idImpuesto>

<periodo>201602</periodo>

</impuesto>

</datosMonotributo>

<metadata>

<fechaHora>2019-06-05T20:55:43.328-03:00</fechaHora>

<servidor>aws.afip.gov.ar</servidor>

</metadata>

</personaReturn>

</ns2:getPersonaResponse>

</soap:Body>

</soap:Envelope>

Como se observa el Ciclo, si detecta un IDImpuesto, Capturo cual es la DescripcionImpuesto, para detectar un Monotributista, el IDImpuesto será 20. En ese caso busco en el XML categoriaMonotributo por si se necesita. Vale aclarar que si se usa un ObtenerTagXML y el Tag no existe da error, que eso es lo que yo creo pq no funciona PADRON\_CATEGO\_MT.

Para los demás categorías de IVA, debemos chequear si esta INSCRIPTO EN IVA, Tenemos 4 opciones mas: 32 Exento, 33 No Inscripto, 34 No Categorizado, 30 Activo (Responsable Inscripto)

Luego de obtener los datos se concluye la consulta.

# Certificados

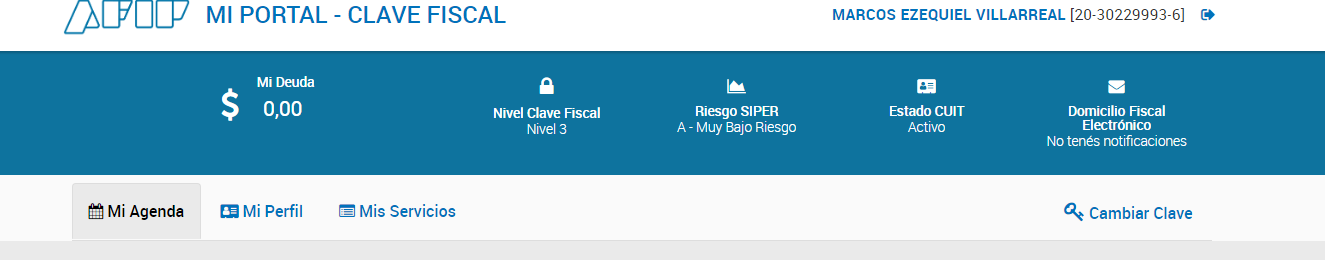
Hay montones de videos que te indican como dar de alta los certificados. Lo dejo al criterio, por si se encuentra alguno mas didáctico.

<http://www.afip.gob.ar/ws/documentacion/certificados.asp>

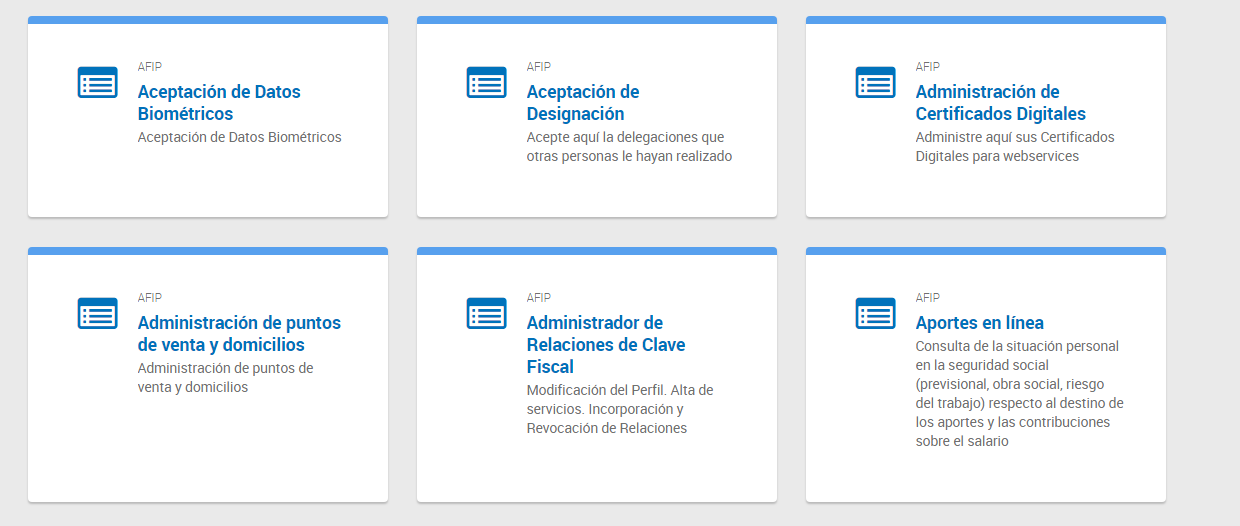
Lo que hay que tener en cuanta es adherirse a los servicios.

[Delegación de Webservices AFIP con el Administrador de Relaciones](http://www.afip.gob.ar/ws/WSAA/ADMINREL.DelegarWS.pdf)

Esta un poco desactualizado con respecto a como entrar y encima AFIP cambio el portal hace unos semanas.



Ahora debemos ir a MIS SERVICIOS



ADMINISTRADOR DE RELACIONES DE CLAVE FISCAL

En la pagina 7 es donde debemos prestar atención.

Para Factura Electronica buscar en WebService “Factura Electronica”

Para Consultar CUIT buscar “Servicio Consulta Padron A5”