MiDNI

Manual de integración

Índice de contenido

Objetivos y alcance

En objetivo de este documento es detallar el API del servicio de extracción y validación de datos para elementos QR proporcionados por la aplicación MiDNI. El objetivo de este documento es (1) definir las necesidades que un desarrollador externo va a requerir y (2) la forma en la que ha de invocar un desarrollador externo y (3) las respuestas que puede esperar de este servicio.

Queda fuera del alcance de este documento detallar el funcionamiento interno de este servicio. Es decir, este documento tratara el servicio como si fuera una caja negra, detallando las salidas que se pueden esperar tras proporcionarle unas entradas concretas.

Librería Java JAR

En esta sección se describe el servicio en su versión de librería Java JAR. (1) En esta sección se describen los requisitos y dependencias de este servicio, (2) la forma de configurar y realizar una invocación a través de un *snippet* de código y (3) la descripción de las potenciales respuestas que se pueden obtener del mismo.

Requisitos y dependencias

Dada la necesidad de usar este JAR como base del servicio REST desarrollado para **IZWS/midni** este JAR se ha generado haciendo uso de la versión de JDK *Java 1.8.0 181-b13*.

Adicionalmente, se hace uso de las librerías de *Bouncy Castle* para la verificación de la firma de los datos incluida en el QR. Para ello en el desarrollo se ha hecho uso de la siguiente librería

```
<dependency>
    <groupId>org.bouncycastle</groupId>
    <artifactId>bcprov-jdk15on</artifactId>
    <version>1.70</version>
</dependency>
```

Configuración e invocación del servicio

Para usar esta librería es necesario realizar dos pasos: (1) configurar y crear una instancia de validador y (2) usar esa instancia para validar el contenido de un QR.

La configuración del validador ser realiza a través de una instancia de la clase ***ValidadorConfig***. En la versión actual de la librería solo se puede configurar la ruta del certificado que contiene la clave publica con la que validar la firma que la DGP ha realizado para los datos del QR. El código para esta inicialización puede ser algo similar al siguiente

```
final ValidadorConfigCertPublicoData certPublico = new ValidadorConfigCertPublicoData();
certPublico.setModo(Modo.PATH);
certPublico.setValor(<<path al certificado>>);

final ValidadorConfig config = new ValidadorConfig();
config.setCertPublico(certPublico);
```

Una vez se tiene la instancia de la clase ***ValidadorConfig*** que contiene la configuración del validar a usar, el siguiente paso es generar una instancia de la clase ***Validador***.

```
final Validador validador = new Validador(config);
```

Con esta instancia de validador se puede extraer y validar el contenido de los QR proporcionando el contenido del mismo tanto en formato de array de bytes como en base 64

```
final RespuestaValidacionDto resultado = validador.validar(<<contenido del QR>>);
```

En la respuesta proporcionada por esta funcionalidad se retornan tanto los errores de validación. El contenido de esta respuesta se describirá en más detalle en la siguiente sección.

Descripción de la respuesta

A continuación, se muestra un ejemplo de potencial respuesta que este servicio puede proporcionar.

```
RespuestaValidacionDto(
          errores=[
                    ErrorValidacion(
                              tipo=FIR_ESTADO_NOVALIDO
                              , info=FIRMA_NO_VERIFICADA
                    ), […]
          ], datos=DatosValidacionDto(
                    cabecera=DatosCabeceraValidacionDto(
                              magicNumber=0xDC
                              , version=V04
                              , pais=ES
                              , identificadorFirmante=ESPN20
                              , referenciaCertificado=2274948240B9368F65E5C80FEBFE5CE4
                              , fechaEmision=Tue Apr 16 00:00:00 CEST 2024
                              , fechaFirma=Tue Apr 16 00:00:00 CEST 2024
                              , referencia=SIMPLE
                              , categoriaDocumento=DNI_ES_MOVIL
                    ), mensaje=DatosMensajeValidacionDto(
                              items=[
                                        ItemMensajeValidacionDto(
                                                  etiqueta=EtiquetaItemMensajeValidacionDto(
                                                            hex=0x40
                                                            , tag=NUMERO_DOCUMENTO
                                                  ), tamagno=9
                                                  , contenido=99999999R
                                        ), […]
                              ]
                    ), firma=DatosFirmaValidacionDto(
                              contenido=mIHk[…]
                              , estado=FIRMA_NO_VERIFICADA
                              , certificado=DatosCertificadoValidacionDto(
                                        dn=CN=APPDNIMOVIL[…]
                                        , serial=4D393EEC9AD3289964D22FB9F744A884
                              )
                    )
          )
)
```

Cabe mencionar que este servicio devolver todos los datos que sea capaz de extraer del QR independientemente de que estos datos sean coherentes o no. Es decir, la utilidad devolverá obtenidos del QR incluso si no son válidos (por ejemplo, porque se haya vulnerado la firma del QR). En estos casos, junto con los datos se devolverá una serie de elementos de error

El servicio en ningún caso va a decidir si se pueden o deben utilizar los datos retornados. Será responsabilidad del usuario de la funcionalidad decidir, se haya retornado o no elementos de error, decidir si hace uso de los datos devueltos o no.

Dicho esto, **la repuesta contendrá dos propiedades**. Las propiedades serán

- **errores**: Un listado de los errores de validación. Estos errores se componen a su vez de dos propiedades. El **tipo** de error que se ha detectado en el proceso de validación del QR y, en caso de ser relevante, **información** adicional sobre el error detectado.
  Se puede ver el catalogo completo de los potenciales errores de validación del proceso en el *Anexo I: Catalogo de errores de respuesta de servicio*.
- **datos**: Los datos que se pueden extraer del propio QR de MiDNI

Los datos de la respuesta tendrán tres propiedades. Las propiedades serán

- **cabecera**: En ese bloque se devuelven los valores de la cabecera del QR de MiDNI. Se pueden entender estas propiedades como metadatos del propio QR. No se entrará en el detalle de estos valores dado que ya están explicados en el documento para desarrolladores de MiDNI.
- **mensaje**: Este bloque se devuelven los ítems encontrados en la sección de mensaje de QR. Se puede interpretar (tomándose algunas libertades) el mensaje de un QR como un mapa clave valor. Cada ítem de esta lista representa uno de estos valores.
- **firma**: En este bloque se devuelve información sobre la firma del QR

Cada ítem del mensaje se compone de las siguientes propiedades.

- **etiqueta**: El código del ítem en cuestión. A nivel de QR es un código hexadecimal. Para facilitar su interpretación esta funcionalidad asignará un nombre semántico a todos los ítems conocidos.
- **tamaño**: El número de bytes que ocupa en el QR el valor del ítem.
- **contenido**: La representación en formato de cadena de caracteres del valor del contenido del ítem.

La firma de los datos contiene información sobre la firma del QR. Esta sección tiene las siguientes propiedades

- **contenido**: La representación binaria de la firma en base 64.
- **estado**: El estado de la validación de la firma. Puedes consultar el catálogo de posibles estados en el *Anexo II: Estado de firma*.
- **certificado**: Datos sobre el certificado con la parte clave pública el firmante. En esta versión del servicio se devuelve el DN del certificado y su número de serie.

Fotografía de la persona

Entre los ítems que se pueden extraer del mensaje, con la etiqueta con el código hexadecimal 0x50, se puede encontrar la imagen de la fotografía usada para el carnet de identidad. El contenido de este ítem son los bytes de la imagen en formato JP2 (o JPEG-2000).

El formato JP2 no es un formato nativo de máquinas Windows. En caso de que quiera usarse en estas máquinas puede ser necesario convertir este formato a otro que este tipo de maquinas puedan usar (como podría ser JPG o PNG). Esta conversión queda fuera del alcance de esta funcionalidad y correspondería al invocante transformarlo al formato que crea adecuado.

Anexo I: Catalogo de errores de respuesta de servicio

La siguiente tabla muestra el catálogo de potenciales errores de validación del servicio.

	Tipo	Desripción
ERRORES DE PROCESADO	B64_NO_VALIDO	No se puede decodificar el String proporcionado como contendido en base 64 del QR de MiDNI. Habría que comprobar que el String es un elemento base64 válido.
	BYTES2DATE_EXCEPTION	No se puede procesar un array de bytes como una fecha. En el caso de que se produzca este error, en el **info** se informará del campo que no se ha podido procesar.
	BYTES2LONG_EXCEPTION	No se puede procesar un array de bytes como un numérico. En el caso de que se produzca este error, en el **info** se informará del campo que no se ha podido procesar.
	BYTES2STR_EXCEPTION	No se puede procesar un array de bytes como una cadena de caracteres. En el caso de que se produzca este error, en el **info** se informará del campo que no se ha podido procesar.
	C40_EXCEPTION	No se puede procesar un array de bytes con el formato C40. En el caso de que se produzca este error, en el **info** se informará del campo que no se ha podido procesar.
	IO_EXCEPTION	Se ha producido un error de tipo IO en la lectura de bytes. En el caso de que se produzca este error, en el **info** se informará del campo que no se ha podido procesar.
ERRORES DE CABECERA	CAB_CATTIPODOCUMENTO_NOVALIDA	El valor de la categoría del tipo de documento no corresponde con un QR de tipo MiDNI
	CAB_IDENTIFICADORFIRMANTE_NOVALIDA	Se ha obtenido un identificador de firmante nulo. Es necesario que el QR de MiDNI contenga un identificador de firmante.
	CAB_MAGICCONSTANT_NOVALIDA	El valor de la *magic constant* no corresponde con un QR de tipo MiDNI. Deberia ser **OxDC**.
	CAB_PAIS_NOVALIDA	El valor del país no corresponde con un QR de tipo MiDNI. Debería ser **ES**.
	CAB_REFERENCIA_NOVALIDA	El valor del país no corresponde con un QR de tipo MiDNI. Debería ser **7** (simple), **8** (completo) o **9** (mayoría de edad).
	CAB_VERSION_NOVALIDA	El valor de la versión no corresponde con un QR de tipo MiDNI. Debería ser **0x03**.
ERRORES DE FIRMA	FIR_DATOS_NOVALIDOS	No se puede validar la caducidad de los datos. O la fecha de validez es nula o ya ha sido superada.
	FIR_ESTADO_NOVALIDO	La firma no ha podido ser validada. En caso de que se haya producido un este error, en el **info** se informará del estado de la firma. Consultar el *Anexo II: Estado de firma* para conocer el catálogo de estados.
	FIR_NUMSERIE_NOVALIDO	El número de serie del certificado usado para validar la firma no coincide con la referencia del certificado de la cabecera.
ERRORES DE MENSAJE	MSG_CONTENIDO_NOVALIDO	El contenido de un elemento del mensaje no es válido según el valor esperado para su etiqueta. En el caso de producirse este error, en el **info** se informará del valor hexadecimal de la etiqueta problemática. Este error puede recibirse múltiples veces.
	MSG_HEX_NORECONOCIDO	En el mensaje se ha encontrado una etiqueta cuyo valore hexadecimal es desconocido para un mensaje de MiDNI. En el caso de producirse este error, en el **info** se informará del valor hexadecimal de la etiqueta problemática. Este error puede recibirse múltiples veces.

Anexo II: Estado de firma

La siguiente tabla muestra el catálogo de potenciales estados de firma.

Estado	Detalle
ALGORITMO_FIRMA_NO_VALIDO	El algoritmo de firmado usado no corresponde con el que debería tener una firma del contenido de un QR de MiDNI. El algoritmo de firma debería ser ***SHA256withECDSA***
CERTIFICADO_NULO	No se ha podido obtener el certificado de la parte pública con la que validar la firma
CLAVE_FIRMA_INCORRECTA	La clave pública obtenida del certificado de es del tipo que debería tener para poder validar una firma el contenido de un QR de MiDNI. El algoritmo de firma debería ser ***ECPublicKey***
ERROR_DE_LECTURA	Se ha producido un error de tipo IO verificando la firma
ERROR_EN_FIRMA	Se ha producido un error en el procesado de la firma
FIRMA_NO_VERIFICADA	Se ha terminar correctamente el proceso de verificación de la firma y **la firma no es válida**.
FIRMA_VERIFICADA	Se ha terminar correctamente el proceso de verificación de la firma y **la firma es válida**.