Consumo de servicio de verificación y firma electrónica del sistema Murachí

Consumo de servicio de verificación y firma electrónica del sistema Murachí

Consumo de servicio de verificación y firma electrónica del sistema Murachí

Objetivos

Obra Publicada con Licencia Creative Commons Reconocimiento – No Comercial – Compartir Igual – 3.0 Venezuela (http://www.creativecommonsvenezuela.org.ve/)
Material Educativo basado en la Metodología de Planificación Estratégica Situacional para Espacios Comunales, desarrollados en la Fundación Centro Nacional de Desarrollo e Investigación (CENDITEL), ente adscrito al Ministerio del Poder para Educación Universitaria, Ciencia y Tecnología, ubicada en Mérida – Venezuela.

Introducción

Introducción

Introducción

Conocimiento previo

Para desarrollar las siguientes actividades del curso es necesario disponer de la configuración del ambiente de desarrolla que se realizó en la lección anterior y disponer de dispositivo criptográfico con el certificado del firmante.

Actividad

Instrucciones Generales

1. Leer con detenimiento cada una de las instrucciones.
2. Revisar los contenidos.
3. Continuar con el proceso de aprendizaje.
4. Realizar la evaluación correspondiente.

Instrucciones Específicas

La lección dispone de 3 contenidos. En seguida encontrarás las instrucciones para cada uno de ellos:

Contenido 1

- Instalar el servidor web apache2
- Descargar los paquetes necesarios para el usar el método *$ajax*
- Crear un formulario HTML que contenga un botón de tipo *Submit*
- Crear un script **JavaScript** que consuma el servicio de Versión de API Murachí

Contenido 2

- Descargar los paquetes necesarios para el usar la librería JavaScript hwcrypto
- Conocer el proceso de firmar electrónica web del sistema Murachí
- Crear un formulario que permita tomar del sistema de archivo documentos PDF
- Crear un scripr **JavaScript** que consuma el servicio de firmar electrónica PDF

Contenido 3

- Crear un formulario que permita tomar del sistema de archivo documentos PDF
- Crear un scripr **JavaScript** que consuma el servicio de verificar firma electrónica PDF

Obra Publicada con Licencia Creative Commons Reconocimiento – No Comercial – Compartir Igual – 3.0 Venezuela (http://www.creativecommonsvenezuela.org.ve/)

Material Educativo basado en la Metodología de Planificación Estratégica Situacional para Espacios Comunales, desarrollados en la Fundación Centro Nacional de Desarrollo e Investigación (CENDITEL), ente adscrito al Ministerio del Poder para Educación Universitaria, Ciencia y Tecnología, ubicada en Mérida – Venezuela.

Contenido 1: Versión del API de Murachí

Contenido 1: Versión del API de Murachí

Contenido 1: Versión del API de Murachí

Objetivos

Esta sección tiene como objetivo realizar un primera interacción con el sistema Murachí, muestra la forma en que se consume el servicio: Versión del API Murachí.

Conocimiento previo

- Manejo de protocolo web
- Nociones básicas de programación
- Programación web

Para los ejercicios se necesita una series de paquetes de javascript y lo pueden descargar en el siguiente enlace: Paquetes disponibles

Actividad

1.- Versión del API de Murachí

El primer ejercicio a realizar es consultar la versión del sistema Murachí.

Para esto, se necesita tener instalado el servidor WEB Apache y disponer del paquete "*jquery.min.js*" ya que se va a trabajar con el método *$.ajax*.

En este ejercicio se genera en *HTML* un formulario que contenga un botón tipo *Submit* que llama a una función *javascript* donde, a través del método *$.ajax* se realiza el consumo del servicio versión del API de Murachí.

Para el método **$.ajax** vamos a usar dos propiedades esenciales y obligatorias para cada una de las peticiones que se van a realizar:

- xhrFields: {withCredentials: true} esta característica puede usarse para establecer la propiedad de 'withCredentials'. Si se establece TRUE permite que se pase los cookies al servidor y permitir solicitudes de dominios cruzados.
- headers: {"Authorization":"Basic YWRtaW46YWRtaW4="} autenticación básica HTTP

Característica del formulario HTML

Control:

- Botón de envío (Submit button)

Atributos:

- action: función de javaScript que procesa el formulario

Código del formulario HTML

```
<form enctype="multipart/form-data" action="javaScript:version_Murachi()" method="post"
id="version" name="Version">
   <h2>Version de Murach&iacute;</h2>
   <br>
   <button type="submit" class="btn btn-primary">Enviar</button>
   <br>
   <div id="respuesta"> </div>
</form>
```

El elemento del formulario con id = "respuesta" se utiliza para presentar el resultado luego de procesar el formulario (consumo del servicio a Murachí)

Función javaScript version_Murachi()

Se utiliza el método $ajax para realizar la petición al servicio Murachí

Opciones de configuración de la petición:

- ulr: Establece la URL en donde se realiza la petición, para esta actividad es "**https://murachi.cenditel.gob.ve/Murachi/0.1/archivos/version**"
- type: Establece el tipo de petición, para esta actividad vamos a utilizar "**GET**"
- contentType: Establece el tipo de codificación que se va a utilziar, para esta actividad es "**application/json**"
- xhrFields y headers: ya indicado al principio de esta sección

Código de la función javaScript "version_Murachi"

```
function version_Murachi() {

   $.ajax({
      url: "https://murachi.cenditel.gob.ve/Murachi/0.1/archivos/version",
      type: "GET",
      contentType: 'application/json',
      xhrFields: {withCredentials: true},
      headers: {"Authorization":"Basic YWRtaW46YWRtaW4="},
   }).done(function(response) {
      var responseString = JSON.stringify(response);
      document.getElementById("respuesta").innerHTML = responseString;
   });
}
</script>
```

La repuesta a la solicitud se procesa por el método ***.done*** a través del argumento ***response*** donde podemos acceder al valor del JSON con la función ***JSON.stringify***

A continuación se muestra el código completo de la página HTML de esta actividad

Código de la página HTML

```
<html>
<head>
<script type="text/javascript" src="jquery.min.js"></script>
<script type="text/javascript">

function version_Murachi() {

   $.ajax({
      url: "https://murachi.cenditel.gob.ve/Murachi/0.1/archivos/version",
      type: "GET",
      contentType: 'application/json',
      xhrFields: {withCredentials: true},
      headers: {"Authorization":"Basic YWRtaW46YWRtaW4="},
   }).done(function(response) {
      var responseString = JSON.stringify(response);
      document.getElementById("respuesta").innerHTML = responseString;
   });
}
</script>
</head>
<body>
<form enctype="multipart/form-data" action="javaScript:version_Murachi()" method="post"
id="version" name="Version">
   <h2>Version de Murach&iacute;</h2>
   <br>
   <button type="submit" class="btn btn-primary">Enviar</button>
   <br>
   <div id="respuesta"> </div>
</form>
</body>
</html>
```

La respuesta del sistema Murachí indica la versión del API en un Json

Ejemplo: {"murachiVersion":"0.1.0"}

Puede descargar el código del siguiente enlace: VersionApiMurachi.html

Obra Publicada con Licencia Creative Commons Reconocimiento – No Comercial – Compartir Igual – 3.0 Venezuela (http://www.creativecommonsvenezuela.org.ve/)
Material Educativo basado en la Metodología de Planificación Estratégica Situacional para Espacios Comunales, desarrollados en la Fundación Centro Nacional de Desarrollo e Investigación (CENDITEL), ente adscrito al Ministerio del Poder para Educación Universitaria, Ciencia y Tecnología, ubicada en Mérida – Venezuela.

Contenido 2: Firmar Electrónica de Documento PDF

Contenido 2: Firmar Electrónica de Documento PDF

Contenido 2: Firmar Electrónica de Documento PDF

Objetivos

En esta sección tiene como objetivo conocer el proceso de firma electrónica Web que usa el sistema Murachí y realizar un ejercicio practico del consumo del servicio Firmar electrónica PDF.

Conocimiento previo

- Manejo de protocolo web
- Nociones básicas de programación
- Programación web

Actividad

Firma electrónica de documento usando el formato PDF

Antes de realizar el ejercicio es necesario conocer el proceso de firma electrónica web que utiliza el sistema Murachí. El diagrama de secuencia muestra el proceso de firma.

Figura 1. Diagrama de secuencia sobre el proceso de firma electrónica usando el formato PDF

Como se puede observar en el diagrama de secuencia del lado del cliente se van a realizar 3 procesos:

- **Subir documento pdf:** En este paso se carga al sistema Murachí el archivo usando el recurso ":/Murachi/0.1/archivos/" y se obtiene como respuesta el identificador único del archivo pdf (:identificador_unico_de_archivo_pdf).
- **Preparar firma del archivo:** En este caso se envía el certificado firmante al sistema Murachí junto con otros parámetros necesarios para la firma electrónica. En esta ocasión se utiliza el complemento de firma esteidfirefoxplugin (Instalado en la sección de configuración del ambiente de prueba) a través de la biblioteca de JavaScript hwcrypto (para mayor información puede visitar JavaScript hwcrypto) con el recurso :Obtener_certificado(getCertificate), para obtener del servidor el hash del archivo PDF (:hash_a_cifrar).
- **Completar la firma del archivo:** En este paso se va a enviar al sistema Murachí el hash cifrado

utilizando el recurso :Murachi/0.1/archivos/pdfs/resenas, para completar la firma electrónica. Aquí se utiliza el complemento de firma esteidfirefoxplugin a través de la biblioteca de JavaScript hwcrypto para acceder al dispositivo criptográfico y cifrar con la clave privada protegida por el dispositivo criptográfico el hash que se obtuvo en la sección anterior, usando el recurso :firmar_hash(sign). Obtenemos del sistema Murachí el identificador único del archivo firmado.

Como el servicio de firma electrónica se realiza en varios pasos vamos a dividir el ejercicio en secciones.

Sección 1: Subir documento pdf

En la primera sección se envía al sistema Murachí el archivo pdf que se va a firmar. Entonces para esta sección se requiere de un formulario que permita tomar del sistema de archivo documentos pdf y luego enviarlo con el método $.ajax usando el recurso /Murachi/0.1/archivos/

Característica del formulario HTML

Control:

- Botón de envío (Submit button)
- Botón de reinicializando (Reset button)
- Selección de fichero (file select)

Atributos:

- action: función de javaScript que procesa el formulario para esta actividad la función es “ SignFilePDF()”
- enctype: este atributo específica el tipo de contenido que sera usado para enviar los datos al servidor, para esta actividad se utilizará “multipart/form-data”

Código del formulario HTML

```
<form enctype="multipart/form-data" action="javaScript:SignFilePDF()" method="post" id="firmar" name="SignFormat">
    <h2>Firmar electr&oacute;nica (PDF)</h2>
    <h3>Seleccione el archivo que va a firmar electr&oacute;nicamente</h2>
    <br>
     <input id="file-sign" class="file" type="file" data-min-file-count="1" name="upload" accept=".pdf" >
     <br>
     <button type="submit" class="btn btn-primary">Enviar</button>
     <button type="reset" id="reset"  class="btn btn-default">Limpiar</button>
     <br>
     <br>
     <div id="seccion1"> </div>
     <br>
     <div id="seccion2"> </div>
     <br>
     <div id="seccion3"> </div>
</form>
```

Los elementos del formulario con id = "sección1", “sección2” y “sección3” se utiliza para mostrar los resultados correspondientes a los 3 procesos descritos para la firma electrónica (consumo del servicio a Murachí).

Función SignFilePDF() Sección 1. Subir documento pdf

En esta sección la función procesa los datos del formulario. En tal sentido, primero se obtiene el archivo seleccionado en el formulario a través del método document.getElementById("file-sign") donde “file-sign” corresponde al id del elemento input de tipo “file” del formulario. Luego se almacena en un objeto de tipo *FormData* que permite el envío del archivo al servidor usando el método $.ajax

Opciones de configuración de la petición $.ajax:

- ulr: Establece la URL en donde se realiza la petición, para esta sección es "**https://murachi.cenditel.gob.ve/Murachi/0.1/archivos**".
- type: Establece el tipo de petición, para esta actividad vamos a utilizar "**POST**".
- dataType: Establece el formato de la respuesta que es permitido, si el servidor devuelve información con un formato diferente al especificado el código fallará. Para este proceso se establece “**json**”.
- data: Establece la información que se enviará al servidor. Para este proceso se envía el archivo almacenado en una variable de tipo FormaData.
- contentType: Establece el tipo de codificación que se va a utilizar, para esta actividad es "**application/json**".
- processData: Establece si la información que se envía al servidor debe ser procesada a una cadena de caracteres. Para evitar esto se debe utilizar el valor “**false**”. En esta sección sólo se quiere enviar el archivo sin ser procesado.
- xhrFields y headers: ya indicado al principio de esta sección.

Código de la función javaScript "SignFilePDF()" sección 1

```
function SignFilePDF() {
    var fileInput = document.getElementById("file-sign");
    var list = fileInput.files;
    var form = $('firmar')[0];
    var data = new FormData();
    data.append('upload', $("#file-sign")[0].files[0]);
    $.ajax({
        url: "https://murachi.cenditel.gob.ve/Murachi/0.1/archivos",
        type: "post",
        dataType: "json",
        data: data,
        cache: false,
        contentType: false,
        processData: false,
        xhrFields: {withCredentials: true},
        headers: {"Authorization":"Basic YWRtaW46YWRtaW4="},
        success: function(response) {
        var responseString = JSON.stringify(response);
        document.getElementById("seccion1").innerHTML = responseString;
        }
    })
}
```

La repuesta de esta sección se procesa por el método ***.done*** a través del argumento *response* donde podemos acceder al valor del JSON con la función *JSON.stringify.*

Código completo de la sección 1:

```
<html>
<head>
<script type="text/javascript" src="jquery.min.js"></script>
<script type="text/javascript" src="gitversion.js"></script>
<script type="text/javascript" src="hwcrypto-legacy.js"></script>
<script type="text/javascript" src="hwcrypto.js"></script>
<script type="text/javascript" src="hex2base.js"></script>
<script type="text/javascript" src="jquery.min.js"></script>
<script type="text/javascript">

function SignFilePDF() {
   var fileInput = document.getElementById("file-sign");
   var list = fileInput.files;
   var form = $('firmar')[0];
   var data = new FormData();
   data.append('upload', $("#file-sign")[0].files[0]);
   $.ajax({
      url: "https://murachi.cenditel.gob.ve/Murachi/0.1/archivos",
      type: "post",
      dataType: "json",
      data: data,
      cache: false,
      contentType: false,
      processData: false,
      xhrFields: {withCredentials: true},
      headers: {"Authorization":"Basic YWRtaW46YWRtaW4="},
      success: function(response) {
         var responseString = JSON.stringify(response);
         document.getElementById("seccion1").innerHTML = responseString;
      }
   })
}
</script>
</head>
<body>
<form enctype="multipart/form-data" action="javaScript:SignFilePDF()" method="post" id="firmar"
name="SignFormat">
   <h2>Firmar electr&oacute;nica (PDF)</h2>
   <h3>Seleccione el archivo que va a firmar electr&oacute;nicamente</h2>
   <br>
   <input id="file-sign" class="file" type="file" data-min-file-count="1" name="upload"
accept=".pdf" >
   <br>
   <button type="submit" class="btn btn-primary">Enviar</button>
   <button type="reset" id="reset"  class="btn btn-default">Limpiar</button>
   <br>
   <br>
   <div id="seccion1"> </div>
   <br>
   <div id="seccion2"> </div>
   <br>
   <div id="seccion3"> </div>
</body>
</html>
```

La respuesta del servicio Murachí es el identificador único del archivo pdf, la respuesta es un json.
Ejemplo: {"containerId":"b43ea1f0-2c1a-463e-a0b8-1804e2041f33"}

Sección 2: Preparar firma del archivo

En esta sección se va a calcular el valor del hash del documento que se envió en la sección 1, en tal sentido antes del cálculo del hash se debe enviar al servidor Murachí toda la información que se agregará al documento tales como el certificado del firmante como información adicional necesaria para la firma electrónica:

- **fileId**: corresponde al identificador del archivo que se encuentra en el servidor y se desea firmar (identificador único del archivo pdf).
- **certificate**: corresponde al certificado del firmante en formato hexadecimal.
- **reason**: corresponde a la razón de la firma (cadena descriptiva del por qué de la firma).
- **location**: corresponde a la ubicación donde se realiza la firma.
- **contact**: corresponde a información de contacto del firmante.
- **signatureVisible**: true para mostrar un indicador visible de firma en la primera página del documento pdf y false para no mostrar un indicador visible de firma en la primera página del documento pdf.

El certificado lo debemos obtener desde el dispositivo criptográfico, para la misma se utiliza el complemento de firma "*esteidfirefoxplugin*" a través de la biblioteca de *JavaScript hwcrypto* y que finalmente se envía al servicio Murachí junto con los parámetros y el identificador único del contenedor.

Para trabajar con la librería **JavaScript hwcrypto**, nuestra página *HTML* debe realizar transferencias de datos seguros (*https://*), para la misma se debe instalar un certificado que respalde la identidad del servidor (https://servidordebian.org/es/squeeze/intranet/ssl_cert/self_signed). Posteriormente se configura el servidor apache2 para que tenga soporte a transferencia de datos seguros (soporte a ssl) (https://servidordebian.org/es/squeeze/internet/http/apache2_ssl).
Con la librería hwcrypto se accede al certificado del firmante que se encuentra en el dispositivo criptográfico usando el recurso *:Obtener_certificado(getCertificate)*, para luego enviarlo al sistema Murachí con el metodo *$.ajax* usando el recurso *:Murachi/0.1/archivos/pdfs*

Código de la función javaScript "SignFilePDF()" sección 2:

```
document.getElementById("seccion1").innerHTML = responseString;
 var fileId = response.fileId.toString();
 var cert;
 window.hwcrypto.getCertificate({lang: "en"}).then(function(response) {
    var cert = response;
    var parameters = JSON.stringify({
        "fileId":fileId,
        "certificate":cert.hex,
        "reason":"Certificado",
        "location":"CENDITEL",
        "contact":"582746574336",
        "signatureVisible":"true"
        });
    $.ajax({
        type: 'POST',
        contentType: 'application/json',
        url:"https://murachi.cenditel.gob.ve/Murachi/0.1/archivos/pdfs",
        dataType: "json",
        data: parameters,
        xhrFields: {withCredentials: true},
        headers: {"Authorization":"Basic YWRtaW46YWRtaW4="},
        success: function(data, textStatus, jqXHR) {
            var responseString = JSON.stringify(data);
            document.getElementById("seccion2").innerHTML = responseString;
        } // respuesta ajax 2
    }) //ajax 2
 }) //windowhwcrypto.getCertificate()
```

Si la petición tiene éxito la repuesta de esta sección se procesa a través de la función success: donde se le pase tres parámetros, data: que corresponde al resultado de la petición formateado según el valor del parámetro dataType (respuesta del servidor), textStatus: que corresponde a una cadena que describe el estado y jpXHR: que es un objeto que permite ejecutar varias funciones.

Código completo de la sección 2 (Asegure de hacer la prueba usando **https://**)

```
<html>
<head>
<script type="text/javascript" src="jquery.min.js"></script>
<script type="text/javascript" src="gitversion.js"></script>
<script type="text/javascript" src="hwcrypto-legacy.js"></script>
<script type="text/javascript" src="hwcrypto.js"></script>
<script type="text/javascript" src="hex2base.js"></script>
<script type="text/javascript" src="jquery.min.js"></script>
<script type="text/javascript">

function SignFilePDF() {

   var fileInput = document.getElementById("file-sign");
   var list = fileInput.files;
   var form = $('firmar')[0];
   var data = new FormData();
   data.append('upload', $("#file-sign")[0].files[0]);

   $.ajax({
      url: "https://murachi.cenditel.gob.ve/Murachi/0.1/archivos",
      type: "post",
      dataType: "json",
      data: data,
      cache: false,
      contentType: false,
      processData: false,
      xhrFields: {withCredentials: true},
      headers: {"Authorization":"Basic YWRtaW46YWRtaW4="},
      success: function(response) {
         var responseString = JSON.stringify(response);
         document.getElementById("seccion1").innerHTML = responseString;
         var fileId = response.fileId.toString();
         var cert;

        window.hwcrypto.getCertificate({lang: "en"}).then(function(response) {
           var cert = response;
           var parameters = JSON.stringify({
              "fileId":fileId,
              "certificate":cert.hex,
              "reason":"Certificado",
              "location":"CENDITEL",
              "contact":"582746574336",
              "signatureVisible":"true"
           });
          $.ajax({
             type: 'POST',
             contentType: 'application/json',
             url:"https://murachi.cenditel.gob.ve/Murachi/0.1/archivos/pdfs",
             dataType: "json",
             data: parameters,
             xhrFields: {withCredentials: true},
             headers: {"Authorization":"Basic YWRtaW46YWRtaW4="},
             success: function(data, textStatus, jqXHR) {
             var responseString = JSON.stringify(data);
             document.getElementById("seccion2").innerHTML = responseString;
      } // respuesta ajax 2
               }) //ajax 2
          }) //windowhwcrypto.getCertificate()
     } // respuesta ajax 1
   }) //ajax 1
}
</script>
</head>
<body>
<form enctype="multipart/form-data" action="javaScript:SignFilePDF()" method="post" id="firmar" name="SignFormat">
   <h2>Firmar electr&oacute;nica (PDF)</h2>
   <h3>Seleccione el archivo que va a firmar electr&oacute;nicamente</h2>
   <br>
   <input id="file-sign" class="file" type="file" data-min-file-count="1" name="upload" accept=".pdf" >
   <br>
   <button type="submit" class="btn btn-primary">Enviar</button>
   <button type="reset" id="reset"  class="btn btn-default">Limpiar</button>
   <br>
   <br>
   <div id="seccion1"> </div>
   <br>
   <div id="seccion2"> </div>
   <br>
   <div id="seccion3"> </div>
   <br>
   <div id="seccion4"> </div>
</form>
</body>
</html>
```

Respuesta:

Field	Type	Description
hash	String	Reseña o hash del archivo que se debe cifrar con la clave privada protegida por el dispositivo criptográfico.

La respuesta del sistema Murachí es el valor del hash del archivo pdf que se subió al servidor en la sección 1.

E j e m p l o
:{"hash":"a3fbfc857e0796b5e5e9fdb8e3183d8e532afe0fe662f3a24eb459f0c22bd472","error":""}

Posibles errores del servicio Murachí:

```
HTTP/1.1 400 Bad Request
{
  "hash": "",
  "error": "El archivo que desea firmar no es un PDF."
}

HTTP/1.1 401 Unauthorized
{
  "error": "acceso no autorizado"
}

HTTP/1.1 500 Internal Server Error
{
  "hash": "",
  "error": "error en carga de certificado de firmante"
}
```

Posibles errores al usar la librería JavaScript hwcrypto:

* **"user_cancel"** - El usuario a cancelado la operación.
* **"no_certificates"** - No hay certificado disponible para el complemento de firma.
* **"technical_error"** - Error técnico inesperado del complemento de firma
* **"no_implementation"** - No se ha encontrado el complemento de firma (No instalado)
* **"not_allowed"** - ejecución no permitida por no usar https
* **"pin_blocked"** - El dispositivo criptográfico esta bloqueado

Sección 3: Completar la firma del archivo
En esta sección se firma el hash del documento a firmar con el certificado firmante almacenado en el dispositivo criptográfico. En tal sentido partimos desde el resultado que obtuvimos en la sección 2 (el valor del hash del archivo pdf).
El cifrado del hash se realiza dentro del dispositivo criptográfico; para la misma vamos a volver a utilizar el complemento de firma "*esteidfirefoxplugin*" a través de la biblioteca de *JavaScript hwcrypto* usando el recurso *:firmar_hash(sign)*. Finalmente se envía el hash cifrado con el método *$.ajax* al sistema Murachí para completar la firma del archivo pdf

Función SignFilePDF() Sección 3. Completar la firma del archivo

(Se parte desde la sección anterior donde se obtuvo el hash del archivo)

Opciones de configuración de la petición $.ajax:

- url: Establece la URL en donde se realiza la petición, para esta sección es "**https://murachi.cenditel.gob.ve/Murachi/0.1/archivos/pdfs/resenas**".
- type: Esatblece el tipo de petición, para esta sección vamos a utilizar "**POST**".
- dataType: Establece el formato de la respuesta que es permitido, si el servidor devuelve información con un formato diferente al especificado el código fallará. Para esta sección se establece "**json**".
- data: Establece la información que se enviará al servidor. Para esta sección se enviá el hash que se obtiene de la siguiente manera: JSON.stringify({"signature":signature.hex}).
- contentType: Establece el tipo de codificación que se va a utilizar, para esta sección es "**application/json**".
- xhrFields y headers: ya indicado al principio de esta sección.

Código de la función javaScript "SignFilePDF()" sección 3:

```
document.getElementById("seccion2").innerHTML = responseString;
 var json_x = data;
 var hash = json_x['hash'];
 alert("hash recibido del servidor "+hash);
 var hashtype = "SHA-256";
 var lang = "eng";
 window.hwcrypto.sign(cert, {type: hashtype, hex: hash}, {lang: lang}).then(function(signature) {
   $.ajax({
       type: 'POST',
        contentType: 'application/json',
        url:"https://murachi.cenditel.gob.ve/Murachi/0.1/archivos/pdfs/resenas",
        dataType: 'json',
        data: JSON.stringify({"signature":signature.hex}),
        xhrFields: {withCredentials: true},
        headers: {"Authorization":"Basic YWRtaW46YWRtaW4="},
        success: function(data, textStatus, jqXHR){
            var responseString = JSON.stringify(data);
            document.getElementById("seccion3").innerHTML = responseString;
            alert('Archivo firmado correctamente: ' + data['signedFileId']);
           document.getElementById("seccion4").innerHTML = "Descargar archivo firmado:
https://murachi.cenditel.gob.ve/Murachi/0.1/archivos/descargas/" + data['signedFileId'];
        } // repuesta del ajax 3
    }) //ajax 3
 }) //windowhwcrypto.sign()
```

Si la petición tiene éxito la repuesta de esta sección se procesa a través de la función success: donde se le pase tres parámetros, data: que corresponde al resultado de la petición formateado según el valor del parámetro dataType (respuesta del servidor), textStatus: que corresponde a una cadena que describe el estado y jpXHR: que es un objeto que permite ejecutar varias funciones.

Código completo de la sección 3

```
<html>
<head>
<script type="text/javascript" src="jquery.min.js"></script>
<script type="text/javascript" src="gitversion.js"></script>
<script type="text/javascript" src="hwcrypto-legacy.js"></script>
<script type="text/javascript" src="hwcrypto.js"></script>
<script type="text/javascript" src="hex2base.js"></script>
<script type="text/javascript" src="jquery.min.js"></script>
<script type="text/javascript">

function SignFilePDF() {

   var fileInput = document.getElementById("file-sign");
   var list = fileInput.files;
   var form = $('firmar')[0];
   var data = new FormData();
   data.append('upload', $("#file-sign")[0].files[0]);

   $.ajax({
      url: "https://murachi.cenditel.gob.ve/Murachi/0.1/archivos",
      type: "post",
      dataType: "json",
      data: data,
      cache: false,
      contentType: false,
      processData: false,
      xhrFields: {withCredentials: true},
      headers: {"Authorization":"Basic YWRtaW46YWRtaW4="},
      success: function(response) {
         var responseString = JSON.stringify(response);
         document.getElementById("seccion1").innerHTML = responseString;
         var fileId = response.fileId.toString();
         var cert;

      window.hwcrypto.getCertificate({lang: "en"}).then(function(response) {
         var cert = response;
         var parameters = JSON.stringify({
            "fileId":fileId,
            "certificate":cert.hex,
            "reason":"Certificado",
            "location":"CENDITEL",
            "contact":"582746574336",
            "signatureVisible":"true"
         });
         $.ajax({
            type: 'POST',
            contentType: 'application/json',
            url:"https://murachi.cenditel.gob.ve/Murachi/0.1/archivos/pdfs",
            dataType: "json",
            data: parameters,
            xhrFields: {withCredentials: true},
            headers: {"Authorization":"Basic YWRtaW46YWRtaW4="},
            success: function(data, textStatus, jqXHR) {
            var responseString = JSON.stringify(data);
            document.getElementById("seccion2").innerHTML = responseString;

            var json_x = data;
            var hash = json_x['hash'];
            alert("hash recibido del servidor "+hash);
           var hashtype = "SHA-256";
           var lang = "eng";

           window.hwcrypto.sign(cert, {type: hashtype, hex: hash}, {lang: lang}).then(function(signature) {

             $.ajax({
                type: 'POST',
                contentType: 'application/json',
                url:"https://murachi.cenditel.gob.ve/Murachi/0.1/archivos/pdfs/resenas",
               dataType: 'json',
               data: JSON.stringify({"signature":signature.hex}),
               xhrFields: {withCredentials: true},
               headers: {"Authorization":"Basic YWRtaW46YWRtaW4="},
               success: function(data, textStatus, jqXHR){
               var responseString = JSON.stringify(data);
               document.getElementById("seccion3").innerHTML = responseString;
               alert('Archivo firmado correctamente: ' + data['signedFileId']);
               document.getElementById("seccion4").innerHTML = "Descargar archivo firmado: https://murachi.cenditel.gob.ve/Murachi/0.1/archivos/descargas/" + data['signedFileId'];
             } // repuesta del ajax 3
          }) //ajax 3
       }) //windowhwcrypto.sign()
      } // respuesta ajax 2
    }) //ajax 2
  }) //windowhwcrypto.getCertificate()
} // respuesta ajax 1
}) //ajax 1
}
</script>
</head>
<body>

<form enctype="multipart/form-data" action="javaScript:SignFilePDF()" method="post" id="firmar" name="SignFormat">
   <h2>Firmar electr&oacute;nica (PDF)</h2>
   <h3>Seleccione el archivo que va a firmar electr&oacute;nicamente</h2>
   <br>
   <input id="file-sign" class="file" type="file" data-min-file-count="1" name="upload" accept=".pdf" >
   <br>
   <button type="submit" class="btn btn-primary">Enviar</button>
   <button type="reset" id="reset"  class="btn btn-default">Limpiar</button>
   <br>
   <br>
   <div id="seccion1"> </div>
   <br>
   <div id="seccion2"> </div>
   <br>
   <div id="seccion3"> </div>
   <br>
   <div id="seccion4"> </div>
</form>
</body>
</html>
```

La respuesta del servicio Murachí sería el identificador único del archivo firmado.

Ejemplo:

{"signedFileId":"847948fb-a5d5-46d3-8176-9c474b22e57b-signed.pdf"}

Para descargar el archivo del sistema Murachí se agregar la dirección https://murachi.cenditel.gob.ve/Murachi/0.1/archivos/descargas/ más el identificador único del archivo firmado.

Ejemplo:
https://murachi.cenditel.gob.ve/Murachi/0.1/archivos/descargas/847948fb-a5d5-46d3-8176-9c474b22e57b-signed.pdf

Se copia este enlace y se coloca en un navegador para descargar el archivo desde el servidor Murachí.

Puede descargar el código en el siguiente enlace: FirmaElectronicaPDF.html

Obra Publicada con Licencia Creative Commons Reconocimiento – No Comercial – Compartir Igual – 3.0 Venezuela (http://www.creativecommonsvenezuela.org.ve/)
Material Educativo basado en la Metodología de Planificación Estratégica Situacional para Espacios Comunales, desarrollados en la Fundación Centro Nacional de Desarrollo e Investigación (CENDITEL), ente adscrito al Ministerio del Poder para Educación Universitaria, Ciencia y Tecnología, ubicada en Mérida – Venezuela.

Contenido 3. Verificar Firma Electrónica

Contenido 3. Verificar Firma Electrónica

Contenido 3. Verificar Firma Electrónica

Objetivos

En esta sección tiene como objetivo realizar un ejercicio practico del consumo del servicio verificar firma electrónica PDF.

Conocimiento previo

- Manejo de protocolo web
- Nociones básicas de programación
- Programación web

Actividad

Realización de la verificación de una firma electrónica PDF:

Para este ejercicio se genera un formulario desde una página HTML que permita cargar y enviar un archivo PDF firmado al servicio de Murachí.
En esta sección se requiere de un formulario que permita tomar del sistema de archivo solo documentos PDF y luego enviarlos al sistema Murachí usando el método $.ajax

Característica del formulario HTML

Control:

- Botón de envío (Submit button)
- Botón de reinicializando (Reset button)
- Selección de fichero (file select)

Atributos:

- action: Función de javaScript que procesa el formulario para esta sección la función es “verifySing()”
- enctype: Este atributo específica el tipo de contenido que será usado para enviar los datos al servidor, para esta sección se utilizará “multipart/form-data”

Código del formulario HTML

```
<h2>Verificar la firma electr&oacute;nica</h2>
<form enctype="multipart/form-data" action="javaScript:verifySing()" method="post"
id="verificar">
   <br>
   <input id="file-check-sign" class="file" type="file" data-min-file-count="1" name="upload"
accept=".pdf">
   <br>
   <button type="submit" class="btn btn-primary">Enviar</button>
   <button type="reset" id="resetVerificacion" class="btn btn-default">Limpiar</button>
   <br>
   <div id="respuestaVerificacion"> </div>
</form>
```

El elemento del formulario con id = "respuestaVerificacion" se utiliza para presentar el resultado luego de procesar el formulario (consumo del servicio a Murachí)

Función javaScript verifySing()

En esta sección la función procesa los datos del formulario. En tal sentido, primero se obtiene el archivo seleccionado en el formulario a través del método document.getElementById("file-check-sign") donde “file-check-sign” corresponde al id del elemento input de tipo “file” del formulario. Luego se almacena en un objeto de tipo FormData que permite el envío del archivo al servidor usando el método $.ajax

Opciones de configuración de la petición:

- ulr: Establece la URL en donde se realiza la petición, para esta sección es "**https://murachi.cenditel.gob.ve/Murachi/0.1/archivos/firmados**"
- type: Establece el tipo de petición, para esta actividad vamos a utilizar "**POST**"
- dataType: Establece el formato de la respuesta que es permitido, si el servidor devuelve información con un formato diferente al especificado el código fallará. Para este proceso se establece “**json**”.
- data: Establece la información que se enviará al servidor. Para este proceso se envía el archivo almacenado en una variable de tipo FormaData.
- contentType: Establece el tipo de codificación que se va a utilizar, para esta sección es "**application/json**"
- xhrFields y headers: ya indicado al principio de esta sección.

Código de la función javaScript "version_Murachi”

```
function verifySing() {
    var formData = new FormData();
    formData.append("upload", $("#file-check-sign")[0].files[0]);
    $.ajax({
        url: "https://murachi.cenditel.gob.ve/Murachi/0.1/archivos/firmados",
        xhrFields: {withCredentials: true},
        headers: {"Authorization":"Basic YWRtaW46YWRtaW4="},
        type: "post",
        dataType: "json",
        data: formData,
        cache: false,
        contentType: false,
        processData: false,
        success: function(response) {
    var responseString = JSON.stringify(response);
    document.getElementById("respuestaVerificacion").innerHTML = responseString;
          }, error: function(response) {
            $("#respuestaVerificacion").html('Error...!!!');
            }
      });
}
```

Si la petición tiene éxito la repuesta de esta sección se procesa a través de la función success: donde se le pase tres parámetros, data: que corresponde al resultado de la petición formateado según el valor del parámetro dataType (respuesta del servidor), textStatus: que corresponde a una cadena que describe el estado y jpXHR: que es un objeto que permite ejecutar varias funciones.

Código completo

```
<html>
<head>
<script type="text/javascript" src="jquery.min.js"></script>
<script type="text/javascript">

function verifySing() {
   var formData = new FormData();
   formData.append("upload", $("#file-check-sign")[0].files[0]);
   $.ajax({
      url: "https://murachi.cenditel.gob.ve/Murachi/0.1/archivos/firmados",
      xhrFields: {withCredentials: true},
      headers: {"Authorization":"Basic YWRtaW46YWRtaW4="},
      type: "post",
      dataType: "json",
      data: formData,
      cache: false,
      contentType: false,
      processData: false,
      success: function(response) {
   var responseString = JSON.stringify(response);
   document.getElementById("respuestaVerificacion").innerHTML = responseString;
         }, error: function(response) {
           $("#respuestaVerificacion").html('Error...!!!');
           }
     });

}
</script>
</head>
<body>
<h2>Verificar la firma electr&oacute;nica</h2>
<form enctype="multipart/form-data" action="javaScript:verifySing()" method="post"
id="verificar">
   <br>
   <input id="file-check-sign" class="file" type="file" data-min-file-count="1" name="upload"
accept=".pdf">
   <br>
   <button type="submit" class="btn btn-primary">Enviar</button>
   <button type="reset" id="resetVerificacion" class="btn btn-default">Limpiar</button>
   <br>
   <div id="respuestaVerificacion"> </div>
</form>
</body>
</html>
```

La respuesta del servicio Murachí es un Json con la información de la(s) firma(s) electrónica(s) del archivo PDF que subió al sistema.

Ejempĺo:

{"fileId":"2c6813f5-a4bd-49f5-93f2-b5d904fa5f9a","fileExist":"true","numberOfSignatures":1,"signatures":[{"signatureType":"approval","signedOn":"22-05-201714:26:29.00","integrityCheck":"true","timeStamp":"null",
"reason":"Certificado","location":"CENDITEL","alternativeNameOfTheSigner":"",
"signerCertificateValidFrom":"2015-06-03 20:00:00.00","signerCertificateStillValid":"true",
"signatureCoversWholeDocument":"true","filterSubtype":"adbe.pkcs7.detached",
"signerCertificateSubject":"C=VE,ST=Merida,L=Merida,O=CENDITEL,OU=GIDSI,CN=Pedro Buitrago,E=pbuitrago@cenditel.gob.ve",
"signerCertificateValidAtTimeOfSigning":"true", "encryptionAlgorithm":"RSA", "timeStampService":"null", "digestAlgorithm":"SHA256","certificatesVerifiedAgainstTheKeyStore":"true",
"documentRevision":"1","nameOfTheSigner":"Pedro Buitrago","totalDocumentRevisions":"1",
"contactInfo":"582746574336", "timeStampVerified":"null", "signerCertificateIssuer":"C=VE,ST=Merida,L=Merida,
O=CENDITEL,OU=GIDSI, CN=Autoridad de Certificación del GIDSI,E=acgidsi@cenditel.gob.ve",
"signerCertificateValidTo":"2017-06-03 19:59:59.00"}], "mimeType":"application/pdf"}

Puede descargar el código desde el siguiente enlace: VerificarFirmaPDF.html

Obra Publicada con Licencia Creative Commons Reconocimiento – No Comercial – Compartir Igual – 3.0 Venezuela (http://www.creativecommonsvenezuela.org.ve/)
Material Educativo basado en la Metodología de Planificación Estratégica Situacional para Espacios Comunales, desarrollados en la Fundación Centro Nacional de Desarrollo e Investigación (CENDITEL), ente adscrito al Ministerio del Poder para Educación Universitaria, Ciencia y Tecnología, ubicada en Mérida – Venezuela.

Refuerzo

Refuerzo

Refuerzo

Actividad de lectura

Instrucciones.

En seguida encontrarás documentación adicional que enriquece el conocimiento obtenido en las actividades realizadas anteriormente. Te invitamos a revisar los siguientes enlaces.

- Documentación de la interfaz de programación del servicio (API) web Murachí API Murachi
- Otra documentación de como usar el API de Murachí Como usar el api
- Información sobre javascript hwcrypto
- Instalar un certificado en el servidor (https://servidordebian.org/es/squeeze/intranet/ssl_cert/self_signed).
- Configurar el servidor apache2 para que tenga soporte a transferencia de datos seguros (soporte a ssl) https://servidordebian.org/es/squeeze/internet/http/apache2_ssl.

Obra Publicada con Licencia Creative Commons Reconocimiento – No Comercial – Compartir Igual – 3.0 Venezuela (http://www.creativecommonsvenezuela.org.ve/)
Material Educativo basado en la Metodología de Planificación Estratégica Situacional para Espacios Comunales, desarrollados en la Fundación Centro Nacional de Desarrollo e Investigación (CENDITEL), ente adscrito al Ministerio del Poder para Educación Universitaria, Ciencia y Tecnología, ubicada en Mérida – Venezuela.