

CONECTOR /LIBRERIA PHP

El conector o librería **ConectorPHP**. es un componente de software que facilita la integración de sistemas basados en php proveyendo la funcionalidad necesaria para trabajar con el API FEL de INFILE.

La libreria esta conformada por dos Metodos:

1. Generacion de archivos XML.

Clase: generar_xml.php, metodo **ToXml()**

2. Consumo del servicio de rest API FEL de INFILE.

Clase: servicio_fel.php, metodo **Certificar()**

1. Iniciando

1. Para empezar a usar la librería es necesario copiar la carpeta **fel** a la carpeta de tu proyecto.
2. En Cualquier XML en las modalidades de **Certificación** y **Anulación** en los campos **FechaHoraEmision** y **FechaEmisionDocumentoAnular** respectivamente se debe de enviar una fecha en el formato **ISO-8601** , todas las fechas en esta librería/conector son de tipo **String** para evitar inconvenientes de formato por temas de configuracion regional desde donde se utilice la librería.

TIP: Una de las formas para manejar la fecha en el formato ISO-8601

```
$fecha = date("c");
```

2. Utilizando La Liberia (Generacion de XML)

Proceso de generación de archivo XML para Certificación

Cada sección del xml es representada a través de una clase y cada atributo de la clase es equivalente a los campos que conforman cada nodo del xml.

A continuación el listado de clases que pueden ser utilizadas para formar cualquier tipo de documento:

Certificado

documento_fel.php

Para los datos Generales

`datos_generales.php`

`datos_emisor.php`

`datos_receptor.php`

`frases.php`

`items.php`

`impuesto_detalle.php`

`total_impuestos.php`

`totales.php`

Para los Complementos

`complemento_nota.php`

`complemento_cambiaria.php`

`complemento_cambiaria.php`

`complemento_especial.php`

`complemento_exportacion.php`

Para la transacción de Anulación

`anulacion_fel.php`

Generación de XML (Certificación)

NOTA: El conector/librería tiene integrado el motor de validaciones de esquema, lo que garantiza que los xml que reciba antes sean pre validados contra los archivos xsd .

Para obtener el XML de cualquier tipo de documento es necesario crear un objeto de la clase DocumentoFel y establecer los parámetros que requiere. Ejemplo:

1. Crear el objeto principal de la clase DocumentoFel

```
$documento_fel = new DocumentoFel();
```

2. Crear el objeto para los datos del Emisor

```
$datos_emisor = new DatosEmisor();
```

3. Establecer los parámetros para los datos del Emisor:

```
$datos_emisor->setAfiliacionIVA("GEN");  
$datos_emisor->setCodigoEstablecimiento(1);  
$datos_emisor->setCodigoPostal("01001");  
$datos_emisor->setCorreoEmisor("fel@fel.com.gt");  
$datos_emisor->setDepartamento("Guatemala");  
$datos_emisor->setMunicipio("Guatemala");  
$datos_emisor->setDireccion("Ciudad de Guatemala");  
$datos_emisor->setNITEmisor("12521337");  
$datos_emisor->setNombreComercial("URBANMEDIASOFT, S.A.");  
$datos_emisor->setNombreEmisor("URBANMEDIASOFT, S.A.");  
$datos_emisor->setPais("GT");
```

4. Agregar el objeto de los datos del emisor al objeto principal

```
$documento_fel->setDatos_emisor($datos_emisor);
```

5. De la misma forma anterior se procederá con los Datos Generales, Datos Receptor y Totales

6. Respecto a los Ítems la clase DocumentoFel recibe un Array de tipo Item. El siguiente ejemplo muestra a través de un ciclo for como se agregan 5 detalles al documento al mismo tiempo que se agregan dos impuestos a cada detalle.

```
for ($i = 1; $i <= 5; $i++) {  
    $items = new Items();  
    $items->setNumeroLinea(i);  
    $items->setBienOServicio("B");
```

```

        $items->setCantidad(5.5);

        $items->setDescripcion("Caja de Lapiceros");

        $items->setDescuento(0.75);

        $items->setPrecio(448.00);

        $items->setPrecioUnitario(448.00);

        $items->setUnidadMedida("UND");

        $items->setTotal(448.00);

        for ($j = 1; $j <= 2; $j++) {

            $impuestos_detalle = new ImpuestosDetalle();

            $impuestos_detalle->setNombreCorto("IVA");

            $impuestos_detalle->setCodigoUnidadGravable(1);

            $impuestos_detalle->setMontoGravable(400.00);

            $impuestos_detalle->setMontoImpuesto(48.00);

            $items->setImpuestos_detalle($impuestos_detalle);

        }

        $documento_fel->setItems($items);

    }

```

7. De la misma forma anterior se procederá para las frases y para los totales de impuestos.

Complementos

9. Cada complemento tiene su propia clase y de igual forma se establecen los datos para cada uno y se agregan al objeto principal documento_fel

Factura Cambiaría

```

//Seccion de datos para el complemento de Facturas Cambiarias

$complemento_cambiaria = new ComplementoCambiaria();

```

```
$complemento_cambiaria->setIDComplemento("Cambiaria");  
$complemento_cambiaria->setNombreComplemento("Cambiaria");  
$complemento_cambiaria->setURIComplemento("reemplazar_por_uri");
```

Factura Exportación

```
// Sección de datos para el complemento de Exportación  
  
$complemento_exportacion = new ComplementoExportacion();  
  
$complemento_exportacion->setIDComplemento("Exportacion");  
$complemento_exportacion->setNombreComplemento("Exportacion");  
$complemento_exportacion->setURIComplemento("reemplazar_por_uri");  
  
$complemento_exportacion->setCodigoComprador("1011");  
  
$complemento_exportacion->setCodigoConsignatarioODestinatario("101213");  
  
$complemento_exportacion->setCodigoExportador("789");  
  
$complemento_exportacion->setDireccionComprador("El Salvador");  
  
$complemento_exportacion->setDireccionConsignatarioODestinatario("El Salvador");  
  
$complemento_exportacion->setIncoterm("EXW");  
  
$complemento_exportacion->setNombreComprador("Carlos Lopez");  
  
$complemento_exportacion->setNombreConsignatarioODestinatario("Carlos Lopez");  
  
$complemento_exportacion->setNombreConsignatarioODestinatario("Carlos Lopez");  
  
$complemento_exportacion->setNombreExportador("Otro Nombre");  
  
$complemento_exportacion->setOtraReferencia("ninguna");  
  
$documento_fel->setComplementos($complemento_exportacion);
```

Factura Especial

```
$complemento_especial = new ComplementoFacturaEspecial();
```

```

$complemento_especial->setIDComplemento("Especial");
$complemento_especial->setNombreComplemento("Especial");
$complemento_especial->setURIComplemento("reemplazar_por_uri");
$complemento_especial->setRetencionISR(75.00);
$complemento_especial->setRetencionIVA(12.00);
$complemento_especial->setTotalMenosRetenciones(397.00);

```

Notas

```

$complemento_notas = new ComplementoNotas();
$complemento_notas->setIDComplemento("Notas");
$complemento_notas->setNombreComplemento("Notas");
$complemento_notas->setURIComplemento("reemplazar_por_uri");
$complemento_notas->setFechaEmisionDocumentoOrigen(date("c"));
$complemento_notas->setMotivoAjuste("Anulacion de documento No.
5");

$complemento_notas->setNumeroAutorizacionDocumentoOrigen("F72513D5-7233-4954-A239-
A7D8D5C92073");

$complemento_notas->setNumeroDocumentoOrigen("55");
$complemento_notas->setRegimenAntiguo("Antiguo");
$complemento_notas->setSerieDocumentoOrigen("ABCD");

```

Generación de XML (Anulación)

NOTA: El conector/librería tiene integrado el motor de validaciones de esquema, lo que garantiza que los xml que reciba antes sean pre validados contra los archivos xsd .

Para obtener el XML de la transacción de anulación, es necesario crear un objeto de la clase AnulacionFel y establecer los parámetros que requiere. Ejemplo:

1. Crear el objeto principal de la clase AnulacionFel

```
$anulacion_fel = new AnulacionFel();
```

2. Establecer los parámetros para la anulación

```
$anulacion_fel->setFechaEmisionDocumentoAnular(date("c"));
$anulacion_fel->setFechaHoraAnulacion(date("c"));
$anulacion_fel->setIDReceptor("12521337");
$anulacion_fel->setNITEmisor("800000001433");
$anulacion_fel->setMotivoAnulacion("Test API FEL");
$anulacion_fel->setNumeroDocumentoAnular("843D7F08-328A-402E-8B5F-EEFB3BC39BAB");
```

```
### Captura de respuesta del proceso de Generación de XML
(Anulacion/Certificacion)
```

```
// Variable para capturar la respuesta recibida del proceso de generación del XML.
Respuesta respuesta;
```

```
// Objeto para enviar los datos para generacion del XML
```

```
$generar_xml = new GenerarXml();
```

```
// Para el caso de la certificacion
```

```
$respuesta = $generar_xml.ToXml($documento_fel);
```

```
// Para el caso de la anulacion
```

```
$respuesta = $generar_xml.ToXml($anulacion_fel);
```

```
echo($respuesta->getXml());
```

3. Utilizando la Libreria (Envío al API)

La segunda funcionalidad de la librería es poder transmitir cualquier xml al API de certificación/anulación de INFILE. Por ejemplo el xml obtenido en el proceso anterior.

Ejemplo:

```
$conexion = new ConexionServicioFel();
```

```

        $conexion-
>setUrl("https://testing.ingface.net:8443/fel/certificacion/dte");

        $conexion->setMetodo("POST");

        $conexion->setContent_type("application/xml");

        $conexion->setUsuario("RHERNANDEZ47941162");

        $conexion->setLlave("1e3f3e2df2e3c04a2e6bc388b069e784");

        $conexion->setIdentificador("454s417431d0");


        echo("Enviando Documento al Servicio FEL.");


        $servicio = new ServicioFel();

        $respuesta_servicio = servicio->Certificar($conexion,
$respuesta->getXml());

```

4. Utilizando la Libreria (Revisando la Respuesta)

La clase de respuesta tiene los mismos atributos que el Json de respuesta del API de INFILE, por lo que examinar la respuesta es tan sencillo como obtener los atributos de la clase con los mismos nombres del archivo json de respuesta.

```

echo("Respuesta Servicio Fel: " + $respuesta_servicio->getResultado());

echo("Origen: " + $respuesta_servicio->getOrigen());

echo("Descripcion: " + $respuesta_servicio->getDescripcion());

echo("Cantidad Errores: " + $respuesta_servicio->getCantidad_errores());

echo("INFO: " + $respuesta_servicio->getInfo());

```

NOTA: La respuesta contiene un campo adicional que es ajeno a la respuesta del API pero que puede contener informacion importante acerca de algún error interno propio de la librería y no del servicio de INFILE.