



# Junta de Andalucía



**Plataforma @firma/Autofirma**

---

**Manual de Integración Firma por Lotes**

Versión: v01r001  
Fecha: 11/08/2025

## HOJA DE CONTROL

<b>Título</b>	Plataforma @firma/Autofirma		
<b>Entregable</b>	Manual de Integración Firma por Lotes		
<b>Nombre del Fichero</b>	20250811-Autofirma-Manual del Integración de firma por lotes.docx		
<b>Autor</b>	UTE		
<b>Versión/Edición</b>	1	<b>Fecha Versión</b>	11/08/2025
<b>Aprobado por</b>		<b>Fecha Aprobación</b>	
		<b>Nº Total Páginas</b>	9

## REGISTRO DE CAMBIOS

Versión	Causa del Cambio	Responsable del Cambio	Area	Fecha del Cambio
v01r001	Creación del documento	UTE	UTE	11/08/2025

## CONTROL DE DISTRIBUCIÓN

Nombre y Apellidos	Cargo	Area	Nº Copias
José Ignacio Cortés Santos	Director de Proyecto	ADA	1

## ÍNDICE

1	Introducción.....	4
2	Integración con firma por lotes.....	5
3	Ejemplo de integración.....	8

## 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 y aplicación nativa de firma, ya sea Autofirma o la correspondiente App para Android y iOS.

- **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.
- **El cliente de firma JavaScript (autoscript.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 para la aplicación nativa del entorno utilizado.
- **Servicios intermedios, trifásicos y de firmas por lotes.** Los servicios intermedios son dos servicios auxiliares de comunicación que sirven, respectivamente, para el almacenaje y recuperación de documentos entre el cliente de firma JavaScript y la aplicación nativa de firma. Por su parte, los servicios trifásicos se utilizan por el cliente de firma JavaScript y la aplicación nativa de firma para poder realizar ciertas firmas en formatos trifásicos. Los servicios de lotes se utilizan entre el cliente de firma JavaScript y la aplicación nativa de firma para la realización de firmas por lotes, utilizando el mecanismo de firma por lotes desarrollado para la Junta de Andalucía a partir de la funcionalidad de firmas por lotes predefinidos JSON que proporciona originalmente Autofirma.

El presente documento está dirigido a los integradores y sirve de complemento al manual de integración original de Autofirma, ya que en este manual se definen las pautas que deben seguir para utilizar la firma por lotes cuando se utiliza Autofirma. Por lo tanto, recomendamos haber leído previamente el correspondiente manual de integración de Autofirma el cual habitualmente está disponible en el Portal de Administración Electrónica del Gobierno de España ([PAe - CTT - General - Cliente de firma electrónica de @firma](#))

## 2 INTEGRACIÓN CON FIRMA POR LOTES

La integración con la firma por lotes ofrece la ventaja de poder realizar las firmas de varios documentos a la vez a través de la app de Autofirma en dispositivos móviles (no obstante, también se puede utilizar desde Autofirma de escritorio).

Aunque Autofirma de escritorio y Autofirma App pueden realizar firmas de varios documentos de forma recursiva, el problema en dispositivos móviles surge porque por cada documento, se debe realizar una nueva llamada a la App. Esto requiere que el usuario cambie manualmente entre la página web y Autofirma para poder firmar cada vez, resultando el trámite de firma con varios documentos algo complejo para el usuario e incluso sucediendo, que, a veces, la propia App llegue a perder el foco quedándose en segundo plano.

Desde la Junta de Andalucía se han implementado varias funcionalidades añadidas al mecanismo de firma por lotes original de Autofirma ya que tenía varias carencias:

- Implementación de una funcionalidad para poder recuperar de las firmas.
- Implementación un sistema de almacenamiento de firmas en B.B.D.D.
- Implementación de un sistema de subida y bajada de contenido a firmar para poder firmar contenidos que superen los 1024 KB que tenía de límite el mecanismo original.
- Creación automatizada de los lotes de firmas.

Para implementar la firma por lotes se debe utilizar un método JavaScript que detallamos a continuación. Este método añade las cuatro funcionalidades que hemos enumerado anteriormente, más las funcionalidades originales de la firma por lotes

Para utilizar este mecanismo se necesitan dos scripts corporativos que proporciona la junta (*autoscript.js* y *batchscript.js*):

- *autoscript.js* es el cliente de firma javascript que proporciona el Estado
- *batchscript.js* incluye las funcionalidades necesarias para realizar la firma por lotes.

Las urls de los scripts son las siguientes:

Entorno de pruebas:

<https://ws024.juntadeandalucia.es/afirma-server-html-pru/js/autoscript.js>

<https://ws024.juntadeandalucia.es/afirma-server-html-pru/js/batchscript.js>

Entorno de producción:

<https://ws024.juntadeandalucia.es/afirma-server-html/js/autoscript.js>

<https://ws024.juntadeandalucia.es/afirma-server-html/js/batchscript.js>

Las urls concretas de los servicios trifásicos, intermedios y servicios de lotes corporativos están todos referenciados internamente en el script *batchscript.js*, por lo que son transparentes a nivel de integración,

es decir, no es necesario configurar específicamente las URL de servicios trifásicos, de lotes, e intermedios.

El método JavaScript que realiza la firma por lotes se denomina “doSignBatch”:

```
doSignBatch(stopOnError,commonAlgorithm,commonFormat,commonSuboperation,commonExtraparams,certFilters,documentArray,formatArray,suboperationArray,extraparamArray,showBatchFinalResultCallback,showErrorCallback);
```

El método realiza una operación de firma por lotes y recibe los siguientes parámetros:

- *stopOnError*: Indica si la firma del lote debería detenerse en el momento en el que se encuentre que una firma no es válida. Cuando se establece a “false” se indica que el proceso debe continuar incluso si alguna de las firmas del lote no puede completarse, y cuando se establece a “true” el proceso se para en el momento en el que se produce el primer error. Cuando se establece a “true”, no se guardará ninguna firma generada hasta que no se hayan generado todas. En caso de error no se guardará ninguna.
- *commonAlgorithm*: Algoritmo de firma que se utilizará por defecto para todo el lote.
- *commonFormat*: Formato de firma que se utilizará por defecto para todo el lote.
- *commonSuboperation*: Operación de firma (sign, cosign o countersign) a realizar por defecto para todo el lote.
- *commonExtraparams*: Parámetros por defecto para todo el lote
- *certFilters*: Filtros para los certificados.
- *documentArray*: Array de datos a firmar. Los datos en base64.
- *formatArray*: Array de formatos de firma a utilizar para cada dato. Si no se indica ninguno, se usará el que se indicó en commonFormat.
- *suboperationArray*: Array de operaciones de firma a aplicar sobre cada dato. Si no se indica ninguna, se usará la que se indicó en commonSuboperation.
- *extraparamArray*: Array de Parámetros a aplicar sobre cada dato. Si no se indica ninguno, se usará el que se indicó en commonExtraparams.
- *showBatchFinalResultCallback*: Función JavaScript de retorno SIN errores (Opcional: *showBatchFinalResultCallback(signatureB64, certificateB64)*).
- *showErrorCallback*: Función JavaScript de retorno CON errores (Opcional: *showErrorCallback(errorType, errorMessage)*).

Internamente, el método detecta si es llamado desde un entorno de escritorio o entorno móvil, de forma que todas las firmas que se realicen estando en un entorno de escritorio se realizarán de forma recursiva y las que se realicen estando en un entorno móvil se realizarán mediante lotes. Esto evita que, en entornos de escritorio, donde la firma recursiva tiene mejor rendimiento, se utilicen innecesariamente los servicios intermedios, trifásicos y de lotes.

Los valores que pueden recibir los parámetros *commonAlgorithm*, *commonFormat*, *commonSuboperation*, *commonExtraparams*, *certFilters*, *formatArray*, *suboperationArray* y *extraparamArray* son los mismos que pueden recibir los parámetros de los métodos de Autoscript (*algorithm*, *format*, *suboperation*, *params*, *extraparams* y *certFilters*), por lo que deberá consultarse el manual de integración de Autofirma.

Así mismo, los métodos *showBatchFinalResultCallback* y *showErrorCallback* son, respectivamente, dos funciones callback JavaScript que devuelven el resultado final y el error correspondiente en caso de que la operación no hay finalizado correctamente. Debemos recordar que la comunicación con Autofirma se realiza de forma asíncrona. Para poder gestionar esto, a todas las funciones que implican llamar a Autofirma se les debe proporcionar una función *callback* a través de la cual obtener el resultado.

La función *showBatchFinalResultCallback* devuelve, en el parámetro *signatureB64*, las firmas codificadas en base64 separadas cada una de ellas por el carácter ':' (sin comillas).



```
try{
    doSignBatch(stopOnError,commonAlgorithm,commonFormat,
    commonSuboperation, commonExtraparams , certFilters, documentArray,
    formatArray, "",
    extraparamArray,showBatchFinalResultCallback,showErrorCallback);
}
}
catch (e)
{
    showErrorCallback (e.name, e.message);
}
}
```

Nos faltaría ver un ejemplo de una función callBack.

Supongamos que tenemos una página web en html, en la cual tenemos un elemento denominado *result* donde se va a mostrar, en bruto, tanto el certificado como las firmas y también tenemos otros tres elementos denominados *firma1*, *firma2*, *firma3* donde se van a mostrar cada una de las firmas realizadas. El código sería el siguiente:

```
function showBatchFinalResultCallback(signatureB64, certificateB64 ) {

    document.getElementById('result').value = "\nCERTIFICADO:\n" +
certificateB64;

    document.getElementById('result').value += "\nFIRMAS:\n" +
signatureB64;

    var signatureArray = signatureB64.split(":");

    document.getElementById('firma1').value = signatureArray[0];
    document.getElementById('firma2').value = signatureArray[1];
    document.getElementById('firma3').value = signatureArray[2];

}
```