

Guía de Integración del componente ReFirma Invoker en Aplicaciones Web

ReFirma Invoker es un componente intermedio entre el navegador de internet y los aplicativos **ReFirma PDF** y **ReFirma PCX**; se ejecuta en el lado del cliente y su funcionalidad consiste en recibir argumentos de una aplicación web, procesar dichos argumentos para luego invocar al aplicativo ReFirma PDF o Refirma PCX y así realizar el proceso de firma digital.

De esta manera, es posible hacer firmas digitales desde aplicaciones web. Para su utilización, es indispensable instalar el aplicativo [ReFirma PDF](#) o [ReFirma PCX](#).

Antes de realizar la integración tenga en cuenta que el componente está disponible solo para entidades del estado, es necesario contar con dos identificadores (clientId y clientSecret), para lo cual deberá de ponerse en contacto con el RENIEC remitiendo un correo a identidaddigital@reniec.gob.pe.

Pasos para la integración con aplicaciones web:

Los pasos del 1 al 3 son genéricos a cualquier lenguaje de programación, el punto 4 esta presentado como ejemplo en lenguaje Java, para otros lenguajes de programación, tendrán que realizar una implementación similar. Si bien es cierto el componente está desarrollado con tecnología Java, ello no es un limitante para que sea utilizado desde otros lenguajes de programación en web.

1. Agregar el archivo JavaScript por URL (**NO DESCARGAR, REFERENCIAR POR URL**):

Para integraciones con ClickOnce:

https://dsp.reniec.gob.pe/refirma_invoker/resources/js/clientclickonce.js

Para integraciones con Java Web Start (Se recomienda realizar las integraciones con ClickOnce ya que esta tecnología ya no tiene soporte de Java para versiones superiores a Java 8):

https://dsp.reniec.gob.pe/refirma_invoker/resources/js/client.js

2. Implementar los siguientes listener:

```
<script type="text/javascript" src="https://dsp.reniec.gob.pe/refirma_invoker/resources/js/client.js"></script>
<script type="text/javascript">
  //
    window.addEventListener('getArguments', function (e) {
      type = e.detail;
      if(type === 'W'){
        ObtieneArgumentosParaFirmaDesdeLaWeb();
      }else if(type === 'L'){
        ObtieneArgumentosParaFirmaDesdeArchivoLocal();
      }
    });
    function getArguments(){
      arg = document.getElementById("argumentos").value;
      dispatchEventClient('sendArguments', arg);
    }

    window.addEventListener('invokerOk', function (e) {
      if(type === 'W'){
        MiFuncionOkWeb();
      }else if(type === 'L'){
        MiFuncionOkLocal();
      }
    });

    window.addEventListener('invokerCancel', function (e) {
      MiFuncionCancel();
    });
  //]]&gt;
&lt;/script&gt;</pre></div>
```

getArguments: Evento llamado por el componente para obtener los argumentos que procesará para la obtención y firma del documento (**Al final se detalla dichos argumentos**).

sendArguments: Evento por el cual se responde enviando los argumentos al componente.

invokerOk: Evento llamado por el componente indicando que se terminó el proceso de firma digital de manera exitosa.

invokerCancel: Evento llamado por el componente indicando que se terminó el proceso de firma digital con error o que el usuario canceló el proceso.

Type === 'W' hace referencia a que el componente va a firmar digitalmente un documento desde la web. En otras palabras, el documento es accesible por una URL. **Type === 'L'** hace referencia a que el componente va a firmar digitalmente un documento que está en la PC del usuario.

Las funciones mostradas a continuación: **ObtieneArgumentosParaFirmaDesdeLaWeb()** y **ObtieneArgumentosParaFirmaDesdeArchivoLocal()** son funciones que serán implementadas por cada programador (Los nombres de las mismas también pueden cambiar) y es desde las mismas que se cargarán los argumentos que se pasarán al componente ReFirma Invoker, estos argumentos están en formato JSON y convertido a Base64.

Una vez obtenidos los argumentos, este puede ser colocado en un hidden text (para el caso de la imagen anterior el hidden text tiene como Id “argumentos”). Una vez que se tiene los argumentos, se llamará a la función **getArguments()**.

Las funciones **MiFuncionOkWeb()**, **MiFuncionOkLocal()** y **MiFuncionCancel()**; son funciones que serán implementadas por cada programador (Los nombres de las mismas también pueden cambiar) en las cuales se puede manejar los mensajes informativos de la aplicación según el resultado del proceso de firma digital.

3. Agregar un div con id “addComponent” en el código html:

```
<div id="addComponent"></div>
```

4. Llamar al componente Refirma Invoker en el evento onclick: **onclick="initInvoker('W');"** ó **onclick="initInvoker('L');"**.

5. Es necesario crear una ruta URL a la cual el componente enviará el archivo firmado digitalmente, esta ruta tiene que recibir archivos multipart “multipart/form-data”. Como ejemplo se muestra código en Java, específicamente un servlet que recibe el archivo.

```
protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
    try {
        if (!ServletFileUpload.isMultipartContent(request)) {
            response.setStatus(HttpServletResponse.SC_PRECONDITION_FAILED);
            return;
        }

        DiskFileItemFactory factory = new DiskFileItemFactory();
        factory.setSizeThreshold(THRESHOLD_SIZE);
        factory.setRepository(new File(System.getProperty("java.io.tmpdir")));

        ServletFileUpload upload = new ServletFileUpload(factory);
        upload.setFileSizeMax(MAX_FILE_SIZE);
        upload.setSizeMax(MAX_REQUEST_SIZE);

        String uploadPath = getServletContext().getRealPath("/") + File.separator + UPLOAD_DIRECTORY;
        File uploadDir = new File(uploadPath);
        if (!uploadDir.exists()) {
            uploadDir.mkdir();
        }

        List<FileItem> formItems = upload.parseRequest(request);
        Iterator<FileItem> iter = formItems.iterator();
        while (iter.hasNext()) {
            FileItem item = (FileItem) iter.next();
            if (!item.isFormField()) {
                String idFile = item.getFieldName();//idFile asignado en los argumentos, se puede utilizar como un id.
                //String fileName = URLDecoder.decode(item.getName(), "UTF-8");
                String fileName = "firmado.pdf";
                String filePath = uploadPath + File.separator + fileName;
                File storeFile = new File(filePath);
                item.write(storeFile);
            }
        }
        response.setStatus(HttpServletResponse.SC_OK);
    } catch (Exception ex) {
        System.out.println(ex.getMessage());
        response.setStatus(HttpServletResponse.SC_INTERNAL_SERVER_ERROR);
    }
}
```

Argumentos que recibe el componente ReFirma Invoker:

Los argumentos son elaborados en formato JSON y luego convertidos a base64 para finalmente ser enviados al componente.

```
{
  "fileUploadUrl": "http://localhost:8080/demoinvoker/uploadServlet",
  "reason": "Soy_el_autor del documento",
  "pageNumber": "0",
  "maxFileSize": "5242880",
  "type": "W",
  "app": "pdf",
  "clientId": "MjU2MzIzODUxNjQ1MjU2MTZ=",
  "clientSecret": "MjU2MzIzODUxNjQ1MjU2MjZ=",
  "posx": "5",
  "dcfilter": ". *FIR. * |. *FAU. *",
  "fileDownloadUrl": "http://sp.reniec.gob.pe/app/invoker/ReFirma_Invoker_Integration.pdf",
  "posy": "5",
  "outputFile": "firmado[R].pdf",
  "protocol": "T",
  "contentFile": "firmado.pdf",
  "stampAppearancelId": "0",
  "isSignatureVisible": "true",
  "idFile": "001",
  "fileDownloadLogoUrl": "http://localhost:8080/demoinvoker/resources/img/iLogo1.png",
  "fontSize": "7",
  "fileDownloadStampUrl": "http://localhost:8080/demoinvoker/resources/img/iFirma1.png",
  "timestamp": "false"
}
```

eyJmaWxlVXBsb2FkVXJsIjoiaHR0cDovL2xvY2FsaG9zdDo4MDgwL2RlbW9pbmZva2VyL3VwbG9h
ZFnlcnZsZXQlLCJyZWZzb24iOiJB3JfZWFxfYXV0b3IgZGVsX2RvY3VtZW50byIsInBhZ2V0dW1iZXli
OilwliwibWF4RmlsZVNpemUiOiIlMjQyODgwlwidHlwZSI6IlciLCJhcHAiOiJwZGYiLCJjbGlbnRJZCI
6lk1qVTJNekl6TORVeE5qUTFNaUyTVRaPSIsImNsaVVudFNiY3JldCI6lk1qVTJNekl6TORVeE5qUT
FNalUyTWpaPSIsInBvc3giOiIlIiwizGNmaWx0ZXliOiLuKkJUIi4qfC4qRkFVLioiLCJmaWxlRG93bmX
vYWRVcmwiOiJodHRwOi8vc3AucmVuaWVjLmdvdYi5wZS9hcHAvW52b2tlci9SUZpcm1hX0lud
m9rZXJfSW50ZWdyYXRpb24ucGRmliwicG9zeSI6ljUiLCJvdXRwdXRGaWxlloiZmlybWfkbt1tSXSS
wZGYiLCJwcm90b2NvbCI6IlQiLCJjb250ZW50RmlsZSI6ImZpcm1hZG8ucGRmliwic3RhbcXBBCHBLY
XJhbmNISWQiOiOilwliwiaXNTaWduYXR1cmVWaXNpYmxlljoidHJ1ZSIsImIkRmlsZSI6IjAwMSIsImZp
bGEyb3dubG9hZExvZ29VcmwiOiJodHRwOi8vbG9jYWxob3N0OjgwODAvZGVtb2ludm9rZXIvcn
Vzb3VvY2VzL2ltZv9pTG9nbzEuG5niWiZm9udFNpemUiOiIlIiwizMlsZURvd25sb2FkU3RhbcXBVC

mwiOiJodHRwOi8vbG9jYWxob3N0OjgwODAvZGVtb2ludm9rZXIvcnVzb3VyY2VzL2ltZy9pRmlybWExLnBuZyIsInRpbWVzdGFtcCI6ImZhbHNlIn0=

[Desde este enlace](#), usted podrá acceder a un archivo comprimido, el mismo que contiene un documento detallando los argumentos que recibe el componente ReFirma Invoker y [código de ejemplo java con Maven](#), el mismo que podrá ejecutar en un Tomcat o WildFly. Además, se tiene [código de ejemplo en PHP](#).

NOTA: Los códigos de ejemplo no contienen identificadores, para su ejecución es necesario poner los mismos identificadores que le fueron enviados. Para el ejemplo en Java poner los identificadores en el archivo “default.properties”. Para el Ejemplo en PHP poner los identificadores en “config.php”.