

JUNTA DE ANDALUCIA

Consejería de Hacienda Industria y Energía


Autofirma y Miniapplet

Manual de integración de firma masiva

Versión: v16r01

Fecha: 13/06/2019

Queda prohibido cualquier tipo de explotación y, en particular, la reproducción, distribución, comunicación pública y/o transformación, total o parcial, por cualquier medio, de este documento sin el previo consentimiento expreso y por escrito de la Junta de Andalucía.

	Consejería de Hacienda Industria y Energía Dirección General de Transformación Digital	Autofirma y Miniapplet Manual de integración de firma masiva
---	---	---

HOJA DE CONTROL

Título	Manual de integración de firma masiva		
Entregable	Manual de integración de firma masiva con Autofirma		
Nombre del Fichero	20190613-Cliente@firma_Manual de Integración de firma masiva_V16r01		
Autor	DGTD		
Versión/Edición	v16r01	Fecha Versión	13/06/2019
Aprobado por		Fecha Aprobación	-
		Nº Total Páginas	14


REGISTRO DE CAMBIOS

Versión	Causa del Cambio	Responsable del Cambio	Área	Fecha del Cambio
v16r00	Primera versión del documento	UTE	UTE	14/05/2019
V16r01	Revisión del documento	DGTD	SCAE	13/06/2019

	<p>Consejería de Hacienda Industria y Energía</p> <p>Dirección General de Transformación Digital</p>	<p>Autofirma y Miniapplet</p> <p>Manual de integración de firma masiva</p>
---	--	--

ÍNDICE

1	Introducción.....	4
2	Cambios incluidos de la versión 1.6.JAv01.....	5
3	Recomendaciones de carga del componente de firma en la página web.....	6
4	Integración con la firma masiva.....	9
4.1	Definición del método <i>multiModeSign(params)</i>	9
4.2	Consideraciones a tener en cuenta sobre el método <i>multiModeSign(params)</i>	10
4.3	Retorno de las operaciones.....	11
4.4	Ejemplo de integración con el método <i>multiModeSign(params)</i>	12
4.5	Parámetros extra comunes.....	14

	<p>Consejería de Hacienda Industria y Energía</p> <p>Dirección General de Transformación Digital</p>	<p>Autofirma y Miniapplet</p> <p>Manual de integración de firma masiva</p>
---	--	--

1 Introducción


El cliente de firma es un paquete tecnológico que engloba tres aspectos diferenciados para dar soporte de firma electrónica a las aplicaciones web que necesiten de su uso para ejecutar sus procedimientos. De esta forma se debe diferenciar entre el cliente JavaScript, el applet y aplicación nativa de firma, ya sea Autofirma o las correspondiente para Android e iOS.

- **Applet.** El denominado Miniapplet de firma, es un applet de Java que se ejecuta embebido en el código HTML de las aplicaciones web que requieren firma electrónica, utilizando el plugin de Java del navegador que carga la página. Debido a los problemas de seguridad que presenta el plugin de Java, los diferentes navegadores están dejando de soportar dicha tecnología, lo que hace necesario buscar alternativas viables para poder mantener las funciones de firma electrónica.
- **Aplicación nativa de firma.** Para entornos de escritorio se trata de *AutoFirma*, una aplicación de escritorio con interfaz gráfica que permite la ejecución de operaciones de firma electrónica. Puede ser utilizada para la firma de ficheros locales o a través de aplicaciones web que integran operaciones de firma electrónica. Existen aplicaciones nativas análogas a AutoFirma para los entornos móviles Android e iOS, pero no disponen de las interfaces para firma de ficheros locales y solo son útiles a través de aplicaciones web. Estos componentes de firma son ajenos a los problemas que presenta el applet y por tanto es la alternativa propuesta para habilitar la firma electrónica.
- **El cliente de firma JavaScript (miniapplet.js).** Es el punto de integración común definido para incluir la posibilidad de firma electrónica en las diferentes aplicaciones web que la necesiten, sirviendo de interfaz transparente entre el uso del applet o la aplicación nativa del entorno utilizado.

El presente documento está dirigido a los integradores y sirve de complemento al manual de integración original del cliente de firma (*MiniApplet-v1-4-manual-integrador.pdf*), definiendo las pautas que deben seguir para habilitar la firma masiva cuando se utiliza Autofirma, ya que el documento anteriormente nombrado no incluye esta funcionalidad.


Hay que tener en cuenta que Autofirma se apoya en el cliente de firma JavaScript (miniapplet.js), el cual aporta transparencia de integración entre el Miniapplet y Autofirma. Sin embargo, la manera de integrar Autofirma con la firma masiva es diferente a la utilizada hasta ahora con Miniapplet.

Debido a lo anterior, no se recomienda el mecanismo de firma masiva utilizado hasta ahora con el miniapplet, siendo sustituido por el mecanismo de firma masiva descrito en este documento.

	Consejería de Hacienda Industria y Energía Dirección General de Transformación Digital	Autofirma y Miniapplet Manual de integración de firma masiva
---	---	---

2 Cambios incluidos de la versión 1.6.JAv01

- Se mejora la compatibilidad con Firefox 58 y superiores.
- Compatibilidad con Java 9
- Posibilidad de configuración de proxy.
- Se activa la comprobación de los certificados de las conexiones SSL. Esto implica que las conexiones con los servicios del servidor intermedio, el servicio de firma trifásica y el servicio de despliegue JNLP deben estar securizadas (en caso de estarlo) con certificados emitidos por una autoridad de certificación reconocida por Java.
- Actualización a JMulticard 1.5. Esta versión desactiva el soporte de las tarjetas de GyD, lo que permite que se carguen con el controlador oficial.
- Actualización a SpongyCastle 1.54.
- Actualización a JXAdES 0.2.0.
- Actualización a PDFBox 2.0.7.
- Corrección del diálogo de inserción de PIN de las tarjetas DNle/FNMT para admitir signos de puntuación
- Se agrega la compatibilidad con el método setStickySignatory() del MiniApplet para la selección única del certificado (sólo comunicación por sockets) y Autofirma de escritorio. No aplica para firmas mediante servicios intermedios ni clientes móviles.
- Se agrega la compatibilidad con los métodos de carga de datos del MiniApplet (sólo comunicación por sockets).
- Se agrega la compatibilidad con el método getCurrentLog() del MiniApplet (sólo comunicación por sockets).
- Mejora significativa en la estabilidad de la conexión entre AutoFirma y el navegador web. Esto corrige problemas de comunicación entre AutoFirma y Chrome.

	<p>Consejería de Hacienda Industria y Energía</p> <p>Dirección General de Transformación Digital</p>	<p>Autofirma y Miniapplet</p> <p>Manual de integración de firma masiva</p>
---	--	--

3 Recomendaciones de carga del componente de firma en la página web

De manera adicional a lo descrito el manual de integración “*Autofirma_Manual del Integrador_V16r00*”, en este apartado vamos a ver una serie de recomendaciones sobre como cargar el componente de firma en una página web. Se describen aspectos comunes de componente original y otros aspectos concretos de la versión publicada por la Junta de Andalucía ***miniapplet-1.6.JAv01-server***.

1. Se recomienda desacoplar el formulario de firma de la página principal, de esta forma evitamos cargar el componente de firma hasta el momento justo de firmar.
2. En la página con el formulario de firma, se recomienda habilitar la carga del componente al principio de la página y no al final como está recomendado para cualquier código JavaScript, el motivo es que de esta forma nos aseguramos de que el componente de firma estará cargado a tiempo cuando se utiliza el modo applet.
3. A continuación se describen los diferentes métodos que debemos usar para cargar el componente de firma una vez tenemos importado el script ***miniapplet.js***
 - **MiniApplet.setTryMiniApplet(boolean)**. Indica si debe intentarse o no la carga del applet mediante el plugin de Java del navegador.
 - **true (valor por defecto)**: Intentará cargar primero el applet si el plugin de Java está habilitado en el navegador. Si no consigue cargar el applet se utilizará la aplicación nativa AutoFirma.
 - **false**: No cargará el applet aunque el plugin de Java este habilitado en el navegador. De esta forma solo será compatible con la aplicación nativa AutoFirma.
 - **MiniApplet.setForceWSMode(boolean)**. Indica si debe usarse el modo *Servlet* de AutoFirma sin el uso del modo *SocketSSL* aunque este esté soportado.
 - **true**: La aplicación nativa AutoFirma utilizará el modo *Servlet* en lugar de *SocketSSL*, aunque este último esté soportado.
 - **false (valor por defecto)**: La aplicación nativa AutoFirma utilizará el modo *SocketSSL* si está soportado, en caso contrario el modo *Servlet* (IE 8 y 9).
 - **MiniApplet.cargarMiniAppletJA()**. Método de carga del componente. **Este método es una novedad de esta versión de Autofirma y se recomienda su uso en lugar de cargarMiniApplet(String), ya que tiene predefinida la URL de acceso al componente de firma.** En el caso de que se utilice el método *cargarMiniApplet(String)*, el parámetro indica la URL de acceso al componente de firma. *MiniApplet.cargarMiniApplet(HOST)*. El parámetro define el prefijo de otras funciones y es altamente recomendable establecer un valor correcto.
 - **MiniApplet.setServlets(HOST + "sign/StorageService", HOST + "sign/RetrieveService")**. Método de configuración de los servlets de almacenamiento y recuperación respectivamente, qué utiliza AutoFirma en modo *Servlet*. Teniendo en cuenta que las versiones 8 y 9 de IE usarán siempre el modo Servlets, al igual que las aplicaciones nativas de los entornos móviles Android e iOS, es altamente recomendable establecer una configuración correcta de este parámetro. Para más información sobre la firma trifásica vea el manual de integración “*Autofirma_Manual del Integrador_V16r00*”

Ejemplo de carga con applet:


```
<!DOCTYPE HTML>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" >
    <script type="text/javascript" src="https://SERVER_HOST/miniapplet-server/miniapplet.js"></script>
  </head>
  <body>
    <script type="text/javascript">
      var HOST = "https://SERVER_HOST/miniapplet-server/";
      MiniApplet.setTryMiniapplet(true);
      MiniApplet.setForceWSMode(false);
      MiniApplet.cargarMiniApplet(HOST);
      MiniApplet.setServlets(HOST + "sign/StorageService", HOST + "sign/RetrieveService");
    </script>
  </body>
</html>
```

Ejemplo de carga sin applet y modo Servlet de AutoFirma:

```
<!DOCTYPE HTML>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" >
    <script type="text/javascript" src="https://SERVER_HOST/miniapplet-server/miniapplet.js"></script>
  </head>
  <body>
    <script type="text/javascript">
      var HOST = "https://SERVER_HOST/miniapplet-server/";
      MiniApplet.setTryMiniapplet(false);
      MiniApplet.setForceWSMode(true);
      MiniApplet.cargarMiniApplet(HOST);
      MiniApplet.setServlets(HOST + "sign/StorageService", HOST + "sign/RetrieveService");
    </script>
  </body>
</html>
```

Ejemplo de carga sin applet y modo SocketSSL de AutoFirma:

```
<!DOCTYPE HTML>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" >
    <script type="text/javascript" src="https://SERVER_HOST/miniapplet-server/miniapplet.js"></script>
  </head>
  <body>
    <script type="text/javascript">
      var HOST = "https://SERVER_HOST/miniapplet-server/";
      MiniApplet.setTryMiniapplet(false);
      MiniApplet.setForceWSMode(false);
      MiniApplet.cargarMiniApplet(HOST);
      MiniApplet.setServlets(HOST + "sign/StorageService", HOST + "sign/RetrieveService");
    </script>
  </body>
</html>
```


	<p>Consejería de Hacienda Industria y Energía</p> <p>Dirección General de Transformación Digital</p>	<p>Autofirma y Miniapplet</p> <p>Manual de integración de firma masiva</p>
---	--	--

4 Integración con la firma masiva

Los métodos de integración de firma electrónica definidos originalmente en el cliente de firma JavaScript se han mantenido y están descritos en el manual de integración original (*Autofirma_Manual del Integrador_V16r00*), de forma que usándolo podremos usar indistintamente el Miniapplet o Autofirma para operaciones de firma simple, es decir, para realizar una sola firma.

En este apartado veremos las novedades introducidas en el cliente de firma JavaScript, para realizar operaciones de firma o multifirma simple o masiva, ya que estas novedades aportan transparencia completa a todas las funciones del cliente de firma. El uso de estas nuevas funciones hace transparente las operaciones de firma masiva para el componente Miniapplet y Autofirma por parte de las aplicaciones.


4.1 Definición del método *multiModeSign(params)*

En este apartado se define la manera de usar el nuevo método de integración de firma definido en el cliente de firma JavaScript. Este método engloba la firma/multifirma simple y masiva, utilizando Autofirma o Miniapplet según si se ha cargado o no el Applet de firma.

- ***multiModeSign(params);***

Los parámetros de entrada de este método de integración son los siguientes, teniendo en cuenta que al habilitar la firma masiva, debe permitir la definición de más de una operación y para habilitarlo se utilizarán tablas de datos:

- ***operationArray***. Tabla de datos obligatoria que define la operación a realizar en cada caso.
 - 'sign'.
 - 'cosign'.
 - 'countersign'.
- ***dataArray***. Tabla de datos obligatoria, qué define los datos a firmar,
 - datos o hashes en base64 para las operaciones 'sign'.
 - firmas en base64 para operaciones 'cosign' y 'countersign'.
- ***originaldataArray***. Tabla de datos obligatoria, qué define los datos originales en base64 y que solo útil para la operación 'cosign'.
 - Documento original en base64 para operaciones 'cosign'.
 - Cadena vacía ("") en los casos de 'sign' y 'countersign'.
- ***arrayLength***. Número entero de control y obligatorio que define el tamaño de las tablas de datos, de forma que todas deben coincidir con este tamaño.
- ***commonSignAlgorithm***. Algoritmo de firma obligatorio y común para todas las operaciones.
 - 'SHA1withRSA'.
 - 'SHA256withRSA'.
 - 'SHA512withRSA'.
- ***commonSignFormat***. Formato de firma obligatorio y común para todas las operaciones.

	<p>Consejería de Hacienda Industria y Energía</p> <p>Dirección General de Transformación Digital</p>	<p>Autofirma y Miniapplet</p> <p>Manual de integración de firma masiva</p>
---	--	--

- 'AUTO' (Solo válido para 'cosign' y 'countersign').
- 'CAAdES'.
- 'XAdES'.
- 'PAdES'.
- ...
- **commonSignParams:** Parámetros extra opcionales y comunes para todas las operaciones, separados por '\n'. Consulte en el Manual de Integración apartado "7.1 Paso de parámetros adicionales a los métodos de firma, cofirma y contrafirma" para saber cómo realizar el paso de parámetros y el apartado de información específica del formato de firma que desee realizar para saber los parámetros soportados por el formato en cuestión.
- **successCallback:** Función JavaScript de retorno SIN errores.
 - Por ejemplo, si en el código JavaScript del integrador se define e implementa la función *successCallback(signatureB64, certificateB64)*. El parámetro se pasará como *successCallback*.
- **errorCallback:** Función JavaScript de retorno CON errores.
 - Por ejemplo, si en el código JavaScript del integrador se define e implementa la función *errorCallback(type, message)*. El parámetro se pasará como *errorCallback*.

4.2 Consideraciones a tener en cuenta sobre el método *multiModeSign(params)*

1. Este método simplemente engloba todas las funciones de firma propias del Miniapplet y Autofirma (métodos *MiniApplet.sign*, *MiniApplet.coSign*, *MiniApplet.counterSign*) y aplica la configuración necesaria para permitir realizar firmas simples o masiva, todo en función de los parámetros de entrada. Teniendo esto en cuenta y haciendo uso del cliente de firma JavaScript, el integrador puede preferir implementar su propia lógica utilizando los métodos y configuraciones originales del cliente de firma.
2. El método utilizará siempre el Miniapplet de firma si este se ha cargado correctamente, utilizando Autofirma solamente cuando se ha detectado que no está cargado el applet. Podemos desactivar el plugin de Java en el navegador deseado para utilizar siempre AutoFirma.
 - Para poder utilizar el Miniapplet, deberá estar presente el archivo *miniapplet-full_1_6_JAv01.jar* junto al JavaScript *miniapplet.js*, siempre que el navegador utilizado cargue correctamente el plugin de Java.
 - Para utilizar AutoFirma, el navegador utilizado no debe cargar el plugin de Java y el usuario debe tener instalado correctamente AutoFirma en el sistema.
3. Cuando el método recibe más de una operación (firma masiva) y se usa el Miniapplet, se introducen automáticamente las llamadas a la instrucción ***MiniApplet.setStickySignatory(boolean)***, aplicando el valor **'true'** antes de la primera operación y **'false'** después de la última. Esto significa que al margen del número de operaciones a realizar, el certificado se pedirá una sola vez por cada operación *multiModeSign*.
4. Cuando el método decide el uso de AutoFirma por no estar cargado el Applet, se realizan todas las operaciones en una sola llamada a la aplicación. Esto significa que al margen del número de operaciones a realizar, el certificado se pedirá una sola vez por cada vez que se invoque el método.
5. Se recomienda (aunque no es obligatorio) el uso de la firma de hashes para evitar que la tabla de datos a firmar sea demasiado grande a la hora de invocar el método. Para habilitar este tipo de firmas es

	<p>Consejería de Hacienda Industria y Energía</p> <p>Dirección General de Transformación Digital</p>	<p>Autofirma y Miniapplet</p> <p>Manual de integración de firma masiva</p>
---	--	--

necesario introducir un parámetro extra común para todas las operaciones y en la tabla de datos a firmar definir el hash en base64 de los datos originales.

- En el caso de usar un algoritmo de firma 'SHA1withRSA' asignaremos los siguientes valores:
 - **dataArray=[]; dataArray[0]='<HASH-128>;**
 - **commonSignFormat='SHA1withRSA';**
 - **commonSignParams='precalculatedHashAlgorithm=SHA\n';**
 - En el caso de usar un algoritmo de firma 'SHA256withRSA' asignaremos los siguientes valores:
 - **dataArray=[]; dataArray[0]='<HASH-256>;**
 - **commonSignFormat='SHA256withRSA';**
 - **commonSignParams='precalculatedHashAlgorithm=SHA-256\n';**
 - En el caso de usar un algoritmo de firma 'SHA512withRSA' asignaremos los siguientes valores:
 - **dataArray=[]; dataArray[0]='<HASH-512>;**
 - **commonSignFormat='SHA512withRSA';**
 - **commonSignParams='precalculatedHashAlgorithm=SHA-512\n';**
6. La definición e implementación de las funciones JavaScript de retorno son responsabilidad del integrador y debe tener en cuenta que la llamada de dichos métodos difiere entre el uso del Miniapplet o Autofirma. En el siguiente apartado se definen detalladamente estas diferencias y una propuesta de retorno para el integrador.


4.3 Retorno de las operaciones

Como podemos ver en la definición del método **multiModeSign(params)** del apartado 4.1, el retorno de las firmas realizadas las obtenemos mediante funciones de *callback* definidas e implementadas en el propio JavaScript del integrador. Dicho esto, hay que tener en cuenta que estas llamadas de retorno difieren de si se está usando el Miniapplet o Autofirma.

- **Miniapplet.** Se llamará a la función de *callback* con cada operación realizada. La firma y el certificado se devolverán siempre en base64.
- **Autofirma.** Se llamará a la función de *callback* una sola vez con el resultado de todas las operaciones realizadas. Devolverá el certificado en base64 y las firmas en base64 de todas las operaciones separadas por el carácter '!'.

Para unificar el funcionamiento de la función de *callback* entre el uso del Miniapplet o AutoFirma, proponemos el siguiente esquema para la función de retorno, aunque queda bajo la responsabilidad del integrador usar cualquier otra estrategia.

```
function showResultCallback(signatureB64, certificateB64) {
    document.getElementById('signerCert').value = certificateB64;
    if(document.getElementById('result').value == "") {
        document.getElementById('result').value = signatureB64;
    } else {
        document.getElementById('result').value += ":" + signatureB64;
    }
}
```

	<p>Consejería de Hacienda Industria y Energía</p> <p>Dirección General de Transformación Digital</p>	<p>Autofirma y Miniapplet</p> <p>Manual de integración de firma masiva</p>
---	--	--

Básicamente, cuando se usa el Miniapplet la función de retorno propuesta añade las firmas de cada operación sobre una variable global y las separa con el carácter ':'. Lo hacemos así, porque cuando se usa Autofirma este será el formato devuelto para todas las operaciones, es decir, todas las firmas en base64 separadas por el carácter ':'. Una vez obtenido el resultado, la aplicación web podrá hacer ***split(':')*** sobre la variable global para obtener una tabla con todas las firmas realizadas, y manteniendo el mismo orden que se haya definido en las tablas de la llamada al método ***multiModeSign(params)***.

En el caso de producirse errores durante las operaciones, la función de retorno si tendrá el mismo funcionamiento, al margen de si se ha usado el Miniapplet o AutoFirma. Una propuesta para la implementación de dicho método es la siguiente.

```
function showErrorCallback(errorType, errorMessage) {
    alert("Type: " + errorType + "\nMessage: " + errorMessage);
}
```

4.4 Ejemplo de integración con el método multiModeSign(params)

A continuación se presenta un ejemplo de integración haciendo uso de método ***multiModeSign(params)***, presente en el cliente JavaScript (miniapplet.js) que se distribuye junto con el Miniapplet de firma a partir de la versión 1.4.JAv02.

El integrador implementa un código JavaScript en la página que realiza las firmas, definiendo los siguientes aspectos.

- Importar el cliente de firma JavaScript (miniapplet.js) en la página que realiza la firmas.
- Implementar una función para realizar la firmas y que invocará al método ***multiModeSign(params)***.
- Implementar las funciones de retorno para obtener las firmas o errores generados durante las operaciones.

```
<script type="text/javascript" src="https://ws024.juntadeandalucia.es/afirma-validator-miniapplet-1_6/miniapplet.js" />
<script type="text/javascript">

    // Función para mostrar logs en el campo 'console' de la página web.
    function showLog(newLog) {
        document.getElementById('console').value =
            document.getElementById('console').value + "\n" + newLog;
    }

    // Función de retorno de firmas y certificado del firmante. Mostrados en los campos 'result' y
    // 'signerCert' de la página web respectivamente.
    function showResultCallback(signatureB64, certificateB64) {
        document.getElementById('signerCert').value = certificateB64;
        if(document.getElementById('result').value == "") {
            document.getElementById('result').value = signatureB64;
        } else {
            document.getElementById('result').value += ":" + signatureB64;
        }
    }
}
```

```
}


// Función de retorno de errores.
function showErrorCallback(errorType, errorMessage) {
    showLog("Type: " + errorType + "\nMessage: " + errorMessage);
}

// Función de firma multi-modo.
function doMultiModeSign() {
    // Inicialización de los parámetros de entrada para el método de firma multi-modo. El
    // integrador normalmente pasará estos valores como parámetros y podrá construir
    // las tablas para la llamada.
    var operationArray = [];
    var dataArray = [];
    var originalDataArray = [];
    var arrayLength = 2;
    var commonSignAlgorithm = "SHA1withRSA";
    var commonSignFormat = "CAdES";
    var commonSignParams = "precalculatedHashAlgorithm=SHA\n";
    var successCallback = showResultCallback;
    var errorCallback = showErrorCallback;

    // Asignación de valores para las operaciones del método de firma multi-modo.
    operationArray[0] = "sign";
    operationArray[1] = "sign";
    dataArray[0] = "IHOhNXrjdy2l6lMkOg5CBWcANEc=";
    dataArray[1] = "VZ+KVRaV4im+Mv4RqCM+JahPTSY=";
    originalDataArray[0] = "";
    originalDataArray[1] = "";

    // Llamada a la función de firma multi-modo y captura de errores.
    try {
        // Reiniciamos el campo 'result' de la página web antes de invocar el
        // método.
        document.getElementById("result").value = "";
        MiniApplet.multiModeSign(operationArray, dataArray,
            originalDataArray, arrayLength, commonSignAlgorithm,
            commonSignFormat, commonSignParams, successCallback,
            errorCallback);
    } catch(e) {
        try {
            showLog("Type: " + MiniApplet.getErrorType() + "\nMessage: " +
                MiniApplet.getErrorMessage());
        } catch(ex) {
            // El Miniapplet de firma no implementa los métodos getErrorType y
            // getErrorMessage, solo implementados por el cliente JavaScript,
            // de forma que capturamos la posible excepción cuando usamos el
            // Miniapplet y mostramos la excepción original.
            showLog("Error: " + e);
        }
    }
}

</script>
```

	<p>Consejería de Hacienda Industria y Energía</p> <p>Dirección General de Transformación Digital</p>	<p>Autofirma y Miniapplet</p> <p>Manual de integración de firma masiva</p>
---	--	--

4.5 Parámetros extra comunes

Se enumeran en este apartado algunos de los parámetros extra comunes más utilizados para las operaciones de firma electrónica.

- Parámetro **format**. Para diferenciar entre los diferentes formatos XAdES.
 - 'format=XAdES Detached'. Define el modo Detached cuando se usa formato XAdES.
 - 'format=XAdES Enveloped'. Define el modo Enveloped cuando se usa formato XAdES.
 - 'format=XAdES Enveloping'. Define el modo Enveloping cuando se usa formato XAdES.
- Parámetro **target**. Para indicar que nodos firmar en las firmas en cascada.
 - 'target=tree'. Indica la firma en cascada de todos los nodos existentes en la firma original.
 - 'target=leafs'. Indica la firma en cascada solo de los nodos hoja existentes en la firma original.
- Parámetro **mode**. Para indicar si la firma debe incluir los datos originales o el resumen de los mismos (solo aplicable a las firmas binarias CAdES).
 - 'mode=explicit'. Indica que se firmen los datos explícitamente, es decir, que solo se haga referencia al hash de los datos firmados.
 - 'mode=implicit'. Indica que se firmen los datos explícitamente, es decir, que solo se haga referencia al hash de los datos firmados.
- Parámetro **precalculatedHashAlgorithm**. Para indicarle al cliente que los datos a firmar se corresponden con el hash del documento original y simplemente debe firmarlo directamente sin calcular su resumen.
 - 'precalculatedHashAlgorithm=SHA'. Indica que los datos a firmar se corresponden con un hash de 128bits.
 - 'precalculatedHashAlgorithm=SHA-256'. Indica que los datos a firmar se corresponden con un hash de 256bits.
 - 'precalculatedHashAlgorithm=SHA-512'. Indica que los datos a firmar se corresponden con un hash de 512bits.

Para más información sobre otros parámetros extra puede consultar el manual del integrador (*Autofirma_Manual del Integrador_V16r00*).